

IBM Customer Engineering Education
Student Guide

7090 Data Processing System

IBM Customer Engineering Education
Student Guide

7090 Data Processing System

PREFACE

This Study Guide is designed for field training the 407 U.R., 729 Customer Engineer on the 7090 Data Processing System. It is not a "Self-Study" course, although some individual sections of the guide are, and others approach it in many respects. It is intended for use in a formal lecture field training course.

The Study Guide is made up of "Objectives" for you to meet, "Reading Assignments" where you can find the information to meet the objectives, and "Review Questions" for you to check yourself to see if you are meeting the objectives. Extra figures are in the Study Guide to illustrate points where the reference manuals have not. Answers to the "Review Questions" are in the "Answer Section" of the Study Guide. Check your answers after completing a section. In this way the answered questions will serve as review.

Address comments concerning the contents of this publication to:
IBM Corporation, CE Education, Dept. 911, Poughkeepsie, New York 12602

7090 STUDENT STUDY GUIDE

CONTENTS

<u>Section</u>	<u>Description</u>	<u>Page</u>
<u>01.00</u>	<u>INTRODUCTION AND BASIC PROGRAMMING</u>	
01.01	Computer Characteristics	
01.02	Data Representations	
01.03	Numbering Systems	
01.04	Instruction Specs. and Symbolic Coding	
01.05	Arith. Inst. and Logical Ops.	
01.06	Control Ops., Indexing and Trap	
01.07	I/O Operations	
01.08	Introduction to FAP	
01.09	Programming in FAP	
01.10	FORTRAN (Self-Study)	
<u>02.00</u>	<u>PACKAGING AND COMPONENT CIRCUITS</u>	
02.01	Packaging	
02.02	Component Circuits	
<u>03.00</u>	<u>POWER SUPPLIES</u>	
03.01	Locations and Controls	
03.02	Standard Modular Supply	
03.03	Core Storage Power Supplies	
03.04	Power Sequencing	
03.05	Marginal Checking	
03.06	Power Converter	
03.07	Miscellaneous Circuits	
03.08	Trouble Shooting	
<u>04.00</u>	<u>CYCLE AND INSTRUCTION</u>	
04.01	Timing Circuits	
04.02	Basic CPU Circuitry Operation	
04.03	Cycle Generation	
04.04	I Cycle	
04.05	Indexing	
04.06	Indirect Addressing	
04.07	Manual Control Operations	
04.08	Diagnostics	
<u>05.00</u>	<u>PRELIMINARY INSTRUCTIONS</u>	
05.01	Data Flow	
05.02	Fixed Point Arith. and Word Transmission	
05.03	Shift Instructions	
05.04	Transfer Instructions	
05.05	Control Instructions	
05.06	Skip Instructions	

Section	Description	Page
	05. 07 Indexing Instructions	
	05. 08 Trap Mode	
	05. 09 I/O Control Instructions	
	05. 10 Sense Indicator Instructions	
	05. 11 Trouble Shooting Problems	
<u>06. 00</u>	<u>7302 OIL CORE STORAGE (SEE NOTE)</u>	
	06. 01 General Concept of Core Storage	
	06. 02 Ferrite Core Theory	
	06. 03 Coincident Current Addressing	
	06. 04 Basic Core Cycle and Addressing	
	06. 05 Sense Amps, Registers and Timing	
	06. 06 Special Circuits	
	06. 07 CE Panel and Temperature Control	
	06. 08 Diagnostic - 9S1	
<u>07. 00</u>	<u>ADVANCED INSTRUCTIONS (SELF-STUDY)</u>	
	07. 01 Fixed Point MPY/DIV	
	07. 02 Intro. to Exponential Math and 7090 Floating Point Numbers	
	07. 03 Floating Point Instructions	
	07. 04 Logical AND/OR Instructions	
	07. 05 Convert Instructions	
<u>08. 00</u>	<u>DATA CHANNEL</u>	
	08. 01 Introduction and I/O Programming	
	08. 02 Functional Units	
	08. 03 Tape Write Select Operation	
	08. 04 Tape Read Select Operation	
	08. 05 TCH-1A Operations	
	08. 06 Non-Data Select Operations	
	08. 07 Data Channel Trap	
	08. 08 Console Operations	
	08. 09 Data Channel Miscellaneous	
	08. 10 9T 55 - Diagnostic	
	08. 11 Introduction to CAU	
	08. 12 Basic CAU	
	08. 13 Card Machine Power Distribution	
	08. 14 Read Card Reader	
	08. 15 Write Punch	
	08. 16 Printer Operations	
	08. 17 Sense Operations	
	08. 18 Manual Operations	
	08. 19 Simulated Machine Error Analysis Problems	
<u>09. 00</u>	<u>FIGURES</u>	
<u>10. 00</u>	<u>REVIEW QUESTION ANSWERS</u>	
	01. 00 Answers	
	02. 00 Answers	
	03. 00 Answers	

NOTE: 7302A Air Memory employs the use of the "7302A Self-Study Guide"
Form #R23-2904

<u>Section</u>	<u>Description</u>	<u>Page</u>
04.00	Answers	
05.00	Answers	
06.00	Answers	
08.00	Answers	
<u>11.00</u>	<u>LAB GUIDE</u>	
<u>12.00</u>	<u>"PIM"</u>	
	PIM is a Programming Instructional Manual for use in the Basic Programming sections of the course (01.00)	

LEGEND - LIST OF COMMON ABBREVIATIONS & SYMBOLS

SR	Storage Register
SC	Shift Counter
ACC	Accumulator
ADD	Adders
MQ	Multiplier/Quotient Register
PR	Program Register
PC	Program Counter
L	Core Location
L _X =	Contents of Core Location "X"
Char.	Characteristic
Diff.	Difference
^C SR	Characteristic of the number in the Storage Register
^C ACC	Characteristic of the number in the Accumulator
^C MQ	Characteristic of the number in the Mult/Quotient Reg.
^F SR	Fraction of the number in the Storage Register
^F ACC	Fraction of the number in the Accumulator
^F MQ	Fraction of the number in the Mult/Quotient Reg.
F. P. T.	Floating Point Trap
unfl.	Underflow
ovfl.	Overflow
\gt	Greater than
\geq	Greater than or equal to
\lt	Less than
\leq	Less than or equal to
2(^F SR)	2 times the fraction of the storage register
PIM	Programming Instruction Manual (Section 12.00 of this S. G.)
IR	Inst. Ref. Manual
SMS	Standard Modular System - Inst. Ref. Manual
TCC	Transistor Component Circuits Manual of Inst.
SG	Student "Study Guide"

01.00 INTRODUCTION & PROGRAMMING

TABLE OF CONTENTS

01.01	Computer Characteristics
01.02	Data Representations
01.03	Numbering Systems
01.04	Instruction Specifications and Symbolic Coding
01.05	Arithmetic Instructions and Logical Operations
01.06	Control Operations, Indexing Concepts and Trap
01.07	Input-Output Operations
01.08	Introduction to FAP
01.09	Programming in FAP

INTRODUCTION Read this!

This is a programming course. It is not intended to teach you how the machine "hardware" works (though it will make it easier for you to learn it later). Do not hypothesize how the hardware does an operation when you are studying the machine operations codes. You should learn exactly how to go about using the op codes in writing programs and how they affect data and control. But I repeat, "Don't get involved in circuitry!" You'll get more than enough of that later.

This study guide is made up of "objectives" for you to meet, "reading assignments" where you can find the information and "review questions" for you to check yourself to see if you are meeting the objectives. Answers to the questions are in the Answer Section of the Study Guide. Check your answers after completing a section. The review questions can be used as review material in this way.

OK, let's go!!

01.01 COMPUTER CHARACTERISTICS

&

01.02 DATA REPRESENTATIONS

Objectives: After the introduction lecture and the following reading assignments, you should be able to do the following without reference.

1. Describe a 7090 system including I/O units and be able to list the numbers of each unit, its function, and speed, and data flow between units.
2. Name and describe all CPU registers as to size, use and data flow.
3. Describe the format of data as it appears on cards, tape and in core.

Reading Assignment: Read "Computer Characteristics" pages 1-11 in the Programming Instruction Manual (hereafter called "PIM")

Review Questions - Computer Characteristics:

1. Draw a block diagram of a 7090 system. Fill in frame numbers and show basic data flow between frames. page 260
2. There are 5 models of Data Channels available on a 7090.
3. MOD III and IV Data Channels have 90KC tape drive feature.
4. MOD I and III Data Channels have Card Adapter Units in addition to TAU.
5. MOD ✓ Data Channel has CAU only. All other model channels can have up to 10 tape drives.
6. Card machine speeds are:
 - a. Printer 150 lines/min.
 - b. Punch 100 cards/min.
 - c. Reader 250 cards/min.
7. Basically the two functions of the Central Processing Units (CPU1 and CPU2) are _____ and _____.
8. Multiplexor is like a "Traffic Cop". It is a switching unit through which data flow is directed. (True or False) T
9. The Core Storage Unit capacity is 32 words. Each word is made up of 36 bit positions.
10. The computer cannot distinguish between a "data" word or "instruction" word in core. Whether a word is used as an instruction or as data depends on the kind of cycle it is brought out on. (True or False) T
11. There are 4 types of machine cycles. They are IEBL.
12. Instructions are brought out of core storage and decoded during an I cycle.
13. The Data Channels initiate B cycles to transfer data and commands to or from storage.
14. E and L cycles are used to "do" an instruction. E cycles make access to core storage to transfer data, but L cycles never do.

15. Most, but not all, instruction words contain the operation "code" in bit positions _____.
16. The address field of a word is in positions 21-35.
17. Draw a simplified diagram of all processing unit registers. Label each and show bit positions each contains. Use arrows to show basic data flow between registers.

01.02 DATA REPRESENTATIONS

Reading Assignment: Read Pages 11 - 21 in PIM

Review Questions - Data Representations

18. The 7090 utilizes "Row Binary" format to read or punch cards. The first 36 bit word occupies _____. The second is in _____ and so on for a total of _____ words per card.
19. Since 2 7090 words are punched in each row only 72 columns of a card are used.
20. In order to punch Hollerith information in a card, it is necessary to set up a card image in core storage by means of a decoder.
21. An "assembly program" which takes a program you have written in "mnemonic" (alphabetic op codes) and converts it to machine language (binary words) utilizes the entire _____ of a row binary card for information about the card. This information is used by a program called a "loader" to read your program into Core Storage.
22. The _____ which tells the "loader" how many words are punched in the card is contained in cols _____ of _____ row. There may be from one to _____ words to be read from each card.
23. The "address" field is in cols. _____ of 9 row. It specifies where in core the loader is to place the 1st word, which is in _____ of the card. The 2nd word, which is in _____ of the card will be placed in the next higher address in core and so on for the rest of the words to be read.
24. The _____ in 9 row left of a card is a logical addition of all the words in the card except _____. This is used by the _____ to check for bit pickup or dropout during the readin operation.
25. The 36 bit 7090 word allows the placement of _____ BCD characters within each word. 729 tape drives utilize BCD coding. Since only whole 36 bit words are written on tape by the 7090, records are always at least _____ or multiples of _____ BCD characters.

01.03 NUMBERING SYSTEMS

Objectives: Without reference you should be able to:

1. Convert numbers from the decimal system to octal and to binary and vice versa.
2. Read binary numbers and verbally express them octally.
3. Do simple arithmetic using binary numbers.

Reading Assignment: Read "Appendix A" pages 131 to 135 of the 7090 Reference Manual

Review Questions - Number Conversions

1. Manually convert 1466 decimal to octal _____ then to binary _____.
2. Convert 32,769 decimal to octal _____
3. Convert 100 octal to decimal _____
4. Convert .25 decimal to octal _____
5. Convert .000968 dec. to oct. _____
6. Convert .001 oct. to dec. _____
7. Convert .350 oct to dec _____
8. Convert 40000 oct. to dec. _____

Reading Assignment: Beginning at Page 136 of 7090 Reference Manual, you'll find Appendix B and on Page 140 Appendix C. Use these and see if you can find the numbers used in the preceding "review questions".

9. Express the following binary numbers octally.
 - a. 0110111101 _____
 - b. 11000101 _____
 - c. 1000 _____
 - d. 11011011 _____
 - e. 10000000 _____
10. Write the four rules for subtracting and the three rules for addition of binary numbers.

01.04 INSTRUCTION SPECIFICATIONS AND SYMBOLIC CODING

Objectives: - Instruction Specs (without reference)

1. Describe the use of each field of a symbolic coding form and how they correspond to what is to be punched in cards.
2. Define the symbols, terms and abbreviations used to represent and describe instructions and registers.
3. Correctly write on a symbolic coding form for program assembly.

Reading Assignment: Read Pages 22 - 24 in PIM and Page 18 in 7090 Ref. Man. ("Programming Examples" on page 108 of Ref. Man. may also be of help)

Review Questions - Instruction Specs.

1. Each line of a coding form is punched into _____.
2. The coding form is layed out in cols. This constitutes the card cols. where the information is to be punched. (True or False)
3. The _____ field, which may be blank, an actual address, or a symbolic address occupies cols. 1 to _____.
4. Col. _____ must be _____ to separate the location field from the operation field. A blank col. also separates the operation field and _____ field.
5. The _____ field begins in Col.8 and may be from _____ to _____ characters.
6. The _____ field may contain nothing, an address, address and tag, or address tag and decrement, depending on what is required by the _____.
7. If there is no variable field where should comments begin? _____
8. Cols.73-80 are used for card numbering or identification. The computer does not read these cols. (True or False)
9. In a description of an instruction the symbol Y is used to denote that an _____ is required.
10. Commas in the variable field are used to _____.
11. In the instruction ADD*. The * means _____. Be careful, an * in the variable field means something entirely different.

12. Define:
- a. C(Y)
 - b. C(MQ)
 - c. C(AC)³⁻¹⁷
 - d. C(AC)

13. The letter X in an instruction denotes use of _____.

01.05 ARITHMETIC INSTRUCTIONS AND LOGICAL OPERATIONS

Objectives - Fixed Point Arithmetic

Without reference

1. Be able to describe for all fixed point instructions
 - a. Where all data to be worked on must be located.
 - b. What operation is performed on the data
 - c. What data is destroyed if any
 - d. Where and in what form is the "answer" or result of the operation

2. Be able to write the correct mnemonic op code and format for all fixed point arithmetic instructions on a coding form (using the 7090 reference card only)

NOTE: In addition to review questions, you will also be given problems to program in the remainder of this guide. They will progress from very simple routines to more comprehensive programs which you will code on a form and some of which you will assemble and execute on the 7090.

Not all instructions will be covered. To locate information on any instruction, use "Appendix G" page 152 or "Appendix E" page 144 of the 7090 ref. manual.

Reading Assignment: Read "Fixed-Point Arithmetic Instructions" Page 25 - 27 in PIM. (Use 7090 Ref. Man. if you need further reference material)

Review Questions - Fixed Point

14. A _____ instruction destroys the old C(AC) and places the C(Y) in _____.

15. A carry into pos. _____ turns on the AC overflow indicator. Carries out of pos. Q are _____.

16. If the product of a multiply does not exceed 35 bits, what register would it be in?

17. The answer from a divide operation will be in the _____. The remainder, if any, is in the _____.

18. If division is to take place, the C(Y) must be _____ than the C(AC). If not, the _____ indicator is turned on and division doesn't take place since the numbers involved will not produce a correct answer.

Reading Assignment: Read Pages 34 and 35 of PIM "Word Transmission Operations"

19. Write a simple program routine for the following. Use actual addresses as shown. Begin program in 201.

These quantities are stored in memory as fields of a part's inventory:

<u>Location</u>	<u>Contents</u>
4001	Receipts
4002	Withdrawals
4003	Adjustments
4004	On Order
4005	Reserve

- a. Compute "Stock Balance" and "Availability"
- 1) Stock Bal = Receipts - Withdrawals + Adjustments
 - 2) Availability = Stock Bal. + On order - Reserve
- b. Store: Stock Balance in location 4006
Availability in location 4007
20. Assume X is a number in storage location 400.
- All results will be assumed to have values that will not exceed the capacity of a single word. Start programming at location 100.
- a. Place X^3 in location 410
 - b. Place X^4 in location 411
 - c. Place X^6 in location 412
21. The integers a, b, c, d, are stored in symbolic locations A, B, C and D respectively.

The quantity X is stored in symbolic location X.

Place $ax^3 + bx^2 + cx + d$ in location X.

Assume that the results will not exceed the capacity of one storage location. Begin the program at location START.

22. Write a simple program routine to do the formula: $\frac{A}{B} = Q$

Assume symbolic addresses and that quantities won't exceed capacity of one storage location. Q initially contains all zeros.

Objectives - Floating-Point Arithmetic and Shifting

You must be able to do the following without reference:

1. Describe the format of a floating point word.
2. Convert any fixed point number to floating point and vice versa.
3. Define when it is necessary to use floating point arithmetic.
4. Define the use and results of shift instructions.
5. Be able to define:
 - a. The format and whereabouts of data required by each floating point operation.
 - b. The format and location of data resulting from any floating point operation.
6. Be able to use F. P. and shift instructions in programs.
7. Be able to describe "Masking" and "Packing".

Reading Assignment: Read Pages 27 - 30 of PIM; "Floating Point Operations"
Page 7 and Page 135 of 7090 Ref. Man.

Review Questions - Floating-Point And Shift

23. In a floating point word, the sign of the fraction is in pos _____ and the sign of the characteristic is denoted by pos _____.
24. The fraction part of a F. P. word is in pos. _____ to _____, a normalized fraction having its high order bit in _____.
25. A F. P. fraction with an exponent of zero has a characteristic of _____.
26. A "normal" F. P. zero has _____ for a char. and _____ for its fraction.
27. FAD results are always normalized. In fact it may be used to normalize a F. P. number by adding it to all zeros in the Accumulator. (True or False)
28. The _____ significant part of the results of a FAD appears in the MQ with a characteristic _____ octal _____ than the _____.
29. In dealing with signs, both fixed and floating point instructions follow the same rules of arithmetic in assigning a sign to the results as you did in grammar school. (True or False)
30. If the results of a FMP do not exceed the size of one FP word, which register would you "store" the results from? _____

31. In both fixed and floating point divide the results appear in the AC.
(True or False) _____
32. Shifting a number left one place is the same as _____ by _____.
Shifting a number right one place is the same as _____ by _____.
33. ALS of one place with a 35 bit fixed point number in the AC will _____
the size of the number. If a floating point number were in the AC, the
results would be _____.
34. On what shift instruction do the following occur?
_____ a. Bits from MQ sign go in AC35
_____ b. Bits from AC35 are lost
_____ c. Bits from MQ sign go in MQ35
_____ d. Bits from MQ35 are lost
35. X_1, X_2, X_3, X_4, X_5 are normalized floating point numbers in locations
400, 401, 402, 403, and 404, respectively. Write a routine to evaluate
 $\frac{X_1 X_2 X_3}{X_4 X_5}$
and store in location 410. If division cannot take place, stop the
calculator (assume numbers small enough to be contained in one word).
Use octal for addressing.

NOTE: When writing a program on a symbolic coding form for assembly,
you may use either symbolic addressing and/or actual (numeric) addresses.
It is easier to use symbols for addresses. However, you must define these
by using them in the location field of the form (only once). If you ever desire
to use actual addresses, THEY MUST BE DECIMAL. The assembly program
will automatically convert the numbers from decimal to octal for you (whether
you like it or not).

All further programs should be written using the above rules.

Reading Assignment - "Logical Operations" - Pages 30-32 in PIM
"Packing and Unpacking" - Pages 32 - 34 PIM

Review Questions:

36. In an "AND" function _____ positions being AND'ed must be _____
to get a result of one.
37. In an "OR" function _____ or _____ positions being a one will result
in one out.
38. In an "Exclusive OR" function the corresponding positions _____ (must,
must not) match to get a one. In other words, one position must be a
_____ and the other a _____ to get a one.

39. One use of logical instructions is _____. That is, "blotting" out what you don't want and keeping only the portion of the word you want. Another use is to put two or more pieces of data into one word to save space. This is called _____.
40. Given the following program, describe what you think it's doing.

SQUEEZ	CLA	MASK
	ANS	DATA
	CLA	PACK
	ALS	18
	ORS	DATA
	TSX	OUT, 2
DATA	TRA	1000
MASK	OCT	777777
PACK	OCT	050000

01.06 CONTROL OPERATIONS, INDEXING CONCEPTS and TRAP

Objectives:

1. Be able to describe the operation of and use in a program of:
 - a. Conditional and Unconditional transfers
 - b. Skip Instructions
 - c. Control Instructions
2. Without reference, be able to describe the concept of indexing.
3. Be able to use and describe all index transmission instructions without references.
4. Be able to describe the operation of multiple TAGing without reference.
5. Be able to describe and use indexing and indexing loops without reference.
6. Without reference, be able to describe transfer trap mode operation, ETM, LTM, TTR.
7. Be able to describe the operation of Floating Point Trap.
8. Be able to describe the use of Sense Indicator Reg. and sense indicator instructions.
9. Be able to describe indirect addressing.
10. Be able to describe the process and find the resulting address for any combination of "Tagging" and "Flagging".

Reading Assignment - "Sense Indicator Operations" Pages 47 and 48 in PIM.
(Refer to 7090 Ref. Man. for more SI ops)

Review Questions - "Sense Indicator Operations"

1. The SI register is a _____ bit register. This register is used for masking and also as indicators for setting and testing by the programmer for monitoring many conditions.
2. _____ of SI instructions transmit a full 36-bit word between the SI Reg and either the AC or storage they are _____. The remaining _____ SI instructions are classified by five functions. They are:
 - A. _____
 - B. _____
 - C. _____
 - D. _____
 - E. _____
3. Describe the following:
 - A. LDI _____
 - B. STI _____
 - C. PIA _____
 - D. RIA _____
 - E. TIO _____

Reading Assignment: "Control Instructions". Read Pages 48 to 50 in PIM.
(Refer to Appendix G or E in 7090 Ref. Man. for further information if needed)

Review Questions: "Control Operations"

4. _____ transfer instructions will always alter the normal sequential processing of instructions without testing.
5. _____ transfer instructions may or may not transfer depending on some condition being met. These should not be confused with _____ type instructions which do not really transfer but "skip" over one or more instructions.
6.
 - a. TZE will transfer if _____.
 - b. TMI will transfer if _____.
 - c. TOV will transfer if _____.
 - d. TNO will transfer if _____.
 - e. TNZ will transfer if _____.
 - f. TPL will transfer if _____.
 - g. TQP will transfer if _____.
 - h. TLQ will transfer if _____.

7. An XEC instruction is something like a transfer that goes out to the address specified and does that instruction, BUT IT THEN COMES BACK to the next instruction after where the XEC was. Look up XEC in 7090 Ref. Man. and see if you can describe what's happening in the following programs.

A. Where will the 7090 halt _____?
 What's in the AC when it halts _____?

```

START  CLA  ONE
        ADD  TWO
        XEC  XRAY
        STO  ANS
STOP   HTR  START
ONE    HTR  1
TWO    HTR  6
XRAY   ARS  ARS1
GO     TRA  XRAY
  
```

B. Where will the 7090 halt _____?
 What's in the AC when it halts _____?

```

START  CLA  EQU
        ADD  HIGH
        XEC  XRAY
        ADD  LOW
ONE    HTR  START
TWO    HTR  START
XRAY   CAS  LOW
HIGH   HTR  6
EQU    HTR  1
LOW    HTR  10
  
```

C. Where will the 7090 halt _____?
 What's in the AC when it halts _____?

```

START  CLA  ONE
        ADD  TWO
        STO  ANS
        XEC  XRAY
        CLA  TWO
TWO    HTR  2
ONE    HTR  1
XRAY   TRA  DO
        ADD  TWO
DO     TNZ  ONE
END    HTR  START
  
```

8. Three quantities are stored in locations ONE, TWO, and THREE. Using the TQL instruction, write a routine to determine which is the lowest number and store it in LOW.

9. When would you use a NOP instruction?

Reading Assignment: "Indexing Concept" and "Comp. Arith."
"Multiple Tags" 35 to 39 in PIM. (7090 Ref. Man. Page 44)

Review Questions: "Indexing Concepts"

10. Indexing is the combining of the contents of an _____ with the _____ portion of an instruction before the instruction is executed.
11. By indexing an instruction we can change the _____ which that instruction works on. If we were to add up a list of a 1000 numbers we need not use 1000 ADD instructions; only one.
12. There are _____ index regs. They may be loaded with either _____ or _____ numbers. When combined with the address part of an instruction, the address may be _____ or _____.
13. The index regs are lettered _____. Which one is to be used to combine with the address portion of an instruction is specified by the _____ field of the instruction which is positions _____.
14. One or any combination of index regs can be used by an instruction. A tag of 1 specifies index reg _____, a tag of 6 specifies index reg _____. This is called _____ tagging.
15. Each index reg is _____ positions long. This is the same size as the the address field of a 7090 word _____ (True, False).
16. _____ arithmetic is used by the indexing process. This means that the contents of the index reg is always _____ from the instructions address field. If an instruction has an address field containing 7 and we specify with a tag an index reg which contains a "True" 3, the resultant address will be _____. If, instead, we wanted to add 3 to the address, we would put the _____ of 3 in the index reg. This is like subtracting a -3 from the address which is the same as _____ +3 to the address.
17. If multiple tags are used, the contents of the index regs involved are _____ together before combining with the address field of an instruction. This can be quite involved since the result is not easily seen. If $XRA = 763_8$, $XRB = 061_8$ and $XRC = 770_8$, the result of using a tag of 7 would mean that _____ would be subtracted from the address. If the address were 6774, the _____ address would be _____.

Reading Assignment: Read "Index Transmission Operations" Page 39 to Page 41 and "Index Register Testing Instructions" Page 50 to 52 in PIM. (Refer to Appendix G in 7090 Ref. Man.)

(OK..... Let's see if you can use indexing in some programs. The way you program a problem may not agree with the way it's given on the answer sheet. As long as it works.....OK. In fact, you may do it a better way.)

18. Do the following problems using index registers.

- (a) X is an integer in storage location X. Place X^9 in location XNINE. (Assume X^9 will not exceed one word)
- (b) Twenty-four numbers are stored in locations NUM to NUM + 23. Place the sum of those that are positive in location sum. Assume that the sum cannot exceed the capacity of a single storage location.
- (c) Thirty-one numbers are stored in locations NUM to NUM + 30. Place the sum of those that are positive in location PSUM and those that are negative in MSUM.

19. Fifty numbers $X_1 X_2 - - - X_{50}$ are stored in locations X to X + 49. Fifty other numbers $Y_1 Y_2 - - - Y_{50}$ are stored in locations Y to Y + 49.

Place $X_1 - Y_1$ in location XY, $X_2 - Y_2$ in XY + 1, etc.

Assume that difference cannot exceed the capacity of a single storage location. Constants needed may be assumed to be stored beginning at location CONS.

20. There is a list of 100 numbers in locations X - 100 to X-1. Using CAS and indexing write a routine to sort the list into ascending order. Assume no two numbers are equal.

Reading Assignment: "Indirect Addressing" Page 11 in 7090 Ref. Man and Page 41 to 46 in PIM.

- 21. Flag bits in a 7090 word are bits in pos _____ and _____.
- 22. On a coding form indirect addressing is indicated by a _____ after the last _____ of the _____.
- 23. (a) What is the sum of the addition? _____
(b) Where will the sum be stored? _____

DATA	CLA*	STOP
STOP	ADD	GO
	STO*	STOP
GO	HTR	10

- 24. (a) Where will the following halt? _____
(b) Where will the "STO" store? _____

DATA	CLA*	STOP
STOP	ADD	GO
	STO*	TRAN
GO	HTR	10
TRAN	TRA*	DONE
DONE	HTR	5

25. Contents of which location will the CLA place in the AC? _____

	CLA*	DO, 4	XRA = 3
	ADD	NUM, 4	XRB = 2
DO	STO	STOP	XRC = 1
STOP	HTR	5	
NUM	HTR	3	

Reading Assignment: "Transfer Trap Mode" Page 11 and ETM, LTM, TTR on Pages 36 and 37 of 7090 Ref. Man., EFTM and LFTM Page 66 of 7090 Ref. Man.

26. Transfer trap mode is used mainly for program debugging. It is a means of keeping track of how far a program runs by storing the address of each transfer it encounters as the program runs. (True or False)

27. The program beginning at START is running:

00	HTR	000	XRA = 000
01	STO	SAVE	XRB = 6
02	CLA	000	XRC = 1
03	STA	05	
04	CLA	SAVE	
05	TTR*	000	
SAVE	HPR	000	
START	ENT		
	CLA	NUM	
	ADD	NUM + 1	
	STO	DATA	
WOW	TRA*	GO, 4	
NUM	HTR	5	
	HTR	6	
DATA	TRA	000	
TRAP	HTR	GO	
GO	LTM		
DONE	HTR	START	

- A. Where will the program halt first? _____
 What will be in the address portion of location 05? _____
- B. If the start key is depressed, where will the program halt again? _____
 Now, what will be in the address portion of 05? _____
- C. Will the HTR at DONE cause a trap to occur? _____

Reading Assignment: Read "Floating Point Trap" Page 79 of PIM Page 30 in 7090 Ref. Man.

28. The capacity of a floating point word is exceeded if the "exponent" goes beyond _____ or below _____. In the machine this would be a "characteristic" above _____ and below _____.

29. Beyond the upper limit of the characteristic is an _____ condition and below the lower limit is called a _____.
30. When a floating point trap occurs due to an underflow or overflow, the machine puts the _____ of the _____ that caused the trap in the _____ of _____.
31. An identifying or _____ code is placed in the _____ portion of 0000 by FP trap and the computer executes the instruction located at _____.
32. Describe the meaning of each of the FP spill code bits.

Bit in 14 _____
 Bit in 15 _____
 Bit in 16 _____
 Bit in 17 _____

01.07 INPUT-OUTPUT OPERATIONS

Objectives: With reference to 7090 Ref. Man. be able to:

1. Describe the buffering and asynchronous operation of data channel.
2. Describe the functions of these basic data channel registers:
 - a. Data Register
 - b. Op. Reg (Indicator Reg.)
 - c. Word Counter
 - d. Address Counter
 - e. Location Counter
3. Describe the difference between a command and an instruction.
4. Describe the format of a command word.
5. Describe and use the following to write I/O routines.
 - a. WRS
 - b. RDS
 - c. LCH
 - d. RCH
 - e. SCH
 - f. TCH
 - g. BTT, ETT
 - h. WEF, TEF
 - i. REW
 - j. TCO, TCN
 - k. TRC
6. Describe and use the eight I/O commands single or in any combination.

7. Describe D. C trap as to how it is enabled, what conditions cause a trap and what the results of trap are.

Reading Assignment: Read from Page 55 to Page 79 in PM.
(Refer to 7090 Ref. Man. for more detailed information)

Review Questions - I/O Operations

1. Data transmission operations of a data channel are initiated by the execution of _____ instructions by CPU. A select instruction to select and start the channel followed by a _____ to load a _____ into the channel registers. From this point the channel operates like a separate computer. This is called _____ operation.
2. Data channel controls the _____ and _____ of all data transmitted between _____ and _____.
3. Draw a diagram of a command word showing its format as to WC, etc.
4. What regs of D. C are stored by SCH?
5. A WRS or RDS instruction has a different op code for each data channel. (True or False)
6. Which I/O unit (tape drive, printer, etc.) is determined by the _____ of the instruction?
7. There are eight commands. See if you can list and describe at least six of them without reference.
8. A bit in 19 of a command means _____ of data if used on a read operation and is _____ on a write operation.
9. What condition will cause each of the following to transfer?
 - a. TCH
 - b. TCO
 - c. TCN
 - d. TRC
 - e. TEF
10. What does a bit in 18 of a command mean? _____
11. If the following command is given, assuming the channel is RDS, where will the first word be stored?

0100	IOCP*	200,	0,	100
0200	TCH*	500,	0,	0
12. What will be the configuration of the record(s) on tape, using the following program?

WRS	1221	0100	IOCP	0,	0,	50
RCHA	100		IORP	1000,	0,	0
HTR			IOSP	2000,	0,	50
			IORP	3000,	0,	20
			IOCD	0,	0,	0

13. a) How many words will be written on tape with the following program?
 b) Using a read select, how many words will be read from the tape that was just written?

WRS	1221	100	IOCP	WC = 50
RCHA	100	101	IOCPN	WC = 30
LCHA	110	102	IORPN	WC = 100
		103	IORTN	WC = 25
		110	IOCD	WC = 4

14. Three records on tape:

1st Rec. = 25 words
 2nd Rec. = 15 words
 3rd Rec. = 20 words

Which words will be read into storage as a result of the following program?

RDS	1201	100	IOCP	WC = 5
RCHA	100	101	IORP	WC = 10
LCHA	110	102	IOSP	WC = 5
		103	IOCD	WC = 20

15. What are the three conditions that may initiate a D.C trap?

A. _____
 B. _____
 C. _____

16. The contents of CPU _____ is stored by a DC trap in some even location between 12 and 30 (octal) dependent on which data channel caused the trap. Channel C would store in loc _____.

17. If channel C caused a trap, CPU would take its next instruction from location _____. If that instruction were not an unconditional transfer, CPU would then go _____.

18. Once a D.C trap occurs, all other traps are prevented until a _____ instruction is given.

02.00 PACKAGING AND COMPONENT CIRCUITS

02.01 PACKAGING

Objectives: Without reference we must be able to accomplish the following:

1. List the pin identification letters and numbers for single and double cards.
2. List the voltage connection pins for single and double SMS cards including the voltage that will be found on each pin.
3. Define the use of the following line connectors:
 - a. Edge Connectors
 - b. Hinge Connectors
 - c. Panel Connectors
 - d. T-Connectors
 - e. Slide Connectors
 - f. *Coaxial Connectors
4. Draw a side view of a 30" SMS sliding gate. Included on this drawing should be the following:
 - a. Panels
 - b. Rows
 - c. Columns
 - d. Edge Connectors
 - e. T-Connectors
 - f. Hinge Connectors
 - g. Panel Connectors
5. List the lead assignments for Edge Connectors and Panel Connectors.
6. Given any 7090 ALD address we must be able to physically locate this point on the machine. An example of this would be locating the following addresses:
 - a. 02D3611H
 - b. 03F21E12
 - c. 01B3T03B
 - d. 02E46H-R
7. Given a SMS card address we must be able to locate this logic block in the ALD's.

Reading Assignment: SMS Manual Pages 5-12 and 17-21, Study Guide Figures 02.01 through 02.04.

Review Questions:

1. Each frame contains _____ gates. The _____ sliding gates are lettered _____. The tailgates are lettered _____.

* More commonly known as biscuit connectors

2. What is the function of the tailgates?
3. The rows on each panel are labeled _____ and the columns _____.
4. Edge connectors are used whenever _____.
5. "T" connectors are labeled _____ for each column and are used whenever _____.
6. SMS card sockets are labeled _____.
7. Interpanel (1 to 2 or 3 to 4) signal wires use pins _____ of the card socket and the ground wires use pins _____.
8. Ribbon cables from row K of panel 3, gate D will be plugged into gate _____, slide connectors columns _____ and _____, rows _____ to _____.
9. Why are the slide connector pins for columns A, B, F, and G labeled left to right and columns C, D, H, and J labeled right to left?
10. _____ and _____ position biscuit connectors are used on cables to the tailgates.
11. Double SMS card contacts are labeled _____ and _____.
12. Voltage connections to the SMS card are through pins _____ to _____.

02.02 COMPONENT CIRCUITS

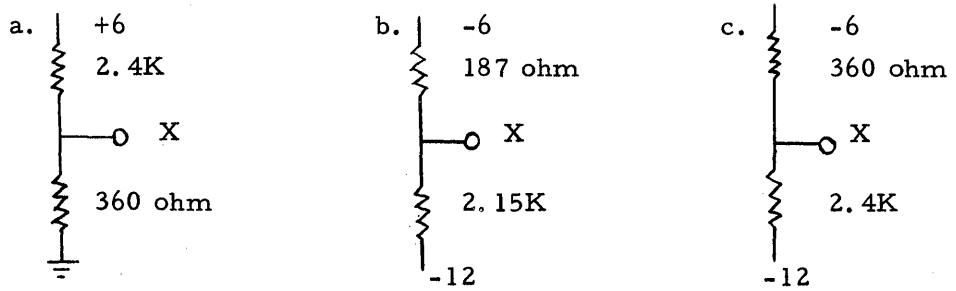
Objectives: Without reference we must be able to accomplish the following:

1. Define the nominal voltage swing of a P or N level.
2. Define the reference level of a P or N line.
3. Draw the output waveforms of any 7090 logic block or group of blocks when given the inputs.
4. Identify the conducting transistors of any 7090 logic block when given the schematic drawing and input voltages.
5. Determine the forward or reverse bias status of the collector to base and base to emitter junction of a PNP or NPN transistor when given the collector, base, and emitter voltages.
6. Define the nominal inherent block delays for alloy and diffused junction logic blocks.

Reading Assignments: SMS Manual Pages 5-6, 19-22, 45-55, 36-41, 89-90 & 56-69
Study Guide Figure 02.05.

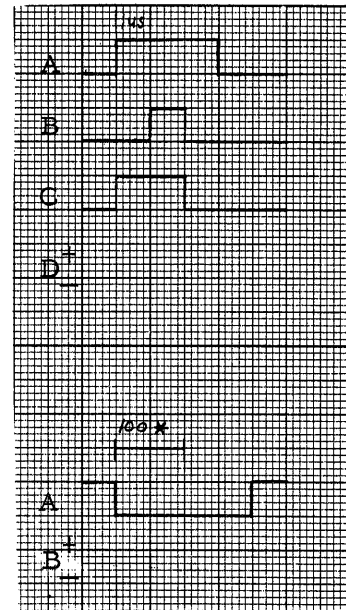
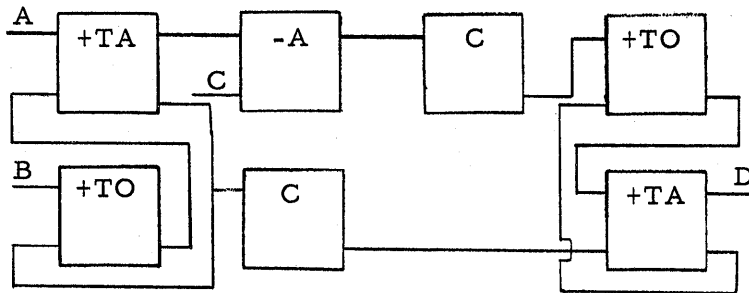
Review Questions:

1. What is the voltage at point X of the following circuits with reference to ground?

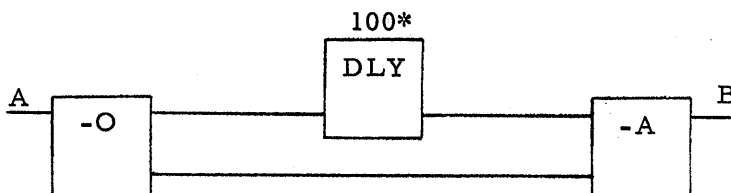


2. The "N" line signal varies approximately _____ volts about a _____ volt reference and usually drives into a (PNP, NPN) type transistor circuit.
3. The "P" line signal varies approximately _____ volts about a _____ volt reference and usually drives into a (PNP, NPN) type transistor circuit.
4. The base of the reference transistor is tied to _____ volts (PNP) or to _____ volts (NPN). The reference transistor collector load establishes the _____ output levels.
5. As the input signal level changes, current switches from the in phase load to out of phase load or vice-versa. Why is the current greater through the input transistor as compared to the reference transistor?
6. The nominal block delay for an alloy junction circuit is _____.
7. The use of remote collector loads is a function of circuit requirements and is indicated and controlled by the _____.
8. What are the main differences between diffused junction type A and type B circuits?
9. When is type B used in preference over type A?
10. The average logic block delay for diffused junction circuits is _____.
11. What function does the 82 μ base resistor and the coil in the collector loads perform? (Diffused Junction Circuits)
12. What is the purpose of an AND circuit extender?
13. If a +AND circuit is a basic N block, what is the designation of the upper level of the out-of-phase output of a -AND circuit? (+N, +P, -N, -P)

14. Is an up-level or a down-level present at the out-of-phase output of a negative binary trigger when it is off?
15. Is the reset to a positive binary trigger negative or positive and to which condition does it reset the trigger?
16. What is the output of Pin D?



17. What is the output of Pin B?



18. With only a negative input pulse of two units duration, draw a circuit which will give a positive output pulse of five units duration whose leading edge coincides with the leading edge of the input pulse (exclusive of logic block delays).

03.00 POWER SUPPLY

03.01 LOCATIONS AND CONTROLS

Objectives: The student without reference, should be able to:

1. List the procedures to apply or remove power to the complete system or individual frames.
2. Define the meaning of each indicator and meter associated with the power system.

The student, with reference, should be able to:

1. Locate any component of the 7090 power system.

Reading Assignment: Read section on Power Controls and Indicators in Power Supply Instruction-Reference Manual.

Review Questions:

1. Where are blower motor CB's located?
2. What relays are located in the SMS supplies?
3. How many 48 volt supplies are there and where are they located?
4. Where is DR1 which controls tape input power?
5. Are MG voltage regulator circuits located in MG box?
6. How is voltage regulation in the system achieved?
7. Where are system power on switches located?
8. How do tape drives receive their power?
9. How is a blower motor failure detected?
10. How many emergency off switches, and where are they found?
11. Why are there 2 power lights on the console?
12. a) What normally happens when PWR switch on a module is turned off?
b) When turned back on?

03.02 STANDARD MODULAR SUPPLY

Objectives: Without references the student should be able to:

1. List the input and output voltages to a SMS power supply.
2. Define the purpose of each transformer in a SMS power supply.
(Use Fig. 63 in the Instruction-Reference Manual).
3. Draw a simplified schematic of a basic power supply.
 - a. Then add to this schematic the circuitry required to make it a M/C power supply.
4. Define the operation of the bias and control winding of a magnetic amplifier.
5. List the conditions that will cause CBI to trip.

With references the student should be able to:

1. Adjust any standard or M/C output voltage of a SMS supply.
2. Adjust a SMS supply if all voltages are low or high.
3. Trace, point by point, the 48 volt path in a SMS frame.

Reading Assignment: Read section on Unit Power Supplies in Power Supply Instruction-Reference Manual. Also section on Magnetic Amplifiers.

Review Questions:

1. The -6 volt supply is 1/4 volt low with TB connection on the O position. Where should the connections be made?
2. Will removal of a mag amp card prevent a module from having proper power?
3. What would be the effect of a broken wire between T3 and RMB6 on an SMS supply?
4. How much of the system loses power -
 - a. When a fuse blows in modular supply A-B of multiplexor?
 - b. When a fuse blows in CPU 1 supply CD?
5. What is the source of the voltage applied to the meter when marginal checking on -12 volts, CPU 2?
6. Ref. Man. Fig. 63: What is the purpose of T5?
7. Ref. Man. Fig. 72: What would be the result of a loss of the input voltage when:
 - a. No fuse is open?
 - b. A fuse blows?
8. Explain the function of R1, R2, R3, and R4 in the modular power supplies. Give an example of the use of at least one set of relay points. Give system page, etc.

03.03 CORE STORAGE POWER SUPPLIES

Objectives: Using only systems we must be able to:

1. List the input output voltages of the 3600 W memory power supply.
2. Explain the operation of the magnetic amplifier as used in the neon detection circuits of memory.
3. List the major differences between the 2KW supply and a SMS supply.
4. Describe the action of the zener diode(s) in the +60 volt, +30 volts and -6V supplies.
5. Explain the operation of the voltage sampling circuits as used in the +30 volt, +60 volt and -6 volt supplies.
6. Describe the principle of operation of a thyatron transistor.
7. Describe the protection circuitry that is employed to protect the core drivers in memory from over-conduction.
8. Adjust individual voltage outputs of the 3600 W supply.
9. Explain how the -18 volt and +85 volts are developed.

Reading Assignment: Read section on Unit Power Supplies in the Power Supply Instruction-Reference Manual

Review Questions:

1. On the CE panel, the 6V-12V meter reads AC from the variac arms and is interpreted as DC voltages. Is this true of the 30V-60V meter also?
2. What would be the result of an open zenner diode in the circuit on systems page 02.01.12.0?
3. Will power be dropped in core storage if the driver check light is turned on?
4. a) On 7302 systems page 02.01.01.0 what are the lines to TB1=5, 7, 9, 11 supplying?
b) Where can these voltages be conveniently measured?
5. What would be the effect of an open diode in the anode lead of the neon detection circuit?
6. What would be the effect of an open zenner diode in the -6 volt supply control circuitry?
7. Systems 02.01.13.0: Output loading varies and output voltage starts to rise. What happens to correct this change?

8. The impedance of a saturable reactor is controlled by varying the _____ in the core.

03.04 POWER SEQUENCING

Objectives: Using only systems, the student must be able to accomplish the following:

1. List the relay sequencing for a normal power on.
2. List the relay sequencing for applying from an emergency off condition.
3. List the relay sequencing for applying power to any individual frames.
4. List the relay sequencing for a normal power off.
5. Describe the principle of operation of the power on variac.
6. Determine all relays, CB's, and contactors associated with any individual frame.

Reading Assignment: Read section on Distribution of Power to System Units in the Power Supply Instruction-Reference Manual.

Review Questions:

1. Emergency Off switch has been pulled out on the console. Can power be brought up now from PCU?
2. Systems 9.02.01.1 shows the HR30 thermal switches. Where do we find these physically?
3. Systems 9.02.02.1 shows no voltage to TB3-6 for variac drive when attempting to bring up power. Where do we locate fuse?
4. How many power sequencing variac cycles will be taken if a DC switch on a module is turned on while the variac is driving down for the first time?
5. CB32 is the _____ control CB.
6. If the CB32 above trips while power is up to the system, the most probable result will be:
 - a. Drop HR30. Why?
 - b. No effect. Why?
 - c. Lose 400 cycle, 208v, 3Ø, to the system. Why?
7. Explain why the relay points for raising the Power On Variac are in parallel while the relay points to lower the Variac are in series.
8. There are _____ power on switches on each frame.

9. Describe the function of CBI SSW in any of the frame interlock circuits.
10. When power is knocked down in Memory which supply drops first, the special voltages or the standard voltages?
11. Which relay is responsible for the two supplies of Memory dropping at different times? How does it accomplish this power off sequence?
12. What is the purpose of DR29, DR30, and DR31?

03.05 MARGINAL CHECKING

Objectives: Without the use of any references, we must be able to:

1. Describe the function(s) of all controls, meters, and indicators associated with marginal checking which are located in the 7151.
2. Define the safe limits of all M/C voltage used in the 7090.
3. List the procedures required to apply a bias to any of the following:
 - a. Core storage +6 volt and -12 volt
 - b. Core storage +60 volt and +30 volt
 - c. Core storage special -6 volt
 - d. Operator's console +6 volt and -12 volt
 - e. SMS frames, any gate, +6 volt and -12 volt

With the use of systems only we must be able to accomplish the following:

1. Trace the selection path of a M/C voltage from the selection push-buttons on the console to the M/C relays in a SMS power supply.
2. Trace the control wires used in biasing the +60 volt and +30 volt memory supply from the toggle switch on the console to the power supplies.

Reading Assignment: Read the marginal check section in the power supply instruction-reference manual.

Review Questions:

1. What is the purpose of the -12V Variac interlock relay points on systems page 9.05.31.1?
2. If DR33 in systems has an open pick coil will it prevent biasing of +6 V M/C voltages?
3. What prevents driving the +60V bias to core too high?
4. What gate selections are made to bias +6V in console?

5. If DR34 N/O FTM, you will be able to marginal check the -12 volts in any frame. True or False? Why?
6. Because of defective power sequence relays channel 2 receives power from cables D and 4 (9.08.01.0) (moved from B & 2 by CE). To bias this channel is it selected as channel 2 or channel 4 from CE console?

03.06 POWER CONVERTER

Objectives: With references we must be able to:

1. Explain the principle of operation of the field flashing and regulating circuits employed with the power converter unit.
2. Adjust the output voltage and regulation circuits of the power converter.
3. Locate all lubrication points on the power converter.
4. Differentiate between a brush type and brushless generator.

Reading Assignment: Read the section on power converter located in the power supply instruction-reference manual.

Review Questions:

1. Describe the use of P1, P2, and P3.
2. How is 400 cycle AC voltage to the system regulated?
3. How often is the generator field flashed?
4. What conditions will cause MG CB-1 to trip?
5. If power is up on the system and fuse 8 (09.02.01.1) opens up, what will happen?
6. What might be the result of an open primary of T3 on a machine running single shift operation? (09.09.01.1)

03.07 MISCELLANEOUS CIRCUITS

This section will cover circuits and components not logically covered in the previous sections of this course. These circuits and components are:

1. Blower Motors and Convenience Outlets
2. PCU 48 volts Special Power Supplies
3. Power and Fuse Indicators
4. Power Cable Distribution

Objectives: With the use of systems only we must be able to accomplish the following:

1. Locate all CB's associated with the convenience outlets and blower motors.
2. Trace the voltage supply path from PCU to any frame for the convenience outlets and blower motors.
3. Locate the 60 cycle and 400 cycle 48 volt power supply.
4. Adjust the output voltage of both 48 volt power supplies.
5. Define the purpose of having a 60 cycle and 400 cycle 48 volt power supply.
6. Trace the turn on path for any power or fuse indicator.
7. Locate all power cable connections within any frame of the 7090 power system.

Reading Assignment: Read the pages applicable to the four topics of this section in the power supply instruction-reference manual.

Review Questions:

1. The 48 volt power supply output is at about 42 volts. What is the logical cause for this trouble?
2. Systems 9.02.04.1 and 9.06.21.1 (2 of 2) both show a power on light. What purpose is served by using two lights?
3. Points of interlock relay DR1 for DC A accomplish what specifically?
4. Answer True or False.
 - a. The blowers in the modules are on as long as HR29 is up.
 - b. The power on lite in CPU 2 lites when the DR1 interlock relay for CPU2 is picked.
 - c. In core storage, CB2 trips when any fuse is blown.
5. What lights are turned on under the following conditions?
 - a. +6V fuse in Data Channel A-B power supply opens.
 - b. Tape Drive. Circuit Protector Open.
 - c. Thermo switch in Multiplexor gate B opens.
 - d. Fuse F2 (09.06.11.1) opens up.
6. On 9.08.03.0 fuse 1 is blown. How will the system be affected?

03.08 TROUBLE-SHOOTING

Objectives: Given the symptoms of a power supply problem we must be able to diagnose these symptoms so that we are able to determine the most logical course.

The following are some trouble-shooting questions to help you develop this diagnostic ability.

1. This question concerns Power Supply A of data channel #1. On system page 09.03.02.1 TB9-2 has an open. Which of the following troubles will occur when power is brought up.
 - a. Power will not come up at all in data channel.
 - b. Power will not come up in Power Supply A.
 - c. Power will come up part way but then will go back down.
 - d. CB1 will trip for Power Supply A.
 - e. CB1 will trip for both Power Supplies in Data Channel.

2. Under normal power ON conditions DC will not come up in any of the modules. Which of the following could cause this.
 - a. CB34 has tripped
 - b. CB35 has tripped
 - c. CB29 has tripped
 - d. CB28 has tripped
 - e. CB 1 has tripped

3. Match the symptoms in Column I with the condition from Column II
 1. Low voltage in all standard supplies in system. a. Blower failure, or faulty thermal switch card.
 2. In CPU 1 CBI in AB supply trips, no open fuses the CB is restored then later CB1 in CD supply trips. b. Defective Mag. Amp. card
 3. One bank of tape drives failing to get power up. c. Motor generator output low
 4. All voltages low, gates A & B, CPU 2 d. DR-1 points in channel dirty
 5. CB1 in channel, gate A-B, fails to trip with open fuse e. Modular supply transformer T4 needs adjustment
 6. Single DC modular voltage high f. Modular supply transformer T3 needs adjustment

4. Assume we have just closed the main Circuit Breaker in the wall box. CB32 (09.02.01.1) is open. What malfunction will occur when we try to bring up power on the system?

5. In taking a normal power on sequence, we find that power comes up on all modules except CPU-1. The variac continues to cycle up and down. Which of the following could cause the trouble?
 - a. HR10-1 (09.02.03.1) shorted
 - b. HR 9-1 (09.02.03.1) fails to make
 - c. HR 9-3 (09.02.02.1) shorted

- d. HR 9-2 (09.02.03.1) fails to make
- e. HR10-2 (09.02.03.1) fails to make

6. While the system is running a program, power drops on Data Channel A. On investigating we find that CB1 for gate A-B supply & CB1 for gate C-D supply are tripped. Which of the following could cause the trouble?

- a. Blower motor on gate A short circuited
- b. -12V fuse on the A-B power supply is open
(Fuse 6 on 09.03.20.1)
- c. HR1-1 N/C fails to make (09.02.02.1)
- d. DR31-1 N/O fails to make (09.02.02.1)
- e. DR16 coil open (09.02.02.1)

04.00 CYCLE & INSTRUCTION TIMING

TABLE OF CONTENTS

04.01	Timing Circuits
04.02	Basic CPU Circuitry Operation
04.03	Cycle Generation
04.04	I Cycle
04.05	Indexing
04.06	Indirect Addressing
04.07	Manual Control Operations
04.08	Diagnostics

04.01 TIMING CIRCUITS

Objectives: Without reference:

1. Be able to describe the frequency and operation of the master oscillator.
2. Be able to trace the operation of the master oscillator through ILD's.
3. Be able to describe the input to the clock, the operation of the clock including all timings, and the outputs generated by the clock.
4. Be able to trace each step of the clock and the outputs generated through ILD's.
5. Be able to describe the operation of the Pulse Generator circuits.
6. Be able to describe how each clock pulse duration is made up and distributed throughout the system, including delays between frames and any compensation for such delays.
7. Be able to describe and trace through ILD's the operation of the CP Set and Hold generation.
8. Using ILD's only, be able to draw a sequence chart of the clock and its outputs and also the CP set and hold generation circuits.

Reading Assignment: Read section 4.1.00 in CE Inst. Ref. Manual
Refer to ILD page 08.00.44 for Master Osc.
Refer to ILD page 08.00.40 for clock

Review Questions - Timing Circuits

1. The master oscillator is a _____ controlled _____ megacycle oscillator located in _____.

2. A machine cycle is _____ usec.
 3. The master oscillator outputs are used to drive the _____ and the _____ generator.
 4. Two outputs from the oscillator which feed the clock are _____ and _____.
 5. The clock is made up of a _____ of _____ triggers.
 6. The clock is started by _____.
 7. Each clock trigger is on for _____.
 8. The _____ of each even clock trigger pulse is gated by _____.
 9. Each clock pulse is _____ in duration.
 10. A machine cycle consists of _____ clock pulses.
 11. Each clock trigger is turned on by _____.
 12. Each clock trigger is turned off by _____.
 13. The reset of the clock comes from ILD page _____.
 14. What 3 methods can turn on the "0" clock trigger?
 15. Describe how the clock is always started at "0" time on PWR ON clear.
 16. The "3" clock trigger comes on at _____ time.
 17. Machine fails to run properly. CE finds that the in phase output of -A (3H) on 08.00.42.1 is shorted to a +N level. Under these conditions what would be the status of each clock trigger (ON or OFF) after each depression and release of the Clear Button? (Explain your Answer)
- NOTE: "Pulse Generator" is on ILD 08.00.48
"Master Clock Pulses" on ILD 08.00.46
18. If the Multiplexor sends an A6(D3) pulse to CPU, what will be the time and duration of the pulse at CPU with respect to Multiplexor time? Why?
 19. Pulses going to CPU compensate for delay. Pulses going to channel _____ (do, do not compensate) for delay.
 20. A Multiplexor A0D1 pulse is _____ in CPU. An A0D1 in MPXR is still labeled _____ in the Data Channel. It must be realized that A0D1 in MPXR may reach D.C. after _____ time however.

21. In addition to logic block delay, there is about _____ delay per foot of cable.

NOTE: CP Set is on ILD page 08.00.47

22. What is the frequency, pulse width and purpose of the "CP Set Pulse"?
23. The CP set pulse width is adjustable (ILD 08.00.47). True or False?
24. On ILD page 08.00.47. What is the variable delay at 4C adjusted for?
25. On 08.00.47.1 it is found that pin A of "C" block (3D) is internally shorted to a +N level all the time. What observation would be made on a scope that was tied to pin D of "DL" (1G). Draw picture observed and indicate frequency and pulse width of waveform.

04.02 BASIC CPU CIRCUITRY OPERATION

Objectives: Without reference be able to:

1. Draw a shift cell and describe its operation in detail.
2. List registers made up of shift cells.
3. Describe in detail the operation of an adder position for any combination of bit inputs.
4. List the grouping of adders for look-ahead.
5. Determine the output of the adder look-ahead for any combination of input bits to the adders.
6. Describe the operation of the program counter and draw a sequence chart of stepping the program counter.
7. Describe the operation of the shift counter and draw a sequence chart of its stepping.

Reading Assignment: Read section 3.0.00, "CPU Internal Functions" (pages 23 to 37) in CE Inst. Ref. Manual.

NOTE: If you haven't read up to page 23 by now, it wouldn't hurt to do it--DO IT ANYWAY!

Review Questions - Basic CPU Circuitry Operation

1. Draw a shift cell from memory.
2. Draw a sequence chart of the operation of shift cell using shift cell you drew only.

3. The basic requirements of the shift cell are _____ and _____.
4. The shift cell is composed of two _____ latches with the _____ phase of the _____ tied to the input of the _____.
5. What is the time relationship between the Set and Hold pulses?
6. List the registers which are made up of shift cells.

NOTE: Adders are on ILD 02.02.1
Section 3.1.12 of C. E. Inst. Ref. Manual.

7. What outputs are available from an adder position?
8. In conjunction with adder look-ahead circuits, the adders are broken into _____ groups with a maximum of _____ positions in any one group.
9. The adders are not a register. They are like a funnel, inputs must be feed in and the output sampled since the adders cannot "hold" info. (True or False)
10. There are _____ stages of Adder LAC.
11. Inputs to 1st stage LAC are _____ and _____ from each _____ position.
12. Inputs to 2nd stage LAC are from _____. They are _____, _____ and _____.
13. Output of 2nd stage LAC returns to _____ inputs to low order position of each adder _____.
14. Why are special circuits used to block carry to adder 8?

NOTE: Program Counter is on ILD 3.05.00.

15. The Program Counter is broken into _____ groups of _____ positions in each group in order to facilitate the _____ circuits.
16. Draw a sequence chart of program counter stepping as a result of 8 step pulses. Show both A and B triggers for positions 17 through 14. Show preset condition with dashed lines.
17. Draw a sequence chart of shift counter for 3 input pulses after a count of 3 has been set in.

18. As a result of the decoded shift instruction, 20_8 is placed in the shift counter. The following malfunction was on the system during this operation: Systems page 03.06.11.1, 01D2H17 (3C), Pin B stays -P.
 - a. What is the output level at pin H of 2D and pin A of 2C?
 - b. What is in the shift counter after the first step?
 - c. What is the total number of shifts?
19. With the first STEP PC 17 pulse after reset, what happens to PC 17?
20. With respect to the rise and fall of the second STEP PC 17, what happens to PC 17?
21. What specific triggers will come on if we set a 6 into the program counter?
22. After a 10_8 is set into the SC, the "SC 16 STEP GATE" on 03.04.17.1 gets shorted to a -N level. What would be the contents of the SC after it has been stepped once?
23. Pin G of the +0 ckt at 3C, page 3.05.07.1, is shorted to +N. The counter is reset, then one STEP PC 17 pulse is generated. What specific triggers of positions 15, 16, and 17 will be on?

04.03 CYCLE GENERATION

Objectives: Without reference be able to:

1. Describe how and when the Master Cycle triggers are turned on and off and their function.
2. Trace the generation of I, E, L and B cycles through ILD's for any given op code.
3. Describe under what conditions a B cycle may be generated.
4. Describe which type of cycles have priority over the others.
5. Describe all the conditions under which End Op can be turned on. When End Op comes on and how it calls for an I cycle.
6. Describe the function of the Multiple Cycle Error Trigger and its effect.
7. Describe what categories of op codes demand E or L cycles.

Reading Assignment: Read section 5.01.00 of CE Inst. Ref. Manual (Fig. 04.1 of this Study Guide will be helpful)

Review Questions - Cycle Generation

1. Name all machine cycles.

2. What is the minimum number of cycles that any instruction can have?
3. What time is the MASTER "E" Time Trigger turned on?
4. What time is the End Op. trigger turned off through normal circuitry?
5. What register in the machine determines the cyclic make-up of an instruction?
6. What would be the cyclic make-up of the following: (FIRST TWO CYCLES)
 - a. +0340
 - b. +0220
 - c. +2000
 - d. +0100
 - e. +0604
7. Which op codes usually produce "Go to E Time" and which "Go to L Time"?
8. What is the basic function of the "B Cycle Interrupt Trigger"?
9. Bringing up the line "Go to I Time" (8.00.12.1) unconditionally means that the next cycle will be an I cycle. (True or False) - Explain.
10. Late time lines (such as I time late) allow _____ pulses to be effective.
11. "Early Time" lines gate _____ pulses.
12. The Master _____ trigger turns on B cycle _____.
13. "B" time together with _____ or _____ turns on B cycle interrupt.
14. What turns on the "Go to I Time" trigger? When is it reset?
15. What time is the "Go to E Time" trigger reset?
16. Name and describe the 3 ways a "B" cycle may occur.
 - a. _____
 - b. _____
 - c. _____

04.04 I CYCLE

Objectives: Without reference be able to:

1. Draw a sequence chart of all I cycle timings.
2. Trace all routings of the Instruction, its address, tag, and dec. during I time through ILD's.

3. Describe what is in any CPU register at any time during I time of a given instruction.

Reading Assignment: Read "I Cycle" in CE Inst. Ref. Manual
Refer to Fig. 5.1-1 in CE Inst. Ref. Manual

Review Questions:

1. What are the three main functions performed during "I" time after the instruction has reached the CPU?
2. At what clock time is the PC gated to the AS?
3. As an instruction is transmitted from core to the CPU, it must pass through what area of the MX?
5. If MF-SB input 5 cannot be conditioned +P, what instruction would be placed in the PR and what would the cycle make-up be? Assume that the PR requires a +P input and that a PAX was arriving during "I" time.
6. What is the purpose for the "DLY" (3B) on 3.06.05.1?
7. On 08.00.22.1, pin B of "C" (3H) is shorted to a -N line. The PC contains 100₈. What location in core would be retrieved during early I time.
8. A SUB is called into the CPU during I time. What instruction will be sitting in the program register if pin E of "+0" (4E) on 03.04.03.1 is shorted to a +P line?
9. During the E cycle of a CLA.....10, the program counter contains 7351. What would be the next location retrieved under program counter control if on 03.05.05.1 the pin F of "+A" (IF) was shorted to a +N line?
10. On 08.00.19.1, a failure in +A (2A) causes Pin G to result in a +P level. The trouble is in the block and not due to any input. What would happen if a SUB instruction was called for?
11. The following is to be executed:

VLM with a count of 15₈ and the multiplicand and multiplier contain some numbers.

What will logically result if on 08.00.23.1, the "C" (3A) is holding pin A at a N level due to a trouble in the block?
12. An LGL with an address of 30₈ is to be executed. If on 08.00.16.1, the +A (5D) is holding the output at -P, what will happen to the LGL?

13. Given: The following program is cycling:

000	CLA	100
001	ADD*	100
002	AXT	5, 2,
003	SUB	10, 2,
004	ADD*	10, 2,
005	TIX	0, 2, 0
006	HTR	4,
007	HTR	2
010	HTR	1
100	Octal	+3

For each of the following items, give the contents of the specified machine circuits.

When Program Counter equals, Time, register or circuit

00000	I8	Adr, Reg, Prog. Reg, Tag Reg.
00001	I10	SR, Prog. Reg.
00001	E4	SR, ADRSWS, ADR Reg.
00001	E11	SR, ADR Reg.
00002	IAE11	SR, ADR Reg.
00003	I10	SR, ADR Sws.

14. Which logic block prevents a one cycle instruction from setting the "end op" trigger at I 10 when the instruction is indirectly addressed?
15. If you have a one cycle instruction with indirect address and on page 08.00.12.1 the block at 5B has pin G shorted to a minus level, will the instruction operate normally, including the indirect address?
16. The machine is reset. The following program is operated.

00000	TRA*	300,,
00300	HPR	400,,
00400	HTR	400,,

The program fails to halt. Which of the following could cause this trouble? More than one may be right.

- 03.05.03.1 +P set PC 8 stays minus
 - 03.06.16.1 +A at 2A pin G stays minus
 - 03.06.16.1 DT at 5B pin BA stays plus
 - 03.06.16.1 +A0 at 2D pin G stays minus
 - 03.06.16.1 +A0 at 2C pin G shorted to -P level
17. 02.01.07.1. The -A at 1D pin H stays minus. Can the machine do a POD 76 type instruction?

18. 02.01.07.1. The +TA at 3E pin G stays minus at all times. Can the machine do a type POD 76 instruction?

19. Program: 00000 CAL 2,,
 00001 ADD 2,,
 00002 HTR 1,,

The -A at 4B pin G, page 02.05.00.1, stays plus. The machine is reset then started. What is the contents of the accumulator and the program counter at the halt?

20. Same problem and questions as in above except the following bug is on the machine. Page 02.12.50.1 C block at 1D has a plus output at all times.

21. Same problem as in above except bug is page 02.05.01.1 C block at 01A3B26 fails to respond to its input (both outputs).

22. What logic block on what page puts the machine into a "Go to E" condition for the E cycle following the I/AE cycle of a flagged instruction which normally goes I, E.

04.05 INDEXING

Objectives: Without reference:

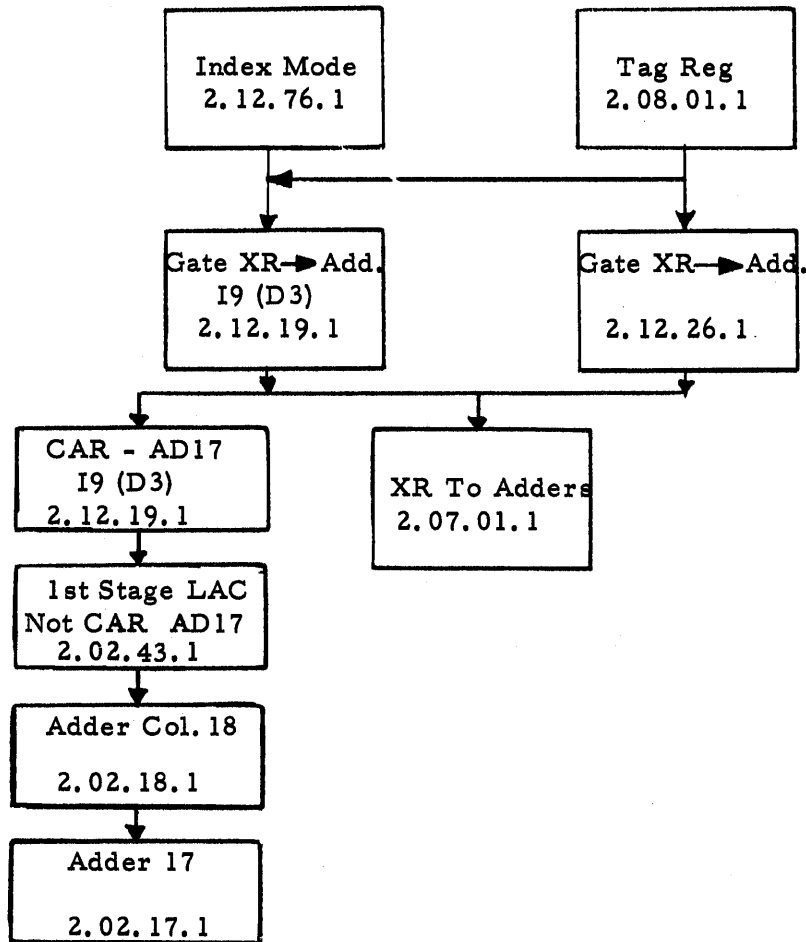
1. Be able to describe the process of indexing done by the machine including:
 - a. What instructions are indexable?
 - b. How and at what time are TAG Regs set?
 - c. What type of arithmetic is used?
 - d. What the result will be for any given address and combination of TAG's.
2. Be able to trace the operation of indexing through ILD's for any given address, TAG combination and XR contents and determine what the adders, XR and AR contain at any time during the process.

Reading Assignment: CE Inst. Ref. Manual Section 3.1.11 describes TAG Regs.
 CE Inst. Ref. Manual Section 7.1.09 is a description of X Regs (Fig. 7.1-6)
 (AT this time you should fully understand the logic of indexing from Basic Programming--you now need to know how it's done. Fig. 04.2 of this study guide should be helpful)

Review Questions - Indexing

1. After the "original" address has been index modified, the new address is called the _____.
2. What does "indexable" or "non-indexable" instruction mean to you?

3. Indexing uses the _____ complement form of arithmetic.
4. Indexing is always a subtractive process. (True or False)
5. Below is a step-by-step logic block diagram of the address modification operation. Using the 7090 Systems Diagrams follow the sequence shown.



6. The following program is in core:

00000	AXT	3,1
1	CLA	7
2	ADD	10,1
3	TIX	2,1,1
4	SUB	7,1
5	HTR	1
6	OCT	10
7	OCT	12
10	OCT	1

The machine has the following bug on it. Page 03.04.03.1, +O at 3B, pin A is shorted to +N. What is the contents of the Accumulator when the machine halts? Where will the machine halt?

7. All three index registers have all ones in them. On 02.12.19.1, Convert Block at 2E, pin B is shorted to -P. CLA 0440 is executed. Will the contents of 0440 be placed in the accumulator?

04.06 INDIRECT ADDRESSING

Objectives:

1. Be able to describe the process of indirect addressing.
2. Be able to trace all functions of indirect address through ILD's.
3. Be able to describe how the IA Trigger is turned on. How it causes an I/AE cycle and how normal cycling is resumed.
4. Be able to determine the resulting address for any combination of flagging and tagging.

Reading Assignment: Section on Indirect Addressing in CE Inst. Ref. Manual

Review Questions - Indirect Addressing

1. The following program is in core:

0000	CLA*	0003,1
0001	SUB*	0010,1
0002	STO*	0005,2
0003	TRA	0000

Index Register A contains 0001
Index Register B contains 0002

- a. The start key is depressed. What will be in the accumulator after the first CLA?
 - b. In relation to the CLA list the following:
 - (1) Direct address
 - (2) Indirect effective address
 - (3) Direct effective address
 - (4) Indirect address
2. What would happen if an indirectly addressed TRA was executed and on 08.00.00.1, the "C" (4B) pin A was held to a +N level due to a trouble within the block?
 3. The IA trigger comes on at _____ of a flagged instruction and goes off at _____.

4. Flagging is recognized by _____.
5. Pre IA trigger comes on at _____ and goes off at _____ of flagged instruction.
6. On 02.10.66.1 what line causes getting an I/AE cycle?

04.07 MANUAL CONTROL OPERATIONS

Objectives:

1. Be able to draw a sequence chart of the manual control single shot circuits on ILD 04.20.04.
2. Be able to describe the operation and timing of the Manual Control Trigger.
3. Be able to describe the function of each of the individual manual operation triggers.
4. Be able to describe the function of the op panel control triggers.
5. Be able to trace the operation of manual controls on the operator's panel and CE panel through ILD's (except the memory nullify which will be covered later).
6. Be able to describe the operation and timings of all the manual control operations using ILD's only.

Reading Assignment: Section 6.0.00 through 6.3.00 in CE Inst. Ref. Manual

Review Questions - Manual Control Operations

1. The Manual Control SS produces a _____ pulse to turn on the _____.
2. The Man. Ctrl. Trig is turned off by _____.
3. The machine is stopped by turning on the Mstr Stp trigger which brings up _____ to prevent any I, E or L cycles. This will allow an I/O operation to complete even though main frame is stopped. (True or False)
4. When the start key is depressed, what insures that we always start with an I cycle after the machine has been cleared?
5. The Program Stop Trigger is turned on by a Halt instruction only. (True or False) (4.20.11.1)
6. What causes End Op of a clear?

7. The Manual Stop trigger comes on at _____ and goes off the next _____ time.
8. The high-low speed mult. step switch selects either a _____ SS for high speed or a _____ SS for low.
9. When the machine is stopped manually, it always stops during an _____ cycle. It goes up to _____ time of that cycle before stopping. When the machine is restarted by hitting the start key it starts at _____ so that it can bring up 1 cycle at the beginning of the cycle.
10. When the machine is in "auto", "Continuous Enter Inst" will cause the instruction in the keys to run at _____ speed. In manual the speed of execution is dependent on _____.

04.08 DIAGNOSTICS

Objectives: Without Reference:

1. Describe the keys and procedure necessary to load and run all diagnostics. Describe 9LD01.
2. Describe the function of the sense switches as used by DEPRX.
3. Describe key settings necessary to select any drive or drives on any channel when using 9T51.
4. Describe the listing which is produced by all diagnostics for both trouble and no-trouble operation.
5. Select and describe which diagnostic to run given a suspected trouble in any area of the machine.

Reading Assignment: Pages 66-70 of CE Handbook
(You should also read the description portion of the listings of:

1. 9M51
2. 9T51
3. 9T55
4. 9M10
5. 9T56
6. DEPRX

NOTE: These are available at each machine location.

Review Questions - Diagnostics

1. What tape drive does diagnostic tape go on?

2. If you wanted to run the 11th and 12th diagnostic on tape, how could you keep the tape from rewinding after loading the 1st one to save time?
3. Describe the function of the 6 sense switches for DEPRX.
4. As a general check of all CPU, what diagnostic should you run 1st?
5. What diagnostic would you run to check data channel data transmission of data to tape?
6. If you suspected prolay trouble on tape B3, what diagnostic would you run? How would you select that drive? How would you get a graph printout?
7. Is it possible for "9S51 Low," which tests the low half of memory, to fail when the fault is in upper memory?
8. You load 9M51 and immediately get a stop at 77766. What does this mean?
9. You load 9M01 from cards and get a stop at 26. What does this mean?
10. What location of core can be displayed to look at the computed checksum for comparison to the punched checksum in a card if 9LD01A was the loader?

05.00 Preliminary Instructions - Study Guide

Table of Contents:

- 05.01 Data Flow
- 05.02 Fixed Point Arithmetic and Word Transmission
- 05.03 Shift Instructions
- 05.04 Transfer Instructions
- 05.05 Control Instructions
- 05.06 Skip Instructions
- 05.07 Indexing Instructions
- 05.08 Trap Mode
- 05.09 I/O Control Instructions
- 05.10 Sense Indicator Instructions
- 05.11 Trouble Shooting Problems

05.01 Data Flow

Objectives:

1. Without reference be able to name the inputs to each of the CPU registers.
2. Without reference be able to describe the data flow paths between all CPU registers.

Reading Assignment:

CE Inst. Ref. Man. Fig. 3.1-3

NOTE: One of the objectives of this course is to help you learn how to use and retrieve information in the CE Inst. Ref. Manual. Therefore, you will be expected to use Appendix A (which is a listing of all the instructions and where the description and flow charts are in the Manual) and Appendix B (which is a listing with systems page numbers for each instruction) to look up information on your own.

Review Questions - Data Flow

1. Name 7 inputs to the Storage Register.
2. Which CPU Registers are made of shift cells?
3. What is the purpose of the inputs to the -OR circuit at Location 5D on Systems 02.02.01.1?
4. List the inputs to Adder Position 30.
5. List the inputs to Adder Position 17.
6. List the inputs to Adder Position 3.
7. List the inputs to Storage Reg. Position Q.

05.02 Fixed Point Arith and Word Transmission

Objectives:

1. With ILD's and the flow chart for any Fixed Pint Instruction (except multiply and divide) be able to describe the state of any line during any given time of an Arith operation in ILD's.
2. Without reference be able to give the rules for addition and subtraction as they pertain to the 7090. (Including signs). (CE Inst. Ref. Manual under description on fixed point arithmetic ops)
3. Without reference be able to describe the operation of fixed point instructions including sign.
4. Without reference be able to describe all word transmission instructions as to their cyclic make up, what data is transmitted, and from and to where.

Reading Assignment

Refer to CE Inst Ref Man. Flow charts for the following Instructions (Appendix A)

CLA	STL	CLM
ADD	STR	XCA
ACL	SLW	
STO	LDQ	
STA	RND	

Note: Some flow charts cover more than one instruction. The instructions above are only one of the group for any given flow chart. You should study all the instructions covered by the flow charts. (ie, The flow chart (Fig 5.3-14) for CLA also covers CAL and CLS)

Review Questions - Fixed Point and Word Transmission

1. During the "E" cycle of an ADD instruction, the Storage Register contains +377777777777_g. The accumulator is +2525_g. The output of the DT at 01B3G03 is shorted to +N. Page 02.02.27.1. What will appear in the accumulator at the completion of the ADD instruction?
2. During "E" time of an ADD instruction, the SR contains bits in cols 30, 31, 32, 33, 34 and 35. The acc contains bits in cols 33 and 35. The output of "And" circuit 01B4G12 on systems page 2.02.46.1 shorts to -6.8v. What answer will appear in the acc after the execution of the instruction?
3. The following program is executed by the 7090:

	CLA	X
	ADD	X+1
	STO	D
	ALS	005
D	HTR	
X	OCT	0001
	OCT	0002

During the execution of the store instruction, the "Set ACC LN5" systems page 2.05.03.1 shorted to a plus N level. What would be the contents of the Accumulator after the machine halted?

4. What is the purpose of the +A circuit at 5I on page 02.09.95.1?
5. At what time during an ADD instruction does the "Q Carry" Trigger get turned on? This is only if the signs are _____ (unlike - alike - doesn't matter) and a Q Carry occurs
6. AC=+, Q=1, P=1, 1-35 377777777777₈. What will the accumulator equal if an ACL is executed and the Data is octal minus zero?
7. What is the purpose of the +AO circuit on systems page 02.12.01.1 at 3H, and the -AO at 2E on page 02.12.38.1?
8.

000	CLA	CONS
001	SUB	CONS+1
002	STO	CONS+2
003	HTR	000
CONS	OCT	7634
	OCT	423
- a. What is the contents of the Accumulator at the Halt?
- b. With +A circuit at 5G, page 02.09.95.1 failing to have a plus output, what will the acc equal?
9.

000	HTR	000
040	CLA	100
041	ADD	100
042	STO	200
043	TRA*	200
100	OCT	40
200	HTR	000

Where should the Program Halt? Where would it Halt if Pin G of -TO at 3G page 03.04.03.1 shorted to -N?

10. 000 CLA 40
 001 ADD 41
 003 SLW 40
 004 CLA 40
 005 HTR 00
 040 OCT +37777777777
 041 OCT +1

- a. What is the contents of the acc at the Halt?
- b. If Pin F of block at 4G on page 02. 09. 00. 1 was shorted to -N, what would the acc equal at the Halt?
- c. Explain your answer.

11. 000 CLA 100
 001 ADD 100
 002 XCA
 003 STQ 200
 004 HTR
 100 OCT 77

What is the contents of the MQ and Location 200 at the Halt with pin II of block 3E page 02. 15. 16. 1 held -N?

12. List the 6 Basic control lines used on an STO Instruction.
13. What is the Basic difference between instructions having POD60 and POD62?
14. What would be the result if the following number came out during "I" time?

+061500100100₈

15. What is the purpose of the out of Phase output on block 1G, page 02. 09. 00. 1?

05. 03 Shift Instructions

Objectives: Without Reference: _____

1. Be able to describe the number of shifts per cycle and the number of cycles necessary for any shift count.
2. Be able to describe from where, and to where each bit of a register is shifted for each of the shift inst.
3. Be able to describe how a shift inst. can "end op"
4. Be able to describe how the "Presense Shift Counter Zero" and "SC=0 trigger" work.

Reading Assignment:

Refer to CE Inst. Ref. Man. for the following instructions:

ALS LRS
LLS RQL
ARS

Review Questions - Shift Instructions

1. 000 AXC 3, 2, 0
 001 CLA 000, 2, 0
 002 ALS 2
 003 HPR 7777

What is the contents of the accumulator at the Halt?

2. What is the functions of the +P Shift Control Line on 02. 11. 79. 1?
3. Which input of 2F page 02. 11. 79. 1 determines the start of the Shift Gate and which determines the end of the Shift Gate?
4. What is the function of the -0 at 4B page 08. 00. 01. 1?
5. Page 03. 04. 19. 1 block 1F pin B is shorted to +N. What will the ACC equal if the following program is operated?

000	CLA	X
001	ARS	20 ₈
002	HTR	
X	OCT	+200000000000

6. Block 5D page 08. 00. 01. 1 has a minus output at all times. What will the Shift Counter equal at I6 of the Halt instruction in the following program?

000	CLA	X
001	LRS	44 ₈
002	HTR	
X	OCT	7776

What is in the Shift Counter at LO of the Halt?

7. How many "L" cycles would it take to complete an ARS of eleven places?
8. What is the maximum number of shifts that may be accomplished in a shifting operation having only one "L" time?
9. Why are the "LRS Control" and "LLS Control" lines out of page 02. 09. 70. 1 not gated by "Shift Gate"?
10. Describe how "L end op" is prevented when the Shift Ctr is stepped from 10₈ to 7₈ during L10.

05.04 Transfer Instructions

- Objectives:** **Without Reference:**
1. Be able to describe the function of the following control lines in ILD's.
 - a. "Minus on any Xfer or STR"
 - b. "Minus on Xfer conditional"
 - c. "Xfer conditional"
 - d. "One cycle transfer Condition Met. "
 - e. "Condition Met"
 2. Be able to describe the modification of the I cycle following a transfer.
 3. Be able to trace through ILD's the operation of all transfers.

Reading Assignment:

Refer to CE Inst. Ref. Man. for the following instructions
(Appendix A)

TQP
TRA
TZE
TLQ

Review Questions: Transfer Instructions

1. What logical function does "+N any trans or store and trap" on page 02.11.55.1 perform?
2. What logical function does "-P minus on trans control" on page 03.06.12.1 perform?
3. Any two cycle transfer will bring up "Xfer conditional" page 03.06.16.1. True - False
4. What is the Routing of SR 21-35 for a two cycle transfer conditions met and not met.
5. What functions does the out of phase output of block 1D page 02.10.08.1 perform?
6. What function does the out of phase output of block IE page 02.10.08.1 perform?
7. Which control line allows the "Transfer to" address to get to the Address Switches on successful one cycle transfers?

8. Which control lines allow the Program Counter into the Address Switches instead of the adders on an unsuccessful one cycle transfer?
9. (Refer to Reference Card of Instruction listing and codes.) There are 22 transfers plus TQO (under optional) and STR which will activate the +0 circuit at 2G page 02. 11. 52. 1. Make a table listing the instructions, their op codes, and which of the AND circuits at 3D, 3F, or 3G will respond.

05.05 Control Instructions

Objectives:

1. Without reference be able to describe the operation of

XEC	HTR
NOP	HPR
SSP	ENK
SSM	CHS
2. Be able to trace the operation of the above instructions through ILD's.

Reading Assignment:

Refer to CE Instruction Ref. Man for instructions named in objectives above.

Review Questions - Control Instructions

1. What instruction codes will bring up the "Halt control" line?
2. Which circuit on page 08.00.01.1 will give the "L end op" on a HPR instruction? On a HTR instruction?
3. (True - False) Depressing "Start" while in manual will not cause the "B cycle Interrupt" tgr to go off.
4. The 7090 is in Manual Status, the following program is in core.

000	CLA	10
001	STO	11
002	HPR	
003	ADD	10
004	HTR	001
005	TRA	000

- a. "Single Step" through the Program. From the operators standpoint what will happen as he executes location 2?
- b. Can he single step to Location 3?
- c. After executing the HTR at Location 4 the operator depresses "Start" and the "Single Step". Describe what will happen.

5. Where will the following programs halt?

Prog.	A		Prog. B		
000	CLA	10	000	CLS	10
001	XEC	11	001	XEC	11
002	HTR	0	002	HTR	0
003	HTR	4	003	HTR	4
004	HTR	5	004	HTR	5
010	OCT	+4	010	OCT	+4
011	TPL	4	011	TPL	4
012	TRA	3	012	TRA	3

6. Execute the programs in Question 5 with the following bugs on the machine. Where will they halt now?
- Block 4G page 02. 11. 51. 1 Stays +N on out of phase output.
 - Block 4E page 03. 06. 05. 1 output stays +N.
 - Block 5F pin B only page 03. 06. 05. 1 output stays -P.
7. Which block on page 02. 12. 92. 1 produces the I6D1 used by a CHS instruction to set the sign?

05. 06 Skip and Sense Instructions

Objectives: Without Reference:

- Be able to describe the operation of:

PBT	ZET	LAS
LBT	NZT	PSE
DCT	CAS	MSE
- Be able to describe the effect of different address of PSE and MSE instructions.
- Be able to describe and trace through ILD's the decoding of Channel, Class, and Address.
- Be able to trace the operation of the instructions listed in "Objective #1" in ILD's with the use of flow charts.

Reading Assignment:

Refer to CE Inst. Ref. Man. for inst. named in Objective #1 above.

Review Questions: Skip and Sense Instructions

1. When does the following program halt?

000	AXT	07635 ₈ , 1,
001	CAL	002,
002	PBT	0001, 1,
003	TRA	5,

NOTE:
(Continued on next page)

004 HTR
 005 LBT
 006 HTR
 007 HTR

2. The following program is in storage.

100 CLA X
 101 CAS Y
 102 NOP
 103 NOP
 104 LAS Z
 105 HPR
 106 HTR
 107 HPR
 X 14
 Y 16
 Z 14

Make a table listing sequentially the times and "AND" circuits on pages 02.11.50.1 and 02.11.51.1 that step the Program Counter.

3. Which Console Connector does the signal from Sense Switch 3 pass through?
4. What conditions are necessary to get "+P Sense PR Gate" on page 2.09.59.1?
5. What conditions are necessary to get "+P PR Adr." on page 3.02.01.1?
6. What conditions are necessary to get "UA02" and what is the origin of the gates decoded as UA02? Give page numbers also.
7. On an PSE 3360, at what times will the POD, SOD, Channel Address, Class and Unit addresses come up?

05.07 Indexing Instructions

Objectives: Without Reference:

1. Be able to describe what instructions (and how they operate) load data into or store data from X Regs.
2. Be able to describe the process of address modification by indexing and trace this operation through ILD's.
3. With reference to the flow chart, for any given indexing instruction be able to determine the line level of any given line associated with that instruction operation in ILD's
4. Be able to describe the operation of index transfer inst. in detail without reference.

Reading Assignment:

Refer to CE Inst. Ref. Man for the following

PAX LXA
PXA TXI
SXA TIX
AXT TSX

(They are representative of flow charts which also include other instructions which you should cover)

Review Questions - Indexing Instructions

1. PAC instruction executed with the following bug on the machine will operate exactly like what other 7090 instruction?

Pin B of 4F on 02. 12. 70. 1

2. True or False? Subtracting the two's complement of a number results in the same thing as adding the true number.
3.

000	CLA	002
001	PDX	001, 2, 7
002	HPR	000

At I 11 of the PDX instruction, what quantity will be felt at adders P to 17?

4. How is Acc. Sign cleared on a PXA or PXD?
5. On the flow chart of the SXA, SXD instructions there is a transfer of AD (3-17) to XR E5 (D1). What is being felt in the adders at this time?
6. True or False? An SXD instruction is indexable.
7. List the control lines from Decoding "TXI" to those that cause AR to PC at L9 (D1) of a TXI.
8. Multiple Choice
What will be loaded into the shift counter when the following program step is executed?

00014 AXT 35, 1,

- a. Nothing is set to the SC on an AXT
- b. 015_8
- c. 035_8
- d. 1_8
- e. 377_8

9. True or False? If block 3E page 02. 12. 19. 1 stayed -N output at all times. it would prevent a TXI from transferring in every case.
 10. XRB = 27₈. What is the contents of XRB after the following instructions are executed if the trouble from Problem 9 above is on the machine?
 11. 030 TXI 44, 2, 5,
 11. At what time is the Adder 3 Carry Trg. turned off on a TIX?
 12. Under what conditions can the Adder 3 Carry Trg. be set by a carry from Adder 5?
 13. On a TSX instruction, at what times is the AR taken to the PC? What control lines perform this function the second time:?
 14. XR's A, B, & C = zero. Execute the following program and indicate the contents of the index regs. at the halt. (All octal numbers)
- | | | |
|-----|-----|----------|
| 000 | TXI | 3, 5, 40 |
| 001 | TXH | 5, 1, 23 |
| 002 | TSX | 3, 4, 7 |
| 003 | TXL | 7, 1, 12 |
| 004 | TNX | 2, 1, 5 |
| 005 | TXL | 1, 1, 20 |
| 006 | TSX | 3, 4, 6 |
| 007 | HTR | 000 |
15.
 - a. Which of the Index transfer instructions do not change the contents of the XRs at any time?
 - b. Which instructions change them only when conditions are met?
 - c. Which instructions change them only when conditions are not met?
 - d. Which instructions are conditional transfers?
 - e. Which instructions are unconditional transfers?

05.08 Trap Mode

Objectives:

1. Without reference be able to describe the logic of Trap Mode and its effect on transfer, both successful and unsuccessful.
2. Without reference be able to describe ETM, LTM and TTR operation.
3. Be able to trace the operation of a trap on any given transfer through ILD's.

Reading Assignment:

Refer to CE Inst. Ref. Man. for description of ETM, LTM, TTR and see Fig 5.3-33

Review Questions - Trap Mode

1. What function does the block at 1H page 02.11.52.1 perform? Why doesn't this affect the operation of a program?
2. What is the cyclic makeup of the following instructions when executed in Trap Mode?
 - a. One cycle transfer condition met
 - b. One cycle transfer condition not met
 - c. Two cycle transfer condition met
 - d. Two cycle transfer condition not met
3. What is the purpose of the "E" cycle on unsuccessful transfers in Trap Mode?
4. What function does the out-of-phase output of block 1G page 02.11.52.1 perform?
5. What logical function does the outputs of block 1H page 02.09.01.1 perform?
6. Why doesn't the op code of a transfer instruction get stored in location zero when in trap mode?

05.09 I/O Control Instructions

Objectives: Without reference:

1. Be able to trace the decoding of I/O instructions through ILD's
2. Be able to describe and trace the operation of the following in ILD.

IOT	TCO
BTT	TRC
ETT	

Reading Assignment:

Refer to CE Instruction Reference Manual for instructions listed in "Objective #2."

Review Questions: I/O Instructions

1. What conditions are necessary to bring up the line "+P Sel. Chan. A"?
Page 06.00.05.1.

2. What conditions are necessary to bring up the line "+P Sel. Chan. B"?
Page 06.00.05.1.
3. Which tailgate connector on Multiplexor will the cable from Block 4A,
page 60.20.04.1, in Channel D be connected to?
4. What function does the "BTT and Chan. Address" line from pin A Block
3C, Page 60.20.05.1, perform?
5. At what time is the "Indicator Sync Trigger" turned on during an I/O
transfer or test instruction?
6. What time duration is "+P Chan. Skip Cntl. CH" plus? Page 60.32.03.1.
7. When the "+P Chan. Skip Cntl. CH" line is plus, does it cause the program
counter to advance? Page 02.11.50.1.
8. A TEF instruction given when the EOF indicator is on (page 60.32.02.1)
will cause the "+P Chan. Cond. Trf. Cntl." line to be plus from _____
to _____ time and this will bring up what control line
in the CPU transfer execution controls?
9. Trace the conditions necessary to bring up the line "+P TRA CD IN
OUT USE" on page 02.10.07.1. Which comes from page 06.01.01.1.

05.10 Sense Indicator Instructions

Objectives:

1. Be able to describe the operation of an individual SI position.
2. Be able to describe the size and purpose of the SI reg.
3. With the use of flow charts be able to trace the operation of each
of all the SI instructions through ILD's.
4. Be able to describe the operation of each of the following Sense Ind.
Ops. without reference.

LDI	TIO
RIS	ONT
SIR	RNT
PAI	STI

Reading Assignment:

Refer to CE Inst. Ref. for above instructions - These are representative of all
24 SI instructions covered by the "Flow Charts".

Review Question - SI Instructions

1. Page 02.06.01.1. Which of the input control lines (Invert - Reset -

Electronic Reset - and Set) are not dependent upon having a one in SR1 position?

2. Under what conditions will "-N SR1 to SI1" be minus? Page 02.06.01.1
3. Assume a one in SR1. What will happen if a positive pulse is gated onto the "+N Set SI1" line? Page 02.06.01.1
4. Assume the lower trigger is latched on. What will happen if the "-P Invert SI" line is held minus (assume as one in SR1) Page 02.06.01.1
5. What happens as the +"-P Invert SI" line is operated to +P in problem #4?
6. What happens if the "-P Invert SI" line is operated minus then plus when there is no "one" in the lower trigger? Page 02.06.01.1
7. Are the Sense Indicators reset when the operator's reset pushbutton is depressed?
8. If Block 5B, page 02.12.64.1, stayed minus N on its output at all times, LDI would act like what other instruction?
9. What is the difference in the way the machine processes the RIR and RIL instructions?
10. The Sense Indicator instructions are considered logical operations because they treat the sign position as another bit and use "P" position in the accumulator. Do any of the SI instructions affect the Acc S position?
11. Execute the following program. Which Sense Indicators are on at the halt?

```
000 CAL      40
001 PAI
002 RIR      765034 (control Field 18-35)
003 IIL      123456
004 PIA
005 LDI      40
006 IIA
007 HTR
008
040 -323456765432
```

12. Execute the following program and give the contents of the Sense Indicators and the accumulator at the halt.

```

005 STL 40
006 CAL 40
007 PAI
010 CAL 41
011 TIF 13
012 IIR 000077
013 ALS 2
014 RNT 000004
015 RIR 7777
016 TIO 12
017 HTR
040 HTR
041 OCT +1

```

05.11 Trouble Shooting Problems

The following are some trouble shooting problems which should prove interesting and help re-inforce your knowledge of preliminary Instructions. They become progressively difficult. If you can do all of them correctly, you probably have gained a more than adequate knowledge of this section of the course.

Preliminary Inst - Trouble Shooting:

1. With the following program in Core storage. Reset and Start are depressed in that order. The computer halts with 0006 in the PC. Note: All addresses are octal.

```

0000 CAL 500
0001 AXT 4, 1
0002 ALS 1
0003 TZE 0005
0004 TIX 0002, 1, 1
0005 HPR
500 0000----1

```

- After halt occurs, the following condition was noted: XRI=2
- a. What should be the result of the above program? Indicate contents of AC, PC, and XRI.
 - b. Without involving an Index Register or an indexing failure, what machine malfunction could create the condition listed above? Explain why and systems page, logic block, and line level.

2. With only the following program in Core Storage, Reset and Start are depressed in that order. Note: All numeric references and addresses are in octal.

	<u>Prog.</u>			<u>Constants</u>
0	CAL	X	(X)	+377777777777
1	ADM	Y		
2	SLW	Z	(Y)	-100000000001
3	ALS	45		
4	LDQ	Z	(Z)	+000000000000
5	LGL	3		
6	TZE	012		
7	LBT			
10	HTR	Error		
11	HTR	OK		
12	HTR	Error		

AC contents upon completion: +00.....04 PC contained 00010

- What should the result of the above program be? Indicate contents of AC, MQ, and PC.
 - What machine malfunction could result in the above listed stop and condition? State your reasons why and list systems page, logic block, and line level.
3. With only the following program in Core Storage. Reset and Start are depressed in that order. All references are octal.

	<u>Prog.</u>			<u>Constants</u>
000	CLA	A	A	OCT -12
001	ADD	B	B	OCT + 4
002	SUB	C	C	OCT -10
003	LDQ	D	D	OCT + 2
004	TLQ	007		
005	HTR			
006	HTR	000		
007	HTR			

The following conditions were noted upon completion:

AC +2—— MQ +2 PC=7

The program was machine cycled through and the following was noted: TLQ "L" Time Complete AC +2, MQ+2, SR+2

- What should be the result of the above program? Indicate contents of AC, MQ and PC
- What machine malfunction could cause the above conditions? State your reasons why and list systems page, logic block and line level

4. With only the following program in Core Storage, Reset and Start are depressed in that order.

	<u>Prog.</u>			<u>Constants</u>	
000	CLA	A	A	OCT	-12
001	ADD	B	B	OCT	+ 4
002	SUB	C	C	OCT	- 3
003	LAS	D	D	OCT	- 5
004	HTR				
005	HTR				
006	HTR				

The following conditions were noted upon completion on ACC -3 PC 005

- Describe the above program and what should be the result.
- What machine malfunction could cause the above conditions? State your reason why and list systems page, logic block and line level.

5. The following routine is used in our 9M03 Indexing Diagnostic. 9M01 and 9M02 have been run successfully.

	CLA	E			
	TNZ	D	E	OCT	+00000
	AXT	7, 1, 0	A	OCT	+00001
B	TIX	B, 1, 1			
	PXA	0, 1, 0			
	CAS	A			
C	HTR	Error			
	TRA	OK			
D	HTR	Error			

- What is this diagnostic routine checking?
- Computer halts at C. What machine failure would create this result? List reasons why, systems page, logic block, line level.

STUDENT STUDY GUIDE

06.00 7302 Oil Core Storage

Objectives:

This study guide is designed to familiarize the students with the IBM 7302 Oil Core Storage. It includes both the general theory of magnetic storage and the way this theory is applied to the actual Core Storage used on the 7090 System.

Note:

This course is a prerequisite for 7302 Air Memory. The "7302A Self Study Guide" will be used in your study of Air Memory.

Contents:

- 06.01 General Concept of Core Storage
- 06.02 Ferrite Core Theory
- 06.03 Coincident Current Addressing
- 06.04 Basic Core Cycle and Addressing
- 06.05 Sense Amps, Registers and Timings
- 06.06 Special Circuits
- 06.07 CE Panel and Temperature Control
- 06.08 Diagnostic - 9S51

06.01 General Concept of Core Storage

Objectives: Without Reference;

1. Be able to describe the need for Core Storage
2. Be able to show the relationship of Core Storage to the rest of the 7090 system.
3. Be able to name and describe the function of all lines between the system and the 7302.
4. Be able to describe the general data flow to and from the 7302.

Reading Assignment:

Read up to page 14 of 7302 CE Inst. Ref. Manual
Refer to Fig. 06.01 of this study guide.

Review Questions - General Concept of Core Storage

1. How many words can be stored in the 7302? How many cores are needed to store these words?

2. What are the 2 basic functions of core storage?
3. What is meant by access time?
4. Access time on the 7302 equals _____.
5. A memory cycle equals _____ usec.
6. What does random access mean?
7. Define the following:
 - a. Read out
 - b. Store
 - c. MDBI
 - d. MDBO
 - e. ORS
 - f. MAR
 - g. MDR
 - h. Memory select
8. Memory select pulse is timed by an A_____ pulse.

06.02 Ferrite Core Theory

Objectives: Without Reference;

1. Be able to describe the physical and magnetic properties of the ferrite Core used in the 7302.
2. Be able to show how binary information can be stored in a core.
3. Be able to describe how this information can be read out of the core after it has been stored.

Reading Assignment:

Up to page 14 in 7302 Inst. Ref. Manual

Review Questions - Ferrite Core Theory

1. The type of core used in the 7302 is an IBM type _____.
2. What are the advantages of using a core-type of storage?
3. What is meant by "destructive read-out"?
4. How would you explain to another, the theory of operation of a core?

5. Core X contains a bit. How does this bit get transferred to the MDR?

06.03 Coincident Current Addressings

Objectives:

1. Be able to describe the need for Coincident Current Addressing.
2. Be able to show the physical arrangement of the cores and windings in a plane.
3. Be able to show the means of selecting a specific core out of a plane.
4. Be able to show the physical arrangement of planes in core array.

Reading Assignment:

Up to page 14 of Inst. Ref. Man. See Figs. 06.02-06.06 this Study Guide. See Figs. 90, 91 in Inst. Ref. Man.

Review Questions : Coincident Current Addressing

1. How many X and Y lines are used in a 7302 core plane?
2. a. How many planes are used in the core array?
b. How are they numbered?
3. Define:
 - a. Half select
 - b. Flip
 - c. Noise------(please, not a loud sound)
 - d. Full select
 - e. Addressing
4. There are _____ sense windings per plane. Why?

06.04 Basic Core Cycle and Addressing

Objectives:

1. Be able to show how X and Y drive current is developed for read and write.
2. Be able to show which out puts of MAR are used for the different address decoders and be able to determine the drive lines selected for any given address.
3. Be able to describe the development of inhibit current and its function.

Reading Assignment:

Read up to page 32 in CE Inst. Ref. Man. See Figs. 06.02, 06.03, 06.04, 06.05, 06.06 of this Study Guide.

Review Questions:

1. What timing pulses are sent to memory from multiplexor besides the memory select pulse?
2. What is the function of the R/W trigger?
3. Why is the memory cycle arranged in a write-read sequence?
4. Full select current equals _____ ma.
5. Define:
 - a. Matrix switch
 - b. G switch
 - c. G switch segment
6. How many primary windings does a G switch have?
7. How many:
 - a. X G switches
 - b. Y G switch segments
 - c. Y G switches
 - d. X G switch segments
 - e. X drivers
 - f. Y drivers
 - g. Z drivers
8. The core addressing scheme uses what positions of MAR to control:
 - a. X segment selection
 - b. Y address decoding
 - c. Y segment selection
 - d. X address decoding
 - e. Z driver selection
9. There are _____ Z windings per plane. These windings run parallel to the _____ windings.
10. How is the expression $A \nabla B$ read and what does it mean?
11. Sixteen pairs of complement lines are developed by the X and Y decoders. Why?

06.05 Sense Amps, Registers and Timings.

Objectives:

1. Be able to describe the functions of the various Core Registers.
2. Be able to draw a sequence chart of the gates and pulses developed during a core cycle.
3. Be able to describe the operation of the Strobe Generation.
4. Be able to describe the sources of noise generated in core and how it is overcome.

Reading Assignment:

Refer to ILD 0.01.00.0 for logic block diagram of Memory. Read up to page 43 in CE Inst. Ref. Manual. Refer to timing chart on ILD 00.01.01.0.

Review Questions - Sense Amps, Registers and Timing.

1. There are _____ sense amplifiers, _____ MAR positions and _____ MDR positions.
2. What is the purpose of a strobe pulse?
3. What is meant by the term "strobe generator"?
4. What MAR positions can be examined to determine the sense winding segment used for a specific address?
5. There are _____ different timed strobe pulses. Each pulse is approx _____ usec. in duration. The strobe generator is triggered by a _____
6. What are the 2 basic outputs of the MDR?
7. Draw a sequence chart of the memory cycle and show the X, Y and Z drive line relationships for read and write time.
8. What means are employed in the 7302 to reduce unwanted noise?
9. On what operation is the DIG and DOG up simultaneously?

06.06 Special Circuits

Objectives:

1. Be able to describe the operation of the component circuits common to Oil Memory only. These circuits are MAR, MDR Sense Amp. and Core Drivers.

Reading Assignment: "Special Circuits" Page 48 of CE Inst. Ref. Man.

Review Questions: Special Circuits

1. What are the 4 component circuits common to the 7302 Oil Memory only?
2. What are the coincident input polarities required to set a MAR position to "1"? To "0"?
3. The mem select line to MAR is an N line. What is its voltage level when at a:
 - a. -N
 - b. +N
4. At what time is MAR reset?
5. At what time is the MDR reset?
6. What inputs must be at what polarity to enter data into the MDR from the system?
7. Why is the sense amplifier output gated with a +DIG to set a MDR position?
8. What is the timing of the DOG?
9. At what time is MDBO 35 available to the system?
10. Draw a logic block diagram of a MDR position showing input and output lines.
11. Why is a differential amplifier used to terminate a sense winding?
12. The logic block diagram of a sense amplifier shows that 4 windings drive 4 differential amplifiers. These amplifiers are terminated by 2 emitter followers which drive a strobe gated amplifier whose output drives a level setter. With the above in mind answer the following:
 - a. What is the purpose of the emitter follower?
 - b. What function(s) does the strobe gated amplifier perform?
 - c. What does the level setter do?
13. What is the load of the following core drivers?
 - a. Z driver
 - b. X driver
 - c. Y driver

14. What is the purpose of the negative feedback circuit from the collector of the power TX to the input of the complementary emitter follower?
15. What input(s) must be at what polarity to cause a Y core driver to conduct?

06.07 CE Test Panel and Temp Control

Objectives:

1. Be able to describe the operation of the timing error circuitry in Core Storage
2. Be able to operate and describe the operation of the CE Test panel circuitry.
3. Be able to describe the flow and control of oil through the temperative control unit and the function of the sensing devices.
4. Be able to describe the troubles indicated by the indicator lights on the CE Test Panel.

Reading Assignment:

Pages 43 to 47; Figs. 44, 45, 46, 47 of CE Inst. Ref. Man.

Review Questions - CE Test Panel - Temp. Control

1. The 7090 uses the "timing error" line to recognize that a timing error existed. (T of F)
2. If the Z driver timing trigger is turned on falsely due to noise, what happens?
3. What do the following CE Test Panel switches accomplish:
 - a. Write Ones
 - b. Check Reset
 - c. Check Ones - Zeros
 - d. Write Zeros
 - e. Test On-Off
4. What condition(s) are met to turn on the following indicator:
 - a. Z driver check
 - b. Cooler power fault
 - c. Power on
 - d. Check stop
 - e. Address check
 - f. Memory test
 - g. Memory power fault

5. The mixing value is adjusted so that the oil entering the tank is _____ degrees F, plus _____ degrees F or minus _____ degrees F.
6. The over temperature light turns on when the oil entering the tank is _____ degrees F or greater. The under temperature light turns on when the oil entering the tank is _____ degrees F or less.
7. What switch on the heat exchanger controls the fan?

06.08 Diagnostic - 9S51

The following four problems are 9S51 printouts of four different troubles on the 7302 Oil Memory. Look over each one carefully and see if you can find the most probable cause for each printout. Consult your instructor for the answers to these problems.

Following the four 9S51 problems is a sample printout of 9S54B. Note the differences between it and 9S51. If you find anything that you can't decipher, consult your instructor.

PROBLEM 1 - PART A

DATA COMPILE PRINT FOR TEST ROUTINE 01. ONES DISCRIMINATION.
TOTAL ERROR COUNT THIS PASS - 4096.

04 TO 07	Y ADDRESSING SEGMENT MAR 8-10								11 TO 14	X ADDRESSING SEGMENT MAR 15-17							
	0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
0	32	32	32	32	32	32	32	32	G 0	128	128	128	128	128	128	128	128
1	32	32	32	32	32	32	32	32	1	128	128	128	128	128	128	128	128
2	32	32	32	32	32	32	32	32	S 2	128	128	128	128	128	128	128	128
3	32	32	32	32	32	32	32	32	W 3	128	128	128	128	128	128	128	128
4	32	32	32	32	32	32	32	32	I 4	0	0	0	0	0	0	0	0
5	32	32	32	32	32	32	32	32	T 5	0	0	0	0	0	0	0	0
6	32	32	32	32	32	32	32	32	C 6	0	0	0	0	0	0	0	0
7	32	32	32	32	32	32	32	32	H 7	0	0	0	0	0	0	0	0
8	32	32	32	32	32	32	32	32	8	0	0	0	0	0	0	0	0
9	32	32	32	32	32	32	32	32	O 9	0	0	0	0	0	0	0	0
10	32	32	32	32	32	32	32	32	U10	0	0	0	0	0	0	0	0
11	32	32	32	32	32	32	32	32	T11	0	0	0	0	0	0	0	0
12	32	32	32	32	32	32	32	32	P12	0	0	0	0	0	0	0	0
13	32	32	32	32	32	32	32	32	U13	0	0	0	0	0	0	0	0
14	32	32	32	32	32	32	32	32	T14	0	0	0	0	0	0	0	0
15	32	32	32	32	32	32	32	32	15	0	0	0	0	0	0	0	0

DRIVER ERROR COUNT ASSUMING THAT ERROR OCCURRED ON READ OUT OF MEMORY.

0	Y SEGMENT								DRI VER	X SEGMENT							
	0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	-01	0	0	0	0	0	0	0	0
512	512	512	512	512	512	512	512	512	+01	512	512	512	512	512	512	512	512
256	256	256	256	256	256	256	256	256	-02	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+02	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-03	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+03	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-04	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+04	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-05	0	0	0	0	0	0	0	0
256	256	256	256	256	256	256	256	256	+05	512	512	512	512	512	512	512	512
256	256	256	256	256	256	256	256	256	-06	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+06	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-07	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+07	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-08	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+08	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-09	0	0	0	0	0	0	0	0
256	256	256	256	256	256	256	256	256	+09	512	512	512	512	512	512	512	512
256	256	256	256	256	256	256	256	256	-10	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+10	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-11	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+11	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-12	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+12	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-13	0	0	0	0	0	0	0	0
256	256	256	256	256	256	256	256	256	-13	512	512	512	512	512	512	512	512
256	256	256	256	256	256	256	256	256	-14	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+14	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-15	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+15	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	-16	256	256	256	256	256	256	256	256
256	256	256	256	256	256	256	256	256	+16	256	256	256	256	256	256	256	256

PROBLEM 1 - PART B

NUMB	DIGIT PLANE ERRORS			DIGIT PLANE SEGMENTS					
	PICK	DROP	NUMB	PICK	DROP	SD	SC	SB	SA
4R71	0	0	3M53	0	0	ZD 1024	0	0	0
4R70	0	0	3M52	0	0	ZC 1024	0	0	0
4R69	0	4096	3M51	0	0	ZB 1024	0	0	0
4Q68	0	0	3L50	0	0	ZA 1024	0	0	0
4Q67	0	0	3L49	0	0				
4Q66	0	0	3L48	0	0				
4Q65	0	0	3K47	0	0				
4Q64	0	0	3K46	0	0				
4P63	0	0	3K45	0	0				
4P62	0	0	3K44	0	0				
4P61	0	0	3K43	0	0				
4P60	0	0	2K42	0	0				
4P59	0	0	2K41	0	0				
4P58	0	0	2K40	0	0				
3P57	0	0	2J39	0	0				
3P56	0	0	2J38	0	0				
3N55	0	0	2J37	0	0				
3N54	0	0	2J36	0	0				

MAKE SURE THAT YOU UNDERSTAND THE ABOVE PRINTOUT COMPLETELY

WHAT MACHINE MALFUNCTION COULD CAUSE THE ABOVE DIAGNOSTIC RESULTS

TRY TO BE AS SPECIFIC AS POSSIBLE

PROBLEM 2 - PART A

DATA COMPILE PRINT FOR TEST ROUTINE 02 ZEROS DISCRIMINATION
 TOTAL ERROR COUNT THIS PASS- 16384

04 Y ADDRESSING									11 X ADDRESSING								
TO SEGMENT MAR 8-10									TO SEGMENT MAR 15-17								
07	0	1	2	3	4	5	6	7	14	0	1	2	3	4	5	6	7
0	128	128	128	128	128	128	128	128	G	0	128	128	128	128	128	128	128
1	128	128	128	128	128	128	128	128		1	128	128	128	128	128	128	128
2	128	128	128	128	128	128	128	128	S	2	128	128	128	128	128	128	128
3	128	128	128	128	128	128	128	128	W	3	128	128	128	128	128	128	128
4	128	128	128	128	128	128	128	128	I	4	128	128	128	128	128	128	128
5	128	128	128	128	128	128	128	128	T	5	128	128	128	128	128	128	128
6	128	128	128	128	128	128	128	128	C	6	128	128	128	128	128	128	128
7	128	128	128	128	128	128	128	128	H	7	128	128	128	128	128	128	128
8	128	128	128	128	128	128	128	128		8	128	128	128	128	128	128	128
9	128	128	128	128	128	128	128	128	O	9	128	128	128	128	128	128	128
10	128	128	128	128	128	128	128	128	U	10	128	128	128	128	128	128	128
11	128	128	128	128	128	128	128	128	T	11	128	128	128	128	128	128	128
12	128	128	128	128	128	128	128	128	P	12	128	128	128	128	128	128	128
13	128	128	128	128	128	128	128	128	U	13	128	128	128	128	128	128	128
14	128	128	128	128	128	128	128	128	T	14	128	128	128	128	128	128	128
15	128	128	128	128	128	128	128	128		15	128	128	128	128	128	128	128

DRIVER ERROR COUNT ASSUMING THAT ERROR OCCURRED ON READ OUT OF MEMORY

Y SEGMENT									DRI	X SEGMENT																																																				
0	1	2	3	4	5	6	7	VER	0	1	2	3	4	5	6	7																																														
0	0	0	0	0	0	0	0	-01	0	0	0	0	0	0	0	0																																														
204820482048204820482048204820482048	+01	204820482048204820482048204820482048	-02	10241024102410241024102410241024	+02	10241024102410241024102410241024	-03	10241024102410241024102410241024	+03	10241024102410241024102410241024	-04	10241024102410241024102410241024	+04	10241024102410241024102410241024	-05	10241024102410241024102410241024	+05	10241024102410241024102410241024	-06	10241024102410241024102410241024	+06	10241024102410241024102410241024	-07	10241024102410241024102410241024	+07	10241024102410241024102410241024	-08	10241024102410241024102410241024	+08	10241024102410241024102410241024	-09	10241024102410241024102410241024	+09	10241024102410241024102410241024	-10	10241024102410241024102410241024	+10	10241024102410241024102410241024	-11	10241024102410241024102410241024	+11	10241024102410241024102410241024	-12	10241024102410241024102410241024	+12	10241024102410241024102410241024	-13	10241024102410241024102410241024	+13	10241024102410241024102410241024	-14	10241024102410241024102410241024	+14	10241024102410241024102410241024	-15	10241024102410241024102410241024	+15	10241024102410241024102410241024	-16	10241024102410241024102410241024	+16	10241024102410241024102410241024

PROBLEM 2 - PART B

DIGIT PLANE ERRORS						DIGIT PLANE SEGMENTS				
NUMB	PICK	DROP	NUMB	PICK	DROP	SD	SC	SB	SA	
4R71	0	0	3M53	0	0	ZD	1024	1024	1024	1024
4R70	0	0	3M52	0	0	ZC	1024	1024	1024	1024
4R69	0	0	3M51	0	0	ZB	1024	1024	1024	1024
4Q68	0	0	3L50	0	0	ZA	1024	1024	1024	1024
4Q67	0	0	3L49	0	0					
4Q66	0	0	3L48	0	0					
4Q65	0	0	3K47	0	0					
4Q64	0	0	3K46	0	0					
4P63	0	0	3K45	0	0					
4P62	16384	0	3K44	0	0					
4P61	0	0	3K43	0	0					
4P60	0	0	2K42	0	0					
4P59	0	0	2K41	0	0					
4P58	0	0	2K40	0	0					
3P57	0	0	2J39	0	0					
3P56	0	0	2J38	0	0					
3N55	0	0	2J37	0	0					
3N54	0	0	2J36	0	0					

MAKE SURE THAT YOU UNDERSTAND THE ABOVE PRINTOUT COMPLETELY

WHAT MACHINE MALFUNCTION COULD CAUSE THE ABOVE DIAGNOSTIC RESULTS

TRY TO BE AS SPECIFIC AS POSSIBLE

PROBLEM 3 - PART A

DATA COMPILE PRINT FOR TEST ROUTINE 02. ZEROS DISCRIMINATION.
TOTAL ERROR COUNT THIS PASS - 4096.

Y ADDRESSING										X ADDRESSING								
04	SEGMENT MAR 8-10									11	SEGMENT MAR 15-17							
TO	0	1	2	3	4	5	6	7	TO	0	1	2	3	4	5	6	7	
0	0	0	0	0	0	0	0	0	G	0	32	32	32	32	32	32	32	
1	0	0	0	0	0	0	0	0		1	32	32	32	32	32	32	32	
2	0	0	0	0	0	0	0	0	S	2	32	32	32	32	32	32	32	
3	0	0	0	0	0	0	0	0	W	3	32	32	32	32	32	32	32	
4	128	128	128	128	128	128	128	128	I	4	32	32	32	32	32	32	32	
5	128	128	128	128	128	128	128	128	T	5	32	32	32	32	32	32	32	
6	128	128	128	128	128	128	128	128	C	6	32	32	32	32	32	32	32	
7	128	128	128	128	128	128	128	128	H	7	32	32	32	32	32	32	32	
8	0	0	0	0	0	0	0	0		8	32	32	32	32	32	32	32	
9	0	0	0	0	0	0	0	0	O	9	32	32	32	32	32	32	32	
10	0	0	0	0	0	0	0	0	U	10	32	32	32	32	32	32	32	
11	0	0	0	0	0	0	0	0	U	11	32	32	32	32	32	32	32	
12	0	0	0	0	0	0	0	0	P	12	32	32	32	32	32	32	32	
13	0	0	0	0	0	0	0	0	U	13	32	32	32	32	32	32	32	
14	0	0	0	0	0	0	0	0	T	14	32	32	32	32	32	32	32	
15	0	0	0	0	0	0	0	0		15	32	32	32	32	32	32	32	

DRIVER ERROR COUNT ASSUMING THAT ERROR OCCURRED ON READ OUT OF MEMORY.

Y SEGMENT									DRIVER	X SEGMENT							
0	1	2	3	4	5	6	7		0	1	2	3	4	5	6	7	
0	0	0	0	0	0	0	0	-01	0	0	0	0	0	0	0	0	
512	512	512	512	512	512	512	512	+01	512	512	512	512	512	512	512	512	
256	256	256	256	256	256	256	256	-02	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+02	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-03	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+03	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-04	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+04	256	256	256	256	256	256	256	256	
512	512	512	512	512	512	512	512	-05	256	256	256	256	256	256	256	256	
0	0	0	0	0	0	0	0	+05	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-06	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+06	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-07	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+07	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-08	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+08	256	256	256	256	256	256	256	256	
0	0	0	0	0	0	0	0	-09	256	256	256	256	256	256	256	256	
512	512	512	512	512	512	512	512	+09	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-10	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+10	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-11	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+11	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-12	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+12	256	256	256	256	256	256	256	256	
512	512	512	512	512	512	512	512	-13	256	256	256	256	256	256	256	256	
0	0	0	0	0	0	0	0	+13	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-14	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+14	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-15	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+15	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	-16	256	256	256	256	256	256	256	256	
256	256	256	256	256	256	256	256	+16	256	256	256	256	256	256	256	256	

PROBLEM 3 - PART B

NUMB	DIGIT PLANE ERRORS			DIGIT PLANE SEGMENTS						
	PICK	DROP	NUMB	PICK	DROP	SD	SC	SB	SA	
4R71	0	0	3M53	0	0	ZD	0	0	0	0
4R70	0	0	3M52	0	0	ZC	1024	1024	1024	1024
4R69	0	0	6M51	0	0	ZB	0	0	0	0
4Q68	0	0	3L50	0	0	ZA	0	0	0	0
4Q67	0	0	3L49	0	0					
4Q66	0	0	3L48	0	0					
4Q65	0	0	3K47	0	0					
4Q64	0	0	3K46	0	0					
4P63	0	0	3K45	0	0					
4P62	4096	0	3K44	0	0					
4P61	0	0	3K43	0	0					
4P60	0	0	2K42	0	0					
4P59	0	0	2K41	0	0					
4P58	0	0	2K40	0	0					
3P57	0	0	2J39	0	0					
3P56	0	0	2J38	0	0					
3N55	0	0	2J37	0	0					
3N54	0	0	2J36	0	0					

MAKE SURE THAT YOU UNDERSTAND THE ABOVE PRINTOUT COMPLETELY

WHAT MACHINE MALFUNCTION COULD CAUSE THE ABOVE DIAGNOSTIC RESULTS

TRY TO BE AS SPECIFIC AS POSSIBLE

PROBLEM 4 - PART A

DATA COMPILE PRINT FOR TEST ROUTINE 12, MAX 2 NOISE ONES CHECKERBOARD.
 TOTAL ERROR COUNT THIS PASS - 4.

04	Y ADDRESSING								11	X ADDRESSING							
TO	SEGMENT MAR 8-10									SEGMENT MAR 15-17							
07	0	1	2	3	4	5	6	7	14	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0	OG 0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0 1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	OS 2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	OW 3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	OI 4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	OT 5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	OC 6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	OH 7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0 8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	OO 9	0	0	0	0	0	0	1	0
10	0	0	0	0	0	0	0	0	OU 10	0	0	0	0	1	0	0	0
11	0	0	0	0	0	0	0	0	OT 11	0	0	1	0	0	0	1	0
12	0	0	0	0	0	0	0	2	OP 12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	OU 13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	OT 14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0 15	0	0	0	0	0	0	0	0

DRIVER ERROR COUNT ASSUMING THAT ERROR OCCURRED ON READ OUT OF MEMORY.

Y SEGMENT								DRIVER	X SEGMENT							
0	1	2	3	4	5	6	7	VER	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0 -01	0	0	0	0	0	0	0	0
0	0	0	0	0	0	4	0	0 +01	0	0	1	0	1	0	2	0
0	0	0	0	0	0	0	0	0 -02	0	0	1	0	0	0	2	0
0	0	0	0	0	0	4	0	0 +02	0	0	0	0	1	0	0	0
0	0	0	0	0	0	2	0	0 -03	0	0	1	0	1	0	1	0
0	0	0	0	0	0	2	0	0 +03	0	0	0	0	0	0	1	0
0	0	0	0	0	0	2	0	0 -04	0	0	0	0	1	0	1	0
0	0	0	0	0	0	2	0	0 +04	0	0	1	0	0	0	1	0
0	0	0	0	0	0	2	0	0 -05	0	0	0	0	0	0	0	0
0	0	0	0	0	0	2	0	0 +05	0	0	1	0	1	0	2	0
0	0	0	0	0	0	2	0	0 -06	0	0	1	0	0	0	2	0
0	0	0	0	0	0	2	0	0 +06	0	0	0	0	1	0	0	0
0	0	0	0	0	0	4	0	0 -07	0	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0	0 +07	0	0	0	0	0	0	1	0
0	0	0	0	0	0	4	0	0 -08	0	0	0	0	1	0	1	0
0	0	0	0	0	0	0	0	0 +08	0	0	1	0	0	0	1	0
0	0	0	0	0	0	3	0	0 -09	0	0	1	0	1	0	2	0
0	0	0	0	0	0	1	0	0 +09	0	0	0	0	0	0	0	0
0	0	0	0	0	0	3	0	0 -10	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0 +10	0	0	1	0	0	0	2	0
0	0	0	0	0	0	3	0	0 -11	0	0	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0 +11	0	0	1	0	1	0	1	0
0	0	0	0	0	0	3	0	0 -12	0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	0	0 +12	0	0	0	0	1	0	1	0
0	0	0	0	0	0	1	0	0 -13	0	0	1	0	1	0	2	0
0	0	0	0	0	0	3	0	0 +13	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0 -14	0	0	0	0	1	0	0	0
0	0	0	0	0	0	3	0	0 +14	0	0	1	0	0	0	2	0
0	0	0	0	0	0	1	0	0 -15	0	0	0	0	0	0	1	0
0	0	0	0	0	0	3	0	0 +15	0	0	1	0	1	0	1	0
0	0	0	0	0	0	1	0	0 -16	0	0	1	0	0	0	1	0
0	0	0	0	0	0	3	0	0 +16	0	0	0	0	1	0	1	0

PROBLEM 4 - PART B

NUMB	DIGIT PLANE ERRORS			NUMB	PICK	DROP	DIGIT PLANE SEGMENTS			
	PICK	DROP	SD				SC	SB	SA	
0A00	0	0	1E18	0	0	ZD	0	0	1	0
0A01	0	0	1E19	0	0	ZC	0	0	0	0
0A02	0	0	1E20	0	1	ZB	0	0	1	0
0B03	0	0	1F21	0	0	ZA	0	0	2	0
0B04	0	0	1F22	0	0					
0B05	0	0	1F23	0	0					
0B06	0	0	1G24	0	0					
0B07	0	0	1G25	0	0					
0C08	0	0	1G26	0	0					
0C09	0	3	1G27	0	0					
0C10	0	0	1G28	0	0					
0C11	0	0	2G29	0	0					
0C12	0	0	2G30	0	0					
0C13	0	0	2G31	0	0					
1C14	0	0	2H32	0	0					
1C15	0	0	2H33	0	0					
1P16	0	0	2H34	0	0					
1D17	0	0	2H35	0	0					

MAKE SURE THAT YOU UNDERSTAND THE ABOVE PRINTOUT COMPLETELY

WHAT MACHINE MALFUNCTION COULD CAUSE THE ABOVE DIAGNOSTIC RESULTS

TRY TO BE AS SPECIFIC AS POSSIBLE

NOW PERFORMING 9S54B-H.
TEST 16. STROBE TIMING
END OF STROBE TEST

SAMPLE OF 9S54

9S54B-H PASS COMPLETE.
NOW PERFORMING 9S54B-L.
TEST 16. STROBE TIMING
END OF STROBE TEST

9S54B-L PASS COMPLETE

TEST	OCTAL	MEMORY	ADR	REG	BIT	09, ZEROS	CHECKERBOARD	COMPLEMENTED.
EXIT	ADDR	000 000	1111 1111			0000000000	1111111111	2222222222
		567 489	0123 4567			0123456789	0123456789	0123456789
72427	37126	111 110	0101 0110			1111111111	1111111111	1111111111
9S54B-L	PASS	COMPLETE						

***** 9S54 DIAGNOSTIC PRINTOUTS *****

1. Normal identification and pass complete print outs are shown on the top of this page.
2. Next is a "Single error heading" and one error printout.
3. On the attached page is a "Compile Printout" * * Note the difference between this and a "9S51" printout.

SAMPLE OF 9S54

DATA COMPILE PRINT FOR TEST ROUTINE 09, ZEROS CHECKERBOARD COMPLEMENTED.
 TOTAL ERROR COUNT THIS PASS - 1.

DIGIT PLANE ERRORS						DIGIT PLANE SEGMENTS			
NUMB	PICK	DROP	NUMB	PICK	DROP	SA	SB	SC	SD
00	0	0	18	0	0	ZA	0	0	0
01	0	0	19	0	0	ZB	0	0	1
02	0	0	20	0	0				
03	0	0	21	0	0				
04	0	0	22	0	0				
05	0	0	23	0	0				
06	0	0	24	0	0				
07	0	0	25	0	1				
08	0	0	26	0	0				
09	0	0	27	0	0				
10	0	0	28	0	0				
11	0	0	29	0	0				
12	0	0	30	0	0				
13	0	0	31	0	0				
14	0	0	32	0	0				
15	0	0	33	0	0				
16	0	0	34	0	0				
17	0	0	35	0	0				

ERROR COUNT LISTED BY G SWITCH MATRIX OUTPUTS

Y ADDRESSING									X ADDRESSING								
TO	SEGMENT MAR 5-7								TO	SEGMENT MAR 4,8,9							
13	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	17	X0	X1	X2	X3	X4	X5	X6	X7
0	0	0	0	0	0	0	0	0	G	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	S	2	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	W	3	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	I	4	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	T	5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	C	6	0	0	0	0	0	1	0
7	0	0	0	0	0	0	0	0	H	7	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	S	9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	E	10	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	C	11	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0

ERROR COUNT LISTED BY G SWITCH DRIVERS INVOLVED.

ERRORS ARE ASSUMED TO HAVE OCCURRED ON READ OUT OF MEMORY.

Y -DRIVER-X	Y -DRIVER-X	Y -DRIVER-X	Y -DRIVER-X
0 -01	0	1 -05	1
1 +01	1	0 +05	0
1 -02	0	0 -06	1
0 +02	1	1 +06	0
0 -03	1	1 -07	0
1 +03	0	0 +07	1
1 -04	1	0 -08	0
0 +04	0	1 +08	1
0 -09	0	1 -13	1
1 +09	1	0 +13	0
1 -10	0	0 -14	1
0 +10	1	1 +14	0
0 -11	1	1 -15	0
1 +11	0	0 +15	1
1 -12	1	0 -16	0
0 +12	0	1 +16	1

06.10 7302A (Air Memory)

Air Memory is a self-study course using the "7302A Self-Study Guide" Form # R23-2904. It contains "objectives", reading assignments and questions to help you learn the differences between Oil and Air Memory.

After completion of the Self-Study Guide your instructor will give you a quiz to see if you can meet the objective satisfactorily.

After completion of Air Memory you can resume use of this study guide with the next section.

07.00 ADVANCED INSTRUCTIONS

(SELF STUDY)

Description:

This package is made up of five general sections, which are quite different from one another in content. The sections are:

- 07.01 Fixed Point MPY/DIV
- 07.02 Introduction to Exponential Math and 7090
Floating Point Numbers
- 07.03 Floating Point Instructions
- 07.04 Logical AND/OR Instructions
- 07.05 Convert Instructions

The entire breakdown is furnished in the index of this package for easy reference by the student to any sub-section.

The first sections on Fixed Point MPY/DIV are prefaced by an open book quiz. If the student feels, after looking at the quiz, that he understands how the instructions work; he need not read the text pertaining to that area. However, if he does use the text for MPY/DIV; he should take them in the sequence they are given (i. e. General Logic, Flow Diagram Text, Summary).

The remaining parts of the package are not prefaced with a quiz. They do have, after each instruction type, review questions and answers. These should be used as self-critiquing guide posts for the student.

It is further suggested that the student follow the entire package through in sequence, rather than skipping around. Whenever the student feels he has gained a knowledge level with which he is satisfied, he should ask to take the final quiz (administered by his instructor).

Materials:

Additional material the student must have available for use with this package is:

1. CE Instruction-Reference Manual #223-6895
2. Volumes 1-7 of 7090 Systems Diagrams

CONTENTS

<u>Section</u>	<u>Description</u>	<u>Page</u>
<u>07.01</u>	<u>Fixed Point Multiply and Divide</u>	
	07.01.1 Multiply: Quiz and Answers General Logic Flow Diagram Text Variable Length Multiply Summary	
	07.01.2 Divide: Quiz and Answers General Logic Flow Diagram Text VDH, VDP Summary	
<u>07.02</u>	<u>Introduction to Exponential Math and FP Numbers</u>	
	07.02.1 Exponential Math	
	07.02.2 FP Numbers	
<u>07.03</u>	<u>Floating Point Instructions</u>	
	07.03.1 Floating Point Add/Sub Over-all Logic Flow Diagram Text Summary Review Questions and Answers	
	07.03.2 Floating Multiply Over-all Logic Flow Diagram Text Summary Review Questions and Answers	
	07.03.3 Floating Divide Over-all Logic Flow Diagram Logic Summary Review Questions and Answers	
	07.03.4 FP Underflow and Overflow Logic Summary	
<u>07.04</u>	<u>Logical AND/OR Instructions</u>	
	07.04.1 Over-all Logic Flow Diagram Text Summary Review Questions and Answers	
<u>07.05</u>	<u>Convert Instructions</u>	
	07.05.1 Over-all Logic Summary Review Questions and Answers	

07.01.1 MULTIPLY

OPEN BOOK QUIZ

FIXED POINT MULTIPLY

1. The shift ctr. is loaded with _____ octal at _____ time of the MPY.
2. The _____ is cleared at E6 (D1).
3. The sign of the _____ and _____ is set _____ if the SR and MQ signs are both minus.
4. The SR will be loaded with the _____ at E7 (D1).
5. With a _____ of zero we would _____ in E time.
6. This condition is checked for as we bring the _____ into the SR.
7. The MPY instr. proceeds to L time if the _____ (is) (is not) equal to zero.
8. The SR and ACC are gated to the _____ all during _____ time.
9. In order to take address to ACC, the _____ cycle tgr must be _____.
10. This tgr is turned on by MQ position _____ output being equal to _____ as we shift.
11. If the _____ tgr does not turn on, we can shift a maximum of _____ times per cycle.
12. The conditions necessary to turn on the presense tgr are _____ and a _____ S.C. pulse.
13. The pulse which turns on the presense tgr is actually CP set and this occurs during the _____ part of any clock pulse.
14. If the presense 35 tgr turns on at L4 time, the MPY _____ cycle tgr will turn on at _____.
15. The output of the _____ tgr has a _____ * delay which causes the timing relationship in question #13.

16. The presense 35 tgr does two main things when it turns on: It (stops) (allows) ACC and MQ shifting and (prevents) (allows) the _____ cycle tgr to turn _____.
17. It is possible to shift the 1st LO on a MPY instr. (true or false)
18. If _____ or more consecutive zeros are in the multiplier, we can shift the _____ and _____ regs right every clock pulse.
19. The shift ctr tells us when we have looked at all positions of the _____ and allows MPY to _____ at the next L10.
20. Give the number of L cycles that each of the following MPY instructions would require.

(a)	LDQ	Z	NOTE:	Common data field, all
	MPY	X		numbers OCTAL
(b)	LDQ	Y	<u>Location</u>	<u>Contents</u>
	MPY	XY		
			X	+377777777777
(c)	LDQ	XY	XY	+000000000000
	MPY	X	Y	-273525672535
			YZ	-000002002002
(d)	LDQ	X	Z	-000000003511
	MPY	YZ		
(e)	LDQ	YZ		
	MPY	Z		
(f)	LDQ	Z		
	MPY	Y		

FIXED POINT MULTIPLY

ANSWERS

1. 43, E3(D1)
2. ACC
3. ACC, MQ, plus
4. Multiplicand
5. Multiplicand, End Op
6. Multiplicand
7. SC
8. ADD, L
9. MPY ADD, ON
10. 34, one
11. MPY ADD, 12
12. MQ34 = 0, Step
13. Latter
14. ADD, L8
15. Pre-sense, 52
16. Stops, allows, MPY ADD, ON
17. True
18. 2, ACC, MQ
19. Multiplier, End Op
20.
 - (a) 6
 - (b) none
 - (c) 4
 - (d) 12
 - (e) 5
 - (f) 6

GENERAL LOGIC

FIXED POINT MULTIPLY

Let us first discuss the logic of fixed point multiply on the 7090. For best results use the data flow chart of the 7090 registers with this write-up. Figure 3.3-1 Manual of Instruction 223-6895.

If you were to do a longhand multiplication of 2 binary numbers the following 3 points would be noted. For example, multiply on a piece of scratch paper 5 x 5 in binary.

1. Each time a bit is found in the multiplier, you write down the multiplicand below the line. We will refer to these numbers as partial products.
2. For each position that you look at in the multiplier, as you scan from right to left, the partial product below the line is shifted one place to the left. This shows the relationship of the partial product to the bit which caused its existence in the multiplier.
3. These partial products are totaled after looking at all multiplier bits.

Let us now look at the 7090 MPY operation. The 7090 uses the MQ as a multiplier which must be loaded prior to the MPY instruction. The SR will become the multiplicand and is fetched during the E cycle of the MPY Instr. With this in mind we will now look at the three points above as they apply to the 7090. Use the data flow chart of the 7090 as reference.

Point #1

In binary multiplication a bit in the multiplier means add in the multiplicand one time, since the highest digit the multiplier can contain is one. Therefore, this is what we do. Each time a bit is found in the MQ we add the SR once, keeping a running total, to develop the product.

Point #2

These partial products must be added in relation to the bit being represented in the multiplier. Since we cannot shift each partial product left in relation to the partial sum (running total), we shift the partial sum to the right. The result is the same. Try this on your longhand method for proof.

Point #3

Instead of adding all partial products upon completion of the MPY, we will keep a running total as we progress. In other words, each partial product will be added to our partial sum each time a bit is found in our multiplier.

One other point: Since the TS of the partial products is kept in the ACC and must be shifted right, and the multiplier is in the MQ and must be scanned from right to left; both functions are properly handled by sampling the multiplier bit in MQ35 and shifting both registers right simultaneously. Therefore, at the completion of the MPY our 70 bit product is in the ACC and MQ and our multiplier is lost.

Let us now summarize the logic of the fixed point MPY on the 7090. The two numbers we are going to multiply are located in the SR and the MQ. The MQ is our multiplier; the SR is our multiplicand. We are going to look at our multiplier one bit at a time, and the position we actually use is MQ35.

Each time we find a bit in the multiplier we add the multiplicand to the ACC. Both of these, our partial sum in the ACC and our multiplier in the MQ, are shifted right. This is done as we scan our multiplier from right to left and adjust our partial sum so we can add in the multiplicand the next time. After we have looked at all bit positions in the multiplier, which makes a total of 35 shifts necessary, our product will be in the ACC. The least significant 35 bits in the MQ and the most significant in the ACC.

FLOW DIAGRAM TEXT

If you feel that you now understand the general logic of a fixed point MPY, let us refer to the flow chart in Figure 5.3-17. I will now take you through the flow chart step by step finding out the exact time and routing that these things occur. Also, emphasis will be placed on why.

We will start at the E cycle of the MPY and discuss the five main events which take place during this cycle. These five events do not vary from one MPY to another. First, 43 octal will be set into the shift counter at E3D1. 43 octal is equal to 35 decimal and is the number of shifts necessary for us to interrogate each position in our multiplier.

Secondly, at E6D1 we will clear the ACC register; the reason being that this is where our partial product will be developed and we will add the storage register to the ACC each time that we find the bit in our multiplier. The first time we find a bit in our multiplier we naturally want the ACC to be zero.

To the right of this we see that the sign of our product is also set during the E cycle. The rule for setting the sign of the product will of course be the same as for any multiply operation. Unlike signs, our product will be minus; like signs, our product will be plus. You will note both the sign of the MQ and ACC are set, since two 35 bit numbers would result in a 70 bit product. For this we need both registers to hold our answer.

To the left of where we set the sign you will see a block which decides if $MQ\ 35=1$. This test, although started during the latter part of this E cycle, will not be important to us until we get to the top of the next section of our flow chart. I will discuss this point further at that time.

Let us now look at the block that says SB to SR at E7D1. Just below this block there is a decision being made which decides if SB 1-35 is = to 0. This test is being made as we bring the multiplicand into the storage register. The purpose of this test is to see if our multiplicand is zero, meaning our answer would be zero, and we should not waste time by taking the L cycles necessary to look at each position of multiplier. Prove this point by looking at the "yes" exit from this decision block. You will see that we clear the MQ and end OP. This results in a zero answer, which is correct.

Assume that our multiplicand is not zero; in fact, let us assume a multiplicand of all bits. Also, we will assume a multiplier of all bits for our first pass through the right side of the flow chart.

On this side of the flow chart you will note that there are two paths which we can follow. First, let us take the left path since there are no conditions involved.

You can see that we take SR to ADD and ACC to ADD all during L time. With both of these inputs to the adders all during L time; when we find that we do have a bit in our multiplier, all that is necessary is taking the output of the adders back to the ACC. THIS RESULTS IN OUR ADDING IN THE MULTIPLICAND ONE TIME. You will note, however, that taking ADD back to the ACC is conditional (from the 2 arrows which come in at the top of the block). Therefore, we must return to the top of the diagram where we will start with the decision block MQ35=1.

This decision block (MQ35=1) is the same block that appeared during the latter part of the E cycle. The result of this test is not used until we get to L time. We are assuming a multiplier and a multiplicand of all bits. Thusly, MQ35 would be equal to one under the conditions we are taking the first time through. Coming out of the yes side of the decision block we turn on what is called the MPY ADD cycle trigger. There are 3 possible times to turn this tgr on; but for our conditions it will turn on at L0D3. Turning it "on" gives us the conditions necessary to take ADD back to the ACC. This results in our having added our multiplicand to the ACC one time; but, we did it as a result of finding a BIT in our multiplier--this bit being MQ35.

After taking the ADD back to ACC, we then shift the MQ and the ACC right. We now have right adjusted the partial sum and also shifted the next multiplier bit into position 35 of the MQ. Next on the flow chart it shows stepping the SC and testing the SC to see if it is zero. Of course, at this time it is not. So we go up to another decision block and this one checks MQ34=1. This is a little misleading so I will explain in some detail at this point.

MQ34=1 is actually being tested at the same time we shift right and step the SC. The logic of this block is actually this: If MQ34 is equal to one at the time we step the shift counter, then we are going to turn on a PRESENSE 35 TRIGGER. This presense 35 trigger is actually looking at the bit going into MQ35 as it is shifted. We do not look at the output of MQ35 again, as we did during the last part of the E cycle. The logic is the same, however, and the presensing you will find is necessary if we should be fast shifting. This will be discussed in more detail later.

In our example, with the multiplier of all bits, the presense trigger should be turned on. This is going to do two major things. First, it will stop further shifting. Second, it will allow the MPY ADD cycle trigger to turn on the very next L0, L4, or L8. In our case it will be the L4D3. This is saying again that we have another multiplier bit and that we should add the SR into our partial sum in the ACC. This is done by the fact that we take ADD back to the ACC once more; this time at L6D1. After doing this we again shift the ACC right 1 place, which adjusts our partial sum and positions our multiplier so we can look at the next bit going INTO MQ35. Stepping of the SC will take place; and since the shift counter is not = 0, we again turn on our PREsense tgr because MQ34 does shift a bit into MQ35. This will again allow the MPY ADD cycle tgr to turn on the next L8D3.

This sequence will continue throughout the MPY instruction. Since we have a multiplier of all bits the presense tgr would be turned "on" during each shift. This allows a maximum of three add cycles during each L cycle. The MPY ADD cycle tgr can be turned on only at L0, L4, and L8. Therefore, with a maximum 35 add cycles we would take a total of 12 L cycles before the shift ctr would become zero in time to allow end operation at L10. This is for a multiplier of all bits.

Now let's look at exactly the opposite case, where the multiplier is all zeros. You will note that we do not test for this condition since the multiplier has to be loaded into the MQ prior to the MPY instruction. We do check the multiplicand for all zeros during E time, but not the multiplier. However, let's see if a multiplier of all zeros cannot be handled somewhat faster than a multiplier of all ones.

Starting again at the top right side of our flow diagram everything would be the same in the left path; where we take SR to ADD and ACC to ADD all of L time, but the decision for MQ35=1 would be negative as we start the first L cycle. Therefore, we would not turn on the MPY add cycle tgr; but we would go down to the right where it says "shift MQ and ACC right the 1st L zero". Next we would step the shift ctr to 42 octal. The shift ctr is not equal to zero, so we would go up and look at MQ34 as we shifted. Since we could never have a bit in MQ34 with a multiplier of all zeros, we could never turn on the presense 35 trigger. If the presense 35 trigger does not turn on, we will go to the block where we shift MQ and ACC right every clock pulse. We will shift right, step the SC and continually test the output of MQ34. We never find a bit so we continue shifting every clock pulse or a max. of 12 times a cycle. We step the shift counter 12 times a cycle. Therefore, it will take a maximum of 4 L cycles to complete the multiply. Our ACC having previously been cleared, we now have zeros for our product; both in the ACC and MQ.

Let us assume that we have a multiplier with alternate ones and zeros; not necessarily every other one a bit, but there are ones and zeros throughout the multiplier. If this would be the case, and MQ35 was = 1 the first time, we would proceed as with the first example. If MQ35 was = 0 the first time, we would proceed as we did in the second example. However, after the first bit had been sensed in MQ35, the path we would take next would depend on presensing the input to 35 as we shifted (or the output of MQ34). As long as we do not encounter a bit while we are shifting, we will shift every clock pulse. As soon as we do encounter a bit in shifting, we turn on the presense tgr. This, you will recall, stopped further shifting immediately. It also allows the MPY add cycle tgr to turn on the next L0, L4, or L8. This tgr will give us time for the adders to settle down before we take the output of the adders back to the accumulator. Then we could start shifting again. Of course, by this time the MPY add cycle tgr and the presense tgr would both be turned off and we would start all over again.

As I mentioned earlier, the presense tgr has a definite purpose; as opposed to our just looking at the output of MQ35. Assume that we had several zeros consecutively in our multiplier. We would be shifting very rapidly (every clock pulse). The presense tgr is looking at the input to 35. When it senses a bit going into 35, it must prevent further shifting with the partial sum in the proper position in the ACC so we may add the SR to it. Therefore, we must sense 35 before the bit actually gets there if we are to stop shifting in time. This is what makes the presense trigger necessary if fast shifting is to be allowed.

There is one other unique feature concerning the presense tgr. This is located on page 02.09.55.1 in your systems diagrams. You will note that the output of this trigger has a 52* delay. The purpose of this delay is to prevent the MPY add cycle tgr from turning on with the same clock pulse that turns on the presense

trigger. For example, assume that we shifted right at L4 time. As we shifted right at L4 time, we recognized a bit going into MQ35. This, then, would turn on the presense tgr. Now, at L4 time the step pulse is a result of a CP set. The CP set comes at the latter part of the L4 clock pulse. Therefore, we would turn on the presense tgr during the latter part of L4 time. We do not want to turn on the MPY ADD cycle tgr until L8, as it would not be reliable with the remaining sliver of the L4 clock pulse. To insure that this does not happen, the presense tgr is delayed 52* insuring us that the MPY ADD cycle tgr WILL NOT turn on with the same clock pulse under which the PRESENSE TRIGGER turns on. Because of this particular circuit design, in order for us to take advantage of the fast shifting, we must have at least 2 consecutive zeros in our multiplier. Let me discuss that one point briefly.

Assume first a multiplier of 5_8 . MQ35 would be equal to one and we could turn on the MPY ADD cycle trigger at L0D3. This would allow adders to ACC at L2D1 and our first right shift at L3D1. At this time, when we shift right and step the shift counter, we would not see a bit going from 34 to 35--which means we will shift again at L4D1. This time we would see a bit going from 34 to 35 at L4 time. This bit will cause the presense tgr to turn on with the CP set falling under the L4 clock pulse, or late at L4 time. But we cannot turn on the MPY ADD CYCLE TGR until L8 because the output of the presense tgr is delayed 52* and by then the L4 clock pulse has passed. Therefore, we still must wait to take the add cycle--even though we shifted on 2 consecutive clock pulses.

Now let's take another case where we have a multiplier of 11_8 . MQ35 would still be equal to 1 and we would still follow the same procedure as before at the 1st L0 time. Turning on the MPY add cycle tgr at L0, take add to ACC at L2 and shift right at L3. We would not see a bit as we shifted at L3, so we shift again at L4. We would not see a bit this time either so we would shift again at L5. At L5 time we would sense a bit and turn on the presense tgr with the CP set under an L5 pulse. This will allow the MPY ADD cycle tgr to come on at L8 also. Now we have actually saved some time. We are adding the SR into the ACC as a result of our fourth multiplier bit and it is still the first L cycle.

VARIABLE LENGTH MULTIPLY and MULTIPLY ROUND

Let us now look at VLM and MPR. First, let's take the differences between VLM and MPY. Refer to the flow chart on the following page - Figure 5.3-19.

As you can see, the major difference is during E time. They take SR 18-35 to the ADD and then ADD 3-17 to the AS and AS 12-17 to the SC. In other words, they are putting into the SC the count found in 12-17 of the VLM instruction. The number of positions we will look at in the multiplier is determined by this count field.

There are, however, a couple of differences which actually do not show on this chart. Let's say, for example, that VLM was indirectly addressed, which would be the case if the count in the count field of this instruction contained bits in 12 and 13. Since this instruction is flagable, bits in 12 and 13 on the count field will force an E I/A cycle to be taken. During the E I/A cycle we would read out the contents of the location which we intended to use as the multiplier. However, the decrement portion of our intended multiplier will now wind up in the shift counter during the following normal E cycle. The address of this location will actually be used to go and get our multiplier. Therefore, you must be very cautious when using variable length multiply that the count field does not cause indirect addressing. A count this large would not be logically correct anyway unless the instr. was being used for a delay.

There is one other thing on VLM which is not shown on the flow chart. This is the clearing of the ACC which we do at E6D1 of our normal MPY. Clearing the ACC on MPY is dependent upon the shift counter not being equal to zero. You never have this possibility on the regular fixed length MPY. However, on VLM if 12-17 of the decrement is blank, you will have a shift count equal to zero. The ACC would not be cleared. The MQ will not be altered, and the original contents would remain when you proceed to the next instruction. It would be worth your time to look in systems and prove these two points about VLM.

Now look at MPR in figure 5.3-18 directly above. MPR is identical to our regular MPY until it ends operation and proceeds into the next I cycle. At this time MQ position 1 is tested for a bit or no bit condition. If MQ 1 is equal to one, then we're going to add one to the part of the product which is in the ACC. The reason for this is that the bit in MQ1 represents, in respect to the ACC, one half or larger value in the MQ. The high order 35 bits of our product in the ACC has been "rounded". Other than this, MPY and MPR are identical.

SUMMARY

Let us now summarize in general terms the MPY operation. During the E cycle of the MPY the multiplicand is brought from the SB into the SR. Prior to this we have set the SC to 43 octal, cleared the ACC in preparation for keeping a running total of partial products and set the sign of our product. Also, we have made ready for the test of MQ35.

We go into the L time of the MPY providing our multiplicand is not = to 0. If it is zero, then we merely clear the MQ and go to the next instruction.

In L time of the MPY instruction the actual multiplication is performed. We test MQ35 at the beginning of the first L cycle to see if a bit was present, in which case we must add the multiplicand into the ACC once. If there is no bit in MQ35 at this time, we shift the MQ and the ACC right and step the shift ctr. From this point on a bit in our multiplier will be sensed by the presense 35 trigger. This is done by looking at MQ34 as we shift right. If the presense 35 tgr is turned on, it means we did sense a bit; therefore, we should stop shifting and take time for an add cycle. By add cycle I mean add the multiplicand into the ACC one time. These two triggers will then be turned off and we can resume shifting right and scanning the multiplier for further bits. When the SC has reached zero, we know that we have looked at all of our multiplier bits and taken all of the necessary add cycles. We can now end OP and proceed to the next instruction.

We can see that the number of L cycles taken on the MPY is directly related to the number of zeros which are in our multiplier. We have the ability to shift every clock pulse as long as consecutive zeros are sensed. Therefore, MPY is a variable cycle instruction and may take from 4 to 12 L cycles or it may end operation in E time, if the multiplicand is zero; thereby taking only 2 cycles.

07.01.2 DIVIDE

OPEN BOOK QUIZ

1. On the DVH instruction the SR contains the _____ and the AC and MQ contain the _____.
2. If the SR is greater than the AC on DVH or DVP during the E cycle test, it means our quotient will be no more than 35 bits. (True or False)
3. The sign of the MQ is set at _____. Why is the sign of the ACC not set at this time also?
4. Explain the logic of the test for a Q carry during the E cycle.
5. Acc positions _____ and MQ positions _____ are gated left on the left shift operation.
6. MQ position 1 is shifted to ACC position 35 in complement form. Explain why.
7. During L time the Q carry means the CSR is larger than the CACC. (True or False)
8. A "no Q carry" condition during L time will cause what 2 things to take place?
9. Explain what the "adders to ACC at L2, L6 or L10" block is accomplishing when it takes place.
10. When the shift counter equals zero, we attempt one more reduction. (T or F)
11. Why is complementing the ACC necessary during I time of the next instruction?
12. Triggers T1 and the programmable divide check tgr. always turn on under the same conditions. (True or False)
13. T1 is useful only on what instruction(s)?
14. When end op occurs due to a divide check, how many cycles will the instruction take?
15. A VDP instruction with a count of 16_8 will take how many L cycles?
16. DVH will stop at I5 if the divide check occurs. What will be the contents of the following registers when the machine stops? (Give a good description since the actual numeric value will depend on a given program)
 - a. Storage Register
 - b. Accumulator
 - c. MQ
 - d. Program Register
 - e. Program Counter
 - f. Shift Counter

FIXED POINT DIVIDE

ANSWERS

1. Divisor, Dividend
2. True
3. (E11) It will be the remainder and should have the same sign as original dividend.
4. The test for a Q carry determines if the SR $>$ than the ACC portion of our dividend. This means a quotient size which will fit in the MQ
5. P-35, 2-35
6. The ACC portion of the dividend is kept in comp form so the SR can be subtracted from it whenever necessary.
7. True
8. Place bit in MQ35 and gate ADD to ACC
9. It is subtracting the divisor (SR) from the dividend (ACC portion)
10. True
11. It was maintained in comp form throughout and must be "re-complemented" to represent our remainder
12. True
13. DVH, VDH
14. Three
15. 5
16. SR - original divisor
ACC - original dividend (hi order 35 pos)
MQ - original dividend (lo order 35 pos)
P. R - divide instruction
P. C - location of the VDP + 1

DIVIDE
GENERAL LOGIC

DVH

Dividing two binary numbers can be done by the same longhand method as we do decimal numbers. For example, let's divide 25 into 50 using the decimal longhand method.

$$\begin{array}{r} 2. \\ 25 \overline{) 50} \\ \underline{50} \end{array}$$

The answer is 2 as you can readily see. Now let's convert the numbers to their binary equivalent and use the same method of division.

$$\begin{array}{l} 25_{10} = 31_8 = 11,001 \\ 50_{10} = 62_8 = 110,010 \end{array} \qquad \begin{array}{r} 10. = 2 \\ 11,001 \overline{) 110,010.} \\ \underline{110\ 01} \\ 000\ 000 \end{array}$$

Thus, we can see that we are not learning a new concept of math--only a different method by which the machine accomplishes the same chain of events. Before further discussion of divide, we must be familiar with the following terms: divisor, dividend, and quotient. These can best be illustrated in the following manner.

$$\text{divisor} / \frac{\text{quotient} + \text{remainder}}{\text{dividend}}$$

The 7090 will divide in this fashion with the register usage as shown.

$$\text{SR} / \frac{\text{MQ} + \text{ACC}}{\text{ACC} + \text{MQ}}$$

As you can see from this illustration, we will divide 35 bits into a 70 bit dividend. Our answer will be a 35 bit quotient, and a 35 bit remainder. Keep in mind that a remainder is only what is left over (remaining) from the original dividend--it is not an extension of the quotient.

There is one limitation on DVH due to the size of the registers. The CSR must be larger than the CACC when we start the division. The reason for this is to insure that our quotient will not be larger than 35 bits. Let me explain this assuming a 3 position register with the illustration below.

<u>At Start</u>		
SR = 3	011/101,101	(Using 3 pos. registers)
ACC = 5	11	Results in <u>4</u> bit answer if FIRST attempt at reducing dividend <u>is</u> successful.
MQ = 5	0101	
	11	
<u>At Finish</u>	100	
The MQ should = 17 ₈	11	
but this is impossible	0011	
in a 3 pos. register	011	

The MQ should be 17_8 ; but how can we hold 17_8 in the MQ if it has only 3 positions? We obviously cannot, which is the reason the CSR must be greater than the CACC initially. The size of the numbers we can divide in fixed point is then limited by the size of the registers.

Realizing that the SR is our divisor and the ACC + MQ is our 70 bit dividend, let us now discuss exactly how the 7090 will perform this division.

The 7090 must do 3 things in order to divide. In fact, they are almost the exact opposite of multiply.

1. It must be able to determine if the divisor will go into the first part of the dividend (is SR = ACC). After determining it will go in, it must be able to subtract the divisor from the dividend.
2. Also, if it finds that the divisor will go into the dividend, it must put a bit in the quotient to show a successful reduction of the dividend (the divisor did go into the dividend one time).
3. After making this attempt at a reduction of the dividend, whether it be successful or not, then the dividend must be adjusted to the left so that we can make another attempt at a successful reduction; or, in other words, see if the divisor will now go into the dividend.

Let us take these 3 points one at a time and see how the 7090 is actually going to perform them. First, in order to determine if the SR, which is our divisor, will go into our dividend we look at the high-order part of our dividend, which is in the acc. To make this decision the 7090 has only to gate the Acc into the adders and the SR into the adders with one of the numbers in complement form. Since we cannot complement the storage register, it would sound logical that the Acc is the one which is complemented. Now we will test for a Q carry or no Q carry, which will tell us if the SR is less than, or equal to, the Acc. If we do not get the Q carry, the SR will go into the Acc. Prove this to yourself by taking the numbers and performing this test. One comment here to strengthen the point we made earlier; if we do not get a Q carry on this first attempt we should get an illegal divide, the reason being that the SR must be greater than the Acc when we start the division.

The second part is that we must put a bit into the quotient each time we recognize that the divisor will go into the dividend. Our quotient is being developed in the MQ and each time we find the SR is less than or equal to the Acc. we have but to force a bit into our quotient.

The third part is in adjusting the dividend so that we can continually make the test to see if the divisor will go into it. We first try to divide into the high-order part (Acc) of the dividend. Whether it is successful or not, we have only to shift the dividend left and we can make another attempt. This would be similar in longhand division to bringing down the next low-order digit in the dividend in an attempt to divide again. Since we are shifting left, the high-order part of our dividend is being

lost and is gradually being replaced with digits from the MQ one at a time. Therefore, we can develop our answer (or quotient) in the MQ as we are adjusting our dividend. The bit is put in MQ35 if we have a successful reduction of SR into ACC. When we shift left to attempt further division, the MQ will be in position to accept the next bit in the quotient should we have another successful reduction.

As was the case with multiply, we are going to use the shift counter to control the number of shifts necessary. Therefore, when the shift counter goes to zero, it will tell us that we have attempted to divide all of the positions that are possible using 35 pos. registers.

DIVIDE FLOW CHART LOGIC

During Part 3 let us use Figure 5.3-20A on the DVH instruction (Page 65) in your Manual of Instruction. We will go through the flow chart block by block trying to answer what is happening and why it is happening. It is assumed that you have the over-all picture of divide and that you understand logically what we are attempting to do. You will now see how the 7090 actually does it.

Starting at the top of the flow diagram at the bottom of the E time block and going to the left, you will see that we take the ACC to the ADD in complement form at E4D3. We then take ADD back to the ACC at E6D1. Therefore, in the ACC at this time is the high-order 35 bits of the dividend in complement form. At E7 time we are going to get our divisor from the location in core storage and put it in the SR. We then gate the SR to the ADD and the ACC to the ADD at E8D3. Keep in mind that the ACC is in complement form.

Returning again to the bottom of the E time block, there is a decision block as this diagram is used for all 4 of the divide instructions. Assuming that we have a divide or halt, we will go down to the bottom of the decision block and set the shift counter with 43_8 at E3D1. You will recall from multiply that this is equivalent to 35_{10} and that there is a number of shifts required to look at all bits of the dividend.

Continuing down from this block, which sets the shift counter, there is a decision block checking for a Q carry. Let's discuss the logic of the decision that this block is indicating. If we do not have a Q carry, it means that the $SR \leq ACC$. If we do have a Q carry, it of course means the opposite; that is that the $SR > ACC$. Remember in Part 2 I mentioned the fact that we do not want a successful reduction (we do not want the divisor to be less than the ACC) on our first attempt; because if this was so, our quotient would be too big for the MQ to hold. Therefore, we will assume that we have numbers within the machine limits and the Q carry will be present. In other words, our $SR > ACC$ to start. We will come back and pick up the no "side" of this decision block at a later time.

Now go to the right and down the path where we see a divide shift at E11D1 and divide shift control. We will see that we shift the Acc P-35 and MQ 2-35 left. You notice that MQ1 is not being shifted left in that same block, but just to the left of it is a block which says complement MQ1 of ACC 35. You will recall back in E time that we complemented the CACC. Since division is going to require a subtractive process, we are going to keep the ACC in complement form at all times. Therefore, when we shift the next "low-order digit" of the dividend into the ACC, we shift it in complemented. To the right of the third block it says, "multiply and divide, step shift counter", which is logical if we are to keep track of the number of shifts.

As we go to the right of the flow chart, which means we can divide, we see that Q carry and $SC \neq 0$ allows us to set the sign of the MQ (quotient). The normal rule for setting the sign applies here; unlike signs, the quotient will be minus--like signs, the quotient will be plus. You will note, however, that we do not set the sign of the ACC because our ACC is to contain the remainder. As previously mentioned, the remainder is what is left over from the original dividend. Therefore, it should retain the same sign as the original dividend.

Going now to the left we see "divide shift control" at E11D1. This allows us then to shift the ACC P-35 and MQ 2-35 left. At the same time we step the SC. However, we complement MQ1 to Acc 35. The reason for this is that the high-order 35 bits of the dividend are in complement form in the ACC. Division will be a subtracted process (we want to subtract the divisor from the dividend each time we find the divisor is smaller or equal); so the Acc is kept in complement form throughout to make this subtraction when necessary. Therefore, when we shift left, we will complement the lower-order bits of the dividend being shifted into the Acc. After this we proceed to L time on the next page of the flow chart.

Now that we are in L time, the actual division is going to take place. During all of L time you will note that the SR and the ACC are being gated to the adders. Keep in mind, however, that the ACC is in complement form. Therefore, the output of the adders is the difference between the dividend portion in the ACC and the divisor which is in the SR. This difference is what we want to take back into the ACC if we find the divisor is smaller or equal. To determine whether or not the divisor (SR) is smaller or equal we are going to check the Q carry out of the adders.

If we do have the Q carry, it means that the CSR (divisor) is larger than that part of the dividend in the ACC. If we do not get a Q carry, it means the CSR is smaller or equal to that part of the dividend which is in the ACC. The latter condition means that we should put a bit in our quotient and we should take the difference back to the ACC as the remaining part of our dividend. Let's look further at the logic of these 2 blocks.

Take two numbers--5 and 7--three bit binary numbers. In the first case let's assume the SR has 5 and the ACC has 7. We know that the ACC is in complement form. Therefore, if we complement 7 we will have zeros in the ACC. If we add these numbers together in the adders, the output would not result in a Q carry. In other words, no Q carry means the True ACC value is larger. We said the Acc originally contained 7; therefore, we have proven this is correct.

Let's take the exact reverse case now where the SR has 7 and the ACC has 5. If we complement the ACC, as is the case, the ACC now contains 2. If we add 2 and 7, we do get a Q carry and this will indicate that the Acc is smaller which is correct. This time the Acc. contained 5.

Now take the condition where the SR and the ACC are equal; say that they both contain a binary 5. For reality let's complement the one in the ACC. We add them together in the adders and we find that we do not get the Q carry. We said the no carry would mean that the SR could go into the ACC; and if they are equal, then this is a True statement. Therefore, no Q carry means that the number in the SR is "smaller than or equal to" the true value of that part of the dividend which is in the ACC.

Now look at the logic which says that when this condition arises (when there is no Q carry), we take the output of the adders back to the ACC and have the correct difference between the SR and ACC. If you will review the previous examples using 5 and 7, you will note that if the ACC is \leq SR the true difference is available from the adders in complement form. This is exactly what we want on divide, however, because that portion of the dividend in the ACC is to remain in complement form throughout the operation. Therefore, each time we find the SR will go into the ACC we merely take the ADD back to the ACC (we have the difference; it is in complement form--perfect!).

The only time our diff. out of the ADD would be the true diff. is when the SR is greater than the ACC. However, this condition would result in a Q carry; so, we would not take ADD back to ACC nor would we put the one in the quotient to indicate a successful reduction. Now let's return to the flow chart and take it one block at a time, starting with the Q carry decision block in the upper left corner.

Assuming the first time that we do get a Q carry, we bypass 2 blocks, and the shift counter is not equal to zero (it should have 43_8 if this is the first time through). We now go up to "divide shift control" at either L3, L7 or L11. We shift the ACC and MQ to the left one place, step the SC to indicate that the shift has taken place and complement MQ1 to acc 35 to maintain our high-order dividend positions in complement form. We then return to where the SR and ACC are being gated to the ADD all during L time (actually, we could have returned directly to the Q carry decision block since these 2 blocks are constantly active).

This time let's assume we do not get a Q carry, which tells us to put a bit in MQ35. At the same time we put a "one" in MQ35 we are going to take the output of the ADD back to the ACC. In other words, we have reduced the dividend by the amount of the divisor and placed the bit in our quotient representative of this fact. We then check the SC to see if it's = 0. If it is not, we left adjust the dividend (continuing to keep our complement form in the ACC) and return to check for another Q carry. This procedure continues until we have taken a total of 35_{10} shifts (43_8) since we started with 35_{10} in the shift counter.

The SC will be stepped during the 12th L cycle at L7. We will return and check for one additional Q carry. The last thing we do on divide is possibly take a reduction cycle (if we do not get a Q carry).

With the shift counter now equal to zero, we will get "multiply/divide end op" at L10 time. End op will take us to I time of the next instruction. Since we have a DVH instruction, we will go to the left. T1 would be off, so we come back to the right and complement ACC to ADD, take ADD back to the ACC, and this will be the end of the DVH instruction (What was in the ACC at I0 and I2 time after ending op was the remainder; but remember, it was in complement form. Therefore, we must re-complement, and we wind up with a quotient in the MQ and remainder in the ACC).

One thing about divide which is quite different from that of multiply is that we have no option for shifting other than the 3 specific times L3, L7 or L11. This means that divide DVH is not a variable cycle instruction. It will always take the same number of cycles if we are to divide. On multiply we could (if we had 2 or more consecutive zeros in our multiplier) shift rapidly; therefore cutting down the number of cycles. On the DVH instruction we must make a comparison between two 35 bit numbers (SR and ACC), to find out if one is smaller or equal to the other before we know if we should put a bit in our quotient or not. Because of the time this takes in the adders, we cannot make divide as rapid as multiply.

Let us now return to the beginning of DVH instruction where we perform the test for the first Q carry, to determine whether or not the 7090 can handle these size numbers. At the bottom of the flow chart on Page 65 we see if we do not get a Q carry it means that the $SR \leq ACC$ the very first time we make the test. This would result in a 36 bit quotient which we cannot handle in the MQ. So we go to the block which says "to divide check circuits" and it goes over to the next page at point A in our flow chart.

Coming in at point A on page 66 you will notice that we turn on the divide check trigger at E11, we get divide check end op at E11, and we turn on T1 at E11. Let us take them one at a time starting with the middle one which says "divide check end op at E11". This is too late for us to actually get the end op trigger on during E time. Therefore, we must proceed into the L cycle. During the L time we can then end and go to I time of the next instruction. So even when a divide check condition exists, we take a minimum of 3 cycles--one I, one E and one L.

Let us now look at the other two conditions starting with the one at the left. It is "turn on divide check trigger at E11". The divide check trigger is a programmable indicator which you must interrogate with the DCT instruction. It will turn on at this point; it will not turn off until it has been tested with a DCT instruction.

To the right it says "turn on T1" at E11. The purpose of T1 is to halt the computer should this be a DVH instruction. If it was a DVP instruction, T1 would serve no purpose. On DVH this trigger actually stops us, as you will see if you proceed down through L time to I time of the next instruction and go to the left on that decision block. On DVH or VDH if T1 is on this turns on the master stop tgr. at I5. The B cycle intrpt. tgr. will come on the very next A11 pulse.

If you recall manual control circuits, you will know that the machine actually stops at I5 time. The reason for stopping this early is so the contents of the registers will not be changed. If we should stop as late as I7 time, which we do under some conditions such as single step, etc., the CSR would be lost.

VDH - VDP

Let's return to page 65 at the top right-hand side of the flow chart and now assume a VDH or VDP. The main difference here, as you may suspect, is only the shift count which now will be determined by the "count field" of the instruction. Instead of taking 35 shifts we may have more or less depending on the count in the decrement portion of the instruction.

You can see that as we take SR S-1-35 to adders P-35 we also gate adders 3-17 to the Addr Sw, then Addr Sw 12-17 into the SC at E2D1. Assuming that the SC \neq 0 we will proceed as we did on the regular divide. If the SC = 0, we will immediately end op and go to "point B" on the flow chart which is on page 66. We do this in time to end op during E time and proceed to the next instruction. Therefore, it is possible that the variable divide instructions may take 2, 3 or many more cycles.

Since we can take 3 reduction cycles for each L cycle, it can easily be determined the length of a VDH or VDP. You have only to recall that the shift counter must go to zero prior to L10 time to allow us to end op. As an example: If we had a VDH with a count field equivalent to 7, we would have to take 3 L cycles to perform the divide.

DIVIDE SUMMARY

There are four fixed point divide instructions--DVH, DVP, VDH, and VDP. These instructions are identical except that the variable length instructions have a count field in the decrement portion. This count determines the actual number of attempts we will make to divide the SR (divisor) into the ACC (dividend). This also will be the limiting factor on the size (number of digits possible) of our quotient. DVH or DVP are fixed counts in that they will always attempt 35 reductions and will result in a 35 bit quotient with a 35 bit remainder in the Acc.

Basically, the division process occurs in the following manner: We have a 70 bit dividend in the Acc at MQ, the most significant portion of which is in the ACC. Our divisor is in the SR. We make a test to determine if the SR will go into the ACC. If the SR does go into the ACC, we put a bit in our quotient (which is the MQ) and subtract the SR from the ACC. After doing this, we shift the entire dividend and quotient to the left one place and see if the SR will now go into the ACC. This series of events takes place a total of 35 times, which results in a 35 bit quotient and a 35 bit remainder.

The logic of this series of events is this: Since the ACC (the high-order part of our dividend) is in complement form, we know that if we do not get a Q carry the SR is less or equal to the ACC. Therefore, we take the difference between these two numbers back to the ACC and place a bit in our quotient.

However, if we do get a Q carry, it is logical that the complemented number is smaller than the number in the SR; therefore, it will not go into the ACC and we do not want to reduce the dividend by the amount of the SR. For this condition, we shift the dividend and quotient one place left and make the same test over again.

To keep track of the shifts we use the SC as we did on the MPY instruction; and when the SC is zero, we will end op and proceed to the next instruction. The only thing remaining to do in the beginning of the next I time is to re-complement the remainder which is in the ACC.

It is possible to get a divide check on any divide instruction; in which case we turn on a "programmable indicator" which can be interrogated by the DCT instruction. We also turn on another trigger (T1) which will stop the computer should we have used the halt instruction option.

The number of cycles that the divide instructions may take depends upon several things. First, on the variable instructions, it depends on whether or not the shift count is zero. Should it be zero, we will end op immediately in E time and take on only 2 cycles.

We also check to see if it's possible that the numbers we are using would result in too big a quotient. If this is so, we will get a divide check (if the $SR \leq ACC$ initially on the divide instruction).

Lastly is the count itself on the variable instructions. It may vary from 1-35₁₀ (or even larger although it has no logical significance) which would determine the number of cycles the instruction would take.

07.02 INTRODUCTION TO EXPONENTIAL MATH AND FP NUMBERS

07.02.1 Exponential Math.

Prior to studying how the Floating Point instructions actually are performed in the machine, a thorough understanding of exponential math is desirable. After all, one of the reasons fixed point algebraic operations are easier is because of their close similarity to the basic math we use every day. This is why we will first spend a few minutes reviewing exponential numbers and math. Once knowing what we have to do, it will be easier to understand how we do it.

One other point: Since it is easier for us to use the decimal system, we will use it for the review. After we know WHAT exponential numbers are, WHY exponential numbers are desirable in computer usage, and HOW exponential math works, we can then readily convert to the binary system; which is the 7090 language.

WHAT

We will start by trying to establish WHAT an exponential number is. For this, let's use the following examples.

	Normal No. Representation		Same No. in Exponential Notation
(a)	31,000,000.	=	3.1×10^7
(b)	.000000017	=	1.7×10^{-8}
(c)	100	=	$.1 \times 10^3$
(d)	.001	=	$1. \times 10^{-3}$

You can see that a number expressed exponentially is made up of two parts. Let's give names to these parts for easy reference.

In example (a) the digits 3.1 would be called the FRACTION part of the exponential number. You can see that the FRACTION could be either a proper or improper fraction (definition of a proper fraction is a fraction whose value is less than one - improper fractions are equal to or greater than one) (see examples c, d). Placement of the decimal point in the fraction is of value only when used as a reference point to the other part of the number called the EXPONENT.

The exponent part of example (a) would be 10^7 . The 10 means "base 10" since we are in the decimal numbering system. The 7 means "to the seventh power, or seven (7) decimal places. Since it is a +7 (only the minus signs will be shown), the decimal places are to the right of the decimal point in the fraction. In examples (b) and (d) the negative exponent means to the left of the decimal point in the fraction.

Knowing now what the 2 parts of exponential numbers are called and how to use the exponent part in determining the true value of the fraction, carefully analyze each of the examples given to see if they are correct.

WHY

We are ready at this time to now answer WHY exponential numbers are desirable for computer usage; especially on the 7090 system because of its easy adaptation to handling scientific calculations. To demonstrate why they can be desirable, let's use the following two examples.

- (a) Here is represented the projected national debt if the Republicans get in office again (as a normal number):
263, 731, 600, 000, 000, 000, 000, 000

Here is the same figure in the much easier to designate exponential notation:
 2.637316×10^{23}

- (b) Now let's go the other way and see an extremely small number--let's say the radius of a pinpoint in miles (as a normal decimal number):
.000 000 000 000 000 000 000 1712

Taken as its proper exponential notation:
 1.712×10^{-22}

Although the numbers used in the example were quite meaningless, the point to be made is the ease with which extremely large or small numbers can be represented exponentially in less digit positions! In the scientific field I'm sure you can see the importance of being able to handle this wide range of numbers as easily as possible. By this means of representing numbers, we greatly increase the use of our machine registers which are limited to sign and 35 bit positions size. That, then, is the WHY of exponential math of the 7090.

HOW

Now to see just what there is to exponential math (or floating point math, as it is known on IBM computers). How does it work? Since it can be assumed we all know how to add, subtract, multiply and divide numbers such as 2.1, 56.7, etc., we will approach it from this point. In fact, we will take them in that exact order and actually prove that what you do in the "longhand" method is identical to exponential math when the rules are analyzed. Don't panic at the thought--this is not a complex or drawn out feat to accomplish (we hope!).

It will be done in this fashion: We will analyze an example using numbers in normal decimal notation. Then we will state the rules and perform the math using the same numbers in exponential notation. Since we want to be able to express the same numbers in both forms, we will use relatively common ones (i. e. not extremely large or small numbers) for ease of writing them.

ADDITION

The numbers we will add are 26310000. and 341.30 written in their normal form. So let's add them.

$$\begin{array}{r} 26310000. \\ \quad 341.3 \\ \hline 26310341.3 \end{array}$$

Not a difficult task once we lined up the decimal point, was it? Now let's represent the same 2 numbers exponentially.

$$\begin{array}{lll} 26310000. & \text{could be written} & 26.31 \times 10^6 \\ 341.30 & \text{could be written} & 3.413 \times 10^2 \end{array}$$

And we still want to add them.

The rule for adding exponential numbers is: Move the decimal point in one of the fractions and adjust its exponent until the exponents are equal. Line up the decimal point in the fraction part and add. As in the example below, we want to make its exponent equal 10^2 so we can add this number to 3.413×10^2 .

In order to make its exponent 10^2 , it is necessary to move the point in the fraction 4 places to the right (or move the fraction itself 4 places left--same results), thusly:

$$26.13 \times 10^6 = 263100. \times 10^2$$

Is it not still the same value??

Now that we have satisfied the (b) part of our rule for adding, we can line up the decimal points of our fractions (rule a) and add.

$$\begin{array}{r} 263100. \quad \times 10^2 \\ \quad 3.413 \times 10^2 \\ \hline 263103.413 \times 10^2 \end{array} \quad \text{is our answer}$$

which equals 26310341.3

Let's now look at what we have accomplished "logically" in relationship to our first example using normal numbers. The exponent is telling us where the true decimal point belongs in relation to where it appears in the fraction. Therefore, if the decimal point in the fractions is lined up and the exponents are equal; then the TRUE decimal points are lined up, are they not? If this is still confusing, it would be worthwhile for you to review the previous example more thoroughly.

SUBTRACT

The rules for subtract are the same as those for add except the fraction parts are naturally "subtracted"--not added. For this reason the following examples will be given, without additional discussion, for your own analysis.

Subtract 341.3 from 26310000.

$$\begin{array}{r} 26310000. \\ - \quad 341.3 \\ \hline 26309658.7 \end{array}$$

Same math done using exponential notation.

$$\begin{array}{r} 2.631 \times 10^7 \\ - 3.413 \times 10^{-1} \\ \hline \end{array}$$

Must first equalize exponents--so let's make the 10^7 equal to 10^{-1} just for kicks.

$$2.631 \times 10^7 = 263100000. \times 10^{-1}$$

(I know it's messy, but anything to prove a point)

Now we're in business.

$$\begin{array}{r} 263100000. \times 10^{-1} \\ - \quad 3413. \times 10^{-1} \\ \hline 263096587. \times 10^{-1} \end{array}$$

Which equals 26309658.7, correct!!

If you thoroughly understand the use and value of the exponent, then the add/sub operations should seem basic to you now. If not, it would be best to study them again.

MULTIPLY

As in previous examples, we will first multiply the 2 numbers in their normal form.

$$\begin{array}{r} 32.51 \\ \times 6.3 \\ \hline 9753 \\ 19506 \\ \hline 204.813 \end{array}$$

(2 dec. places)
(1 dec. place)

Add no. of dec. places

Mark of in product

We can express the same 2 numbers exponentially as $(.63 \times 10^1)$ times $(.3251 \times 10^2)$. The rules for multiplying these 2 numbers are:

- Multiply the fraction parts as you would any dec. fractions
- Give the product an exponent equal to sum of the exponents of the two numbers; as in the next example.

4 dec. places	$.3251 \times 10^2$	Add the exponents
<u>2 dec. places</u>	$.63 \times 10^1$	
	9753	
	19506	
6 dec. places	$.204813 \times 10^3$	

The answer equals 204.813, right??

By adding the exponents (which represent the true number of dec. places in our fraction), we are basically saying then that the number of dec. places in our product should be equal to the total number of places in our multiplier and multiplicand! Isn't this exactly what we did in our first multiply example using normal dec. fractions? If you say yes, we agree and can proceed to divide. If not, perhaps this one additional example will help (and let's use a negative exponent for a change) (you may note that they are the same numbers in reality).

no dec. places	3251×10^{-2}	Add exponents
2 dec. places	$.63 \times 10^{-1}$	
	9753	
	19506	
	2048.13×10^{-1}	
	2 dec. places	

Answer equals 204.813

So far so good! We have shown that exponential math handles the dec. point the same way as our "longhand" methods. The only difference is our use of the exponent to designate its true position. If we can make a similar analogy with "longhand divide", we're in!!

DIVIDE

Again, let us start with an example of our longhand method using "normal" numbers so we can later compare logic.

	6.12
5.3/32.436	318
	63
	53
	106
116	106

You may have noted that we moved the dec. point to the right in our dividend the number of dec. places in our divisor. Or, what we really did was subtract the number of dec. places in our divisor from the no. of decimal places in our dividend; the purpose of which was to determine the position of the dec. point in our quotient.

All of that leads us to the rules for dividing exponential numbers.

- (a) Divide the fraction parts as any normal dec. fractions (including movement of dec. point)
- (b) Subtract the divisor exponent from the dividend exponent.

Here's an example using the same numbers:

$$.53 \times 10^1 / .32436 \times 10^2$$

Divide Fractions

$$\begin{array}{r} .612 \\ .53 \overline{) .32436} \\ \underline{318} \\ 63 \\ \underline{53} \\ 106 \\ \underline{106} \\ 000 \end{array}$$

Subtract Exponents

$$\begin{array}{r} 10^2 \\ -10^1 \\ \hline 10^1 \end{array} \quad \text{Exponent of product}$$

Answer equals $.612 \times 10^1$
Or, (expressed normally) 6.12

Let's do one more example, using random numbers and checking our answer by multiplying our divisor times the quotient which should equal our dividend.

(Note)
The divisor fraction has no dec. places, so it is not necessary to move the point in the dividend fraction.

$$\begin{array}{r} 4.59 \times 10^{-2} \\ 231 \times 10^{-2} \overline{) 1062.31 \times 10^{-4}} \\ \underline{924} \\ 1383 \\ \underline{1155} \\ 2281 \\ \underline{2079} \\ 202 \times 10^{-6} \end{array} \quad \begin{array}{l} \text{Sub exponents} \\ \\ \\ \\ \text{Remainder} \end{array}$$

Let's check our answer:

$$\begin{array}{r} 4.59 \times 10^{-2} \\ 231 \times 10^{-2} \overline{) 1062.29 \times 10^{-4}} \\ \underline{459} \\ 1377 \\ \underline{918} \\ 000 \end{array} \quad \begin{array}{l} \text{quotient} \\ \text{divisor} \\ \\ \\ \end{array}$$

To add in remainder of
we must first equalize characteristics

$$\begin{array}{rcl}
 1060.29 \times 10^{-4} & = & 106029. \times 10^{-6} \\
 202 \times 10^{-6} & = & \underline{202. \times 10^{-6}} \\
 & & 106231. \times 10^{-6} \\
 & & \text{(or) } 1062.31 \times 10^{-4}
 \end{array}$$

Looks as if our proof is conclusive. However, one additional point merits mention; and that is the remainder and its exponent. If you recall, a "remainder" is what is left over (remaining) from the dividend after we have stopped dividing.

Example:

$$\begin{array}{r}
 5.47 \\
 6 \overline{) 32.87} \\
 \underline{30} \\
 28 \\
 \underline{24} \\
 47 \\
 \underline{42} \\
 5
 \end{array}$$

Five is our remainder, or what is left after taking 6 from 32.87 a total of 5.47 times. But what does this 5 represent?

5, .5, 50, or .005??!

Let's take this approach. If it is left over from the dividend, it must be representative of the portion of the dividend it was left over from (Oh boy!!). The last position of the dividend we divided into was the 7 (last digit brought down) which we can see equals 7/100 in reality. Since the 5 is "left over" from this division, it seems logical that it represents 5/100. Right? Let's reverse the process and see--

Add in the 5/100 remainder

$$\begin{array}{r}
 5.47 \\
 \underline{6} \\
 32.82 \\
 \underline{.05} \\
 32.87
 \end{array}$$

That convinces me--so in using exponential numbers we can do the same thing to get an exponent for our remainder which will represent its true value.

$$\begin{array}{rcl}
 & & 4.59 \times 10^{-2} \\
 231 \times 10^2 / 1062.31 \times 10^{-4} & & \underline{\hspace{1.5cm}} \\
 & 924 & \underline{2} \quad \text{Subtract no. of dec. places} \\
 & 1383 & \underline{10^{-6}} \quad \text{divided into .} \\
 & 1155 & \\
 & 2281 & \text{Exponent for remainder} \\
 & \underline{2079} & \\
 & 202 \times 10^{-6} &
 \end{array}$$

We can take the exponent of the original dividend and subtract the number of dec. places we have divided into. In the above example, it was 2 places (the 3 and the 1).

This relationship between remainder and dividend is important for complete understanding of floating point divide. This is especially true in working with "double precision" instructions such as the 7094 has.

This looks like conclusive proof that the exponential number is as flexible as plain old "integers". Also, that exponential math is, in reality, applying the same rules we are familiar with on an everyday basis.

This section has been quite basic, and several of you probably skimmed it lightly, which is O. K. ; but understanding it is a necessity if floating point is to be logical. If it is understood thoroughly, then the next section will be much easier and more interesting.

The fraction part of a floating point number is probably the easiest, so let's start there. In fact, there is only one basic difference between the fractions we were working with in the decimal system and those we will have in the binary machine language notation. That is; a floating point fraction must always be represented as a proper fraction (less than one, such as .5, .005, .9, etc). All digits in the fraction to the right of the point.

For our first example we will take the number 32_{10} and convert it to its floating point representation. Knowledge of number conversions is assumed.

$$32_{10} = 40_8 = 100\ 000_2$$

With the binary point shown = $100\ 000_2$

We now have the number converted--all left to be done is make the number a proper fraction, or move the binary point 6 binary positions to the left.

.100 000

Now the floating point fraction is finished; but it must have the proper exponent with it to show where that binary point really belongs in the fraction.

To do this, let's first see the breakdown of a 7090 word representing a floating point number.

S	1	8 bits thru	8	9	27 bits thru	35
Sign of Fraction		Characteristic			Fraction	

The fraction consists of 27 binary positions and is always treated as though the binary point were just left of pos. 9. It also has a sign bit (the fraction can naturally be a positive or negative number).

Positions 1-8 will contain our "exponent" or CHARACTERISTIC as it is called in F. P. Notation. (We will abbreviate "CHAR" for characteristic). So we now will refer to char and fraction in place of exponent and fraction.

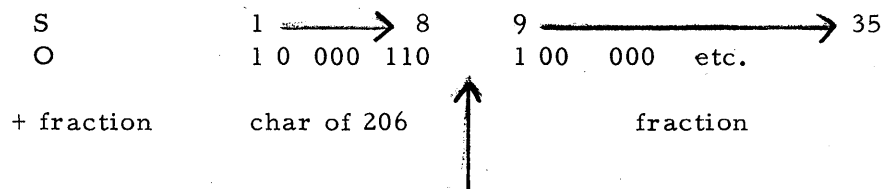
One other note to make in our previous example is that the "char" (positions 1-8) does not have use of the sign bit (that gives with the fraction). Still we must be able to represent a + or - exponent, right?

This we do by picking the mid point of the range of numbers that 8 binary positions can represent. Let's see, in 8 positions we can represent from 000_8 to 377_8 --or shown in binary: $00,000,000_2$ to $11,111,111_2$. The midpoint of this range of numbers would be 200_8 . Now we say any char. with a bit in pos. 1 (200_8 or larger) represents a positive exponent and those without a bit in 1 (less than 200_8) represent a negative exponent. The size of the exponent is determined by how far the char deviates from 200 (200 itself actually equals a zero exponent). Let's look at some typical examples. Remember, we are representing exponents in the binary system (base 2), not the decimal (base 10), because our fraction is a binary number.

<u>Char</u>	<u>Power of 2 Being Represented</u>	
200_8	2^0	
202_8	2^2	
207_8	2^7	
177_8	2^{-1}	
171_8	2^{-7}	
210_8	2^{10}	(power expressed octally)
210_8	2^8	(power expressed in decimal)

Now let's put the proper characteristic with that fraction we figured out a few minutes earlier.-- $32_{10} = 40_8 = 100,000$ --We moved the binary point 6 binary places left to make it a proper fraction: $.100,000$

We must indicate, in the char, where the binary point really belongs or a char equal to 26, which is 206_8 . In binary the char. and fraction would look like this:



"Imaginary" binary point assumed by the 7090 so the char value has meaning and can treat all fractions as "proper fractions"

Let's do one more example with a negative number-- $-.017578_{10}$.

$$-.017578_{10} = -.011_8 = -.000\ 001\ 001_2$$

It is already a proper fraction with the binary point to the left, so we do

not have to shift it. Therefore, the char. we give with the fraction shown this way should equal 2^0 or 200 (the binary point is correct as shown in the fraction, so no exponent is needed to show where it really belongs).

	S	1	8	9	35
	1	1 0	0 0 0	0 0 0 . 0 0 0	0 0 1 0 0 1
<u>Neg. fraction</u>		Char of 200			Fraction

This would not be the case normally, however, because this is what is referred to as an UNNORMALIZED F. P. number (no bit in 9, the high order position of the fraction). So, let's make it a NORMALIZED F. P. number by shifting the fraction to the left (or the point to the right if you prefer) like so:

Unnormalized	=	200 . 011 ₈
In Binary	=	10 000 000 . 000 001 001
Shift fraction left 5 to normalize 5	=	10 000 000 . 1 00 1
Sub "5" from char so value is not lost	=	01 111 011 . 100, 1
Now Normalized	=	173 . 44 ₈

The decimal fraction $-.017578_{10}$ is equal to -173.440000000_8 as a 7090, normalized, F. P. number. It has a sign bit for the fraction, a characteristic which can represent a + or - exponential value to the base 2 and a fraction part which will be treated by the char as a "proper" fraction (binary point to the extreme left).

The two types of F. P. numbers are "normalized" and "unnormalized". The normalized, by definition, contain a bit in the high order position of the fraction (pos. 9). It can also be put this way: The normalized fraction is \geq one half--the unnormalized $<$ than one half.

One other "definition" is that of a normal FP zero and an unnormal FP zero. These I will show by the following examples.

	<u>Char</u>	<u>Fraction</u>	
(a)	000 ₈	.000000000 ₈	"normal" zero
(b)	132 ₈	.000000000 ₈	"unnormal" zero
(a)	Char <u>and</u> Fraction both = to zero		
(b)	Fraction <u>only</u> = to zero		

07.03 FLOATING POINT INSTRUCTIONS

Objectives:

A satisfactory understanding of the Floating Point instructions should enable you to accomplish the following ON EACH INSTRUCTION TYPE.

WITHOUT REFERENCE:

1. Be able to perform correctly the Floating Point math, in the same manner as the 7090, on any given Floating Point numbers.

WITH REFERENCE:

1. Explain the reason and need for any block of logic in the flow charts for a particular instruction.

NOTE: One method of doing this would be to site example F. P. numbers showing necessity for a particular block or blocks.

2. Achieve a satisfactory grade on the quiz following this material covering MPY/DVP and Floating Point instructions.

7090 FAD (Floating Add) Overall Logic

For this discussion of 7090 logic it will be assumed that a thorough knowledge of the following points has been obtained. You should be familiar with the terms characteristic and fraction as they apply to the 7090 system. The rules of exponential or floating point math should also be familiar to you. We can then concern ourselves in this discussion of how the 7090 logically performs exponential math.

Floating Add, FAD, is accomplished in three basic steps. First is the equalizing of the characteristics; second is the addition of the fractions; and third is the normalizing of our answer. The last part is necessary because we always wish to wind up with a normalized answer providing we are using normalized floating point numbers.

First, let us start with the equalizing of the characteristics. One of our numbers to be added will be placed in the ACC by a previous instruction; probably a clear and add, CLA. The other number will be brought into the SR during execution of the FAD instruction itself. These two numbers, then, must be added; but prior to the addition of the fraction, their characteristics must be made equal.

We have said earlier that we can equalize characteristics by adjusting the binary point in the fraction or by shifting the fraction itself. In a 7090 floating point word, the binary point is always assumed to the left of position 9. Therefore, the 7090 will adjust the characteristic by shifting the fraction.

We now have the option of either shifting the fraction to the right or to the left, depending on whether we try to make the larger characteristic smaller or the smaller characteristic larger. Let's assume first of all that we have, in the ACC register, the number with the larger characteristic and we wish to make it smaller (so that it may equal the characteristic in the storage register).

If we were to shift the fraction in the acc. left, which we would have to do in order to reduce our characteristic, the digits in the fraction being shifted past position 9 would be lost. As you can see, this would not be a desirable result since these would be the most significant bits or digits in our fraction. So let's take a look at our alternate method of equalizing characteristics.

With the smaller number (or the number with the smaller characteristic) in the ACC, we would want to make the characteristic larger. This can be done as we shift the fraction to the right. Since we can shift the ACC right, with ACC 35 being shifted into MQ9, we would not lose bits in our fraction. That is, of course, assuming that we did not shift a significant number of times so we shifted past MQ pos 35. Let's take a look at how the 7090 actually does accomplish this.

When the number is brought into the SR during FAD, the first thing done is to test to see which number has the smaller characteristic (the ACC or SR). If the smaller is in the SR, then they are swapped. If not, they are left the way they are. In either case, the difference in the characteristics is placed in the shift counter and the fraction in the ACC will be shifted right the proper number of places. If this characteristic difference is too great, I am sure that you can see that some bits may be lost in our fraction from shifting past the MQ pos. 35. When this shifting has been accomplished, we have theoretically equalized our characteristics; and our "reference" binary points remain lined up between bit positions 8 and 9 in our SR and ACC. Of course, this is only an imaginary point for reference for all characteristics in 7090 format.

The actual addition of the fraction now takes place. This, if you recall the add instruction, is going to be handled in exactly the same way except only bit positions 9-35 of the SR and ACC will be added together. However, we must not forget that the fraction uses the sign position and complement addition may take place.

We will now normalize our answer. As you can see, should we have to subtract fractions due to unlike signs, it is possible that the fraction of our answer will not contain a bit in position 9. Since we want our final answer to be normalized, if possible, a test will be made to see if the fraction contains a bit in this position. If it does not, the ACC and MQ fraction positions will be shifted left until a bit is shifted into 9. A count of the number of positions that it is necessary to normalize is kept by the shift counter. This number of shifts will then be subtracted from the characteristic of the ACC. We now should have in our ACC a normalized floating point answer. Of course, had the ${}^F\text{ACC}$ and ${}^F\text{MQ}$ been all zeros, this normalizing would not be attempted.

You probably have already noticed that it is possible for us to have some bits left in the ${}^F\text{MQ}$. These bit positions in the ${}^F\text{MQ}$, 9-35, will also be given a characteristic. This characteristic will be the ${}^C\text{ACC}$ minus 27_{10} . This is a way of saying that the bits in the ${}^F\text{MQ}$ are an extension of the ${}^F\text{ACC}$. However, they are 27_{10} binary places removed. This part of the number in the MQ would be rather insignificant in comparison to the number in the ACC and would be useful (normally) only under "double precision" instruction; such as on the 7094.

7090 FSB (Floating Subtract)

A similar discussion of the logic of FSB will not be necessary because of its similarity to FAD. If you recall fixed point ADD and SUB, you will remember the only difference was when the number to be added or subtracted was brought into the SR. SUB differed from ADD only in that the sign of the number being brought from storage was inverted. This is exactly the same case of the same relationship FSB has to FAD. As the floating point number is being brought from storage, the sign of the fraction is inverted. It proceeds from that point identically to FAD; all the way from equalizing characteristics to normalizing the final answer.

FLOW DIAGRAM TEXT

Floating Add

In Fig. 5.3-23A on Page 72 in the green Manual of Instruction. For our first trip through the flow chart, we will use the example where we CLA location X which contains a floating point number of 204.41_8 . We will FAD location Y to this which contains the floating point number 206.55_8 . As we go through the flow chart, you will notice there are several blanks which you must fill in. Fill in these blanks using the two examples which I have just given.

At the beginning of the FAD instruction, the objective is listed for the E cycle. It says this should bring the char. and fraction to the SR. The char. and fraction that we are going to add to should already be in the ACC from a previous instruction; such as CLA.

After the E cycle we are going to proceed to L time; however, the L time will be broken up into 5 different L times by use of the tally counter. The tally counter is just a group of 5 triggers which we will turn on and gate with L time in sequence. The object of the tally counter is to make one L time different from another L time during the same instruction.

More than one L cycle may be taken with the tally counter at 1, 2, etc. However, when L time 2nd step (L time & tally counter 2 tgr) is completed, the tally counter will be stepped to 3. Now we have "L time, 3rd step".

The tally counter is stepped at 11 time of the cycle and goes as high as 5 (L time, 5th step). In its normal (reset) condition tgr. number 1 is on, just waiting for a F.P. instruction. We have, from this point, to merely step the counter to achieve L time 2nd step, 3rd step, etc.

We have proceeded to L time of 1st step on FAD. During L time 1st step we should align the fractions and test to see if the alignment of the fractions will exceed the capacity of the registers. Since we are going to align the fractions by shifting the ^FACC to the right, if we should shift more than 54_{10} times; we would lose all bits which we might have in our ACC fraction.

We must first determine which number has the smaller char. This is done, as you can see on the flow chart, by taking a 1 to adder 8 along with the complement of the ^CACC at A0D3. We are also taking the entire SR to the adders during first step L time. We then check for a "Q carry" condition. If we do have a Q carry, it means that the smaller char. (or their equal) is already in the ACC. If we do not get a Q carry, it means that the smaller char. is in the SR; therefore, we must exchange the two numbers.

At the same time we were making this decision, you will notice that the _____ is being reset so that we can shift the ^FACC into the _____ when we do align the fractions. We also turn on trigger _____

With the example stated earlier, we started with the char of _____₈ in the ACC and a char of _____₈ in the SR. Therefore, the result of our comparison would be (a) (no) Q carry,₈ and we would have to exchange the two numbers. Exchanging of the two numbers takes place in this fashion. The SR is gated to the adders all during first step L time. We gate the ACC to the SR at _____ and take the output of the adders into the ACC at _____. The exchange of the characteristics and the fractions is completed at this time. You will note that the SR and ACC signs are exchanged directly and do not go through the adders.

There is a note on your flow diagram saying that ACC P is also gated to the SR sign pos. at this time. This condition could cause an error if ACC P position did contain a bit and the fraction in the ACC actually had been plus. However, we will talk about this condition later on as it is, in some respects, a logic error.

Now that we have made the test to see if exchanging the numbers was necessary, before we attempted to align fractions, we now want to compute the char. difference. This is done on the flow chart where we take a "1 to adder 8" as we complement ACC 2-8 to the adders. This is at _____ time. This logic is this: We are going to gate the comp. of the smaller char. into the adders. We get the 2's complement with the "hot 1" going into adder 8. The difference is put in the shift counter, so we will know how many places to shift to align fractions. We can say that the ^CACC is smaller because (by this time) in first step L cycle we must have the no with the smaller char. in the ACC. If it was not there initially, then we exchanged them at A6 time. Therefore, the char. difference will be in true form as a result of this computation.

Next on the flow chart they show a decision block which is checking the output of adder 1 or 2. This is where they are going to determine if the char. difference is too large for our registers to handle. You will recall that we said a char. difference of more than 54₁₀ (which is equal to 66₈) would mean that our entire fraction would be lost when we shifted to the right. However, 54₁₀ (66₈) is quite a difficult number to test for. It is much easier for us to test for a difference of 77₈. This is done by checking the output of adder 1 or 2. A bit out of ADD 1 or 2 would indicate the char. difference is at least 100₈. Of course, if the char. difference falls between 66₈ and 100₈, we would still lose all of our fraction in the ACC; but we would attempt to equalize the characteristics nevertheless.

In our particular case, the difference is 2. Therefore, we will not have a bit out of adders 1 or 2 as a result of computing this difference. This 2 will be put in the shift counter at _____. Now we know exactly how many places to shift the fraction to the right so that our binary points are lined up. You can also see at this time that if our char. difference was greater than 77₈, we would clear the ACC Q-35. This would result in a normalized zero in the ACC. This is not our case so we will proceed to "L time 2nd step" on the top of page 73 in the flow chart.

During L time 2nd step the major objective is to shift the ^FACC the number of places necessary. If you look at the actual blocks, you will see that we shift the ACC 9-34 right every clock time. We shift MQ 1-34 right every clock time; however, ACC 35 goes into MQ 9. Therefore, even though we shift the entire MQ to the right, MQ 9 is only able to get bits coming in from ACC 35. You will recall that the MQ

was reset back in first step L time so there are only zeros being shifted from MQ 1-8. The _____ is stepped and we shift right twice. Then we will proceed to L time 3rd step when the SC = 0.

During L time 3rd step we do two things: We are going to add the fractions and check for possible "7090 double precision" handling. In our particular case we will go through this part of our flow chart exactly as our example takes us. The first thing to do is check _____ to see if it's equal to zero. In our particular example we would only have shifted the ^FACC twice; therefore, the MQ would still be equal to zero and we would turn on _____. This, as you will see at a later date, is a test being made which determines if 7090 double precision handling is necessary. Our particular example does not require it, so T2 is turned on.

Next, we come to SR and ACC signs decision block. If the signs of our fraction are unlike, then we must do a complement addition (subtraction). Our signs are alike so we proceed down from the decision block and perform true addition. This is done by simply gating SR to the ADD and ACC 9-35 to the ADD at _____. We also take ADD Q-35 back to the ACC at _____. We now have added the ^FSR and ^FACC. You will note at this particular time that we gate the SR to the ADD and we take all of the ADD back to the ACC. We are not only adding the fractions; but we are giving the ACC its proper characteristic at the same time. It should have the char. of the larger, which is being gated in at this time from the SR.

Off to the right there is another circuit which is testing for a special condition at this time. Our example would not have this overflow condition so the column 9 carry trigger would remain off. We will come through this chart another time with an example having this special condition. Now that we have completed adding the fractions and have given the ACC the proper char. we will proceed to 4th step L time at the top of page 75.

Reading the objectives for 4th step L time, we find that the F_{ACC} gets "zero tested" and "double precision handling" would be accomplished if necessary. In our example only the "zero test" is significant. This is done by gating the comp of the F_{ACC} into the ADD with a "hot one" to ADD-35. A 9 carry would indicate the $F_{ACC} = 0$ and turn on the 9 carry trigger.

Our path is from the "OFF" side of the "FP trigger" decision block ($F_{MQ} = 0$ back in 3rd step) anded with the "NO" condition from the ADD 9 carry decision block ($F_{ACC} \neq 0$ - 4th step). This takes us down to the "ACC-9 = 1" decision block at the bottom of Page 75.

The decision here is that if $ACC - 9 = 1$, then we do not need to normalize; so do not go to 5th step L time. End op in 4th step and proceed to Fig. 5.3.23E (enters at lower left on Page 76). Our next example will get us into 5th step.

"F. P. End op" causes us to proceed to I time of the next instruction. However, during the 1st half of this cycle we must do some minor housekeeping on the FAD. Read the objectives listed and we will take them in order.

To compute the C_{MQ} , the 2's comp of 27_{10} is forced into the adders along with the C_{ACC} . This subtracts 27_{10} from the C_{ACC} and is the proper char for the F_{MQ} , even though (in our example) the fraction part is zero.

Next, the sign of the MQ is set to agree with that of the ACC. If the no. in the MQ is to be an "extension" of the no. in the ACC, it sounds logical that the signs should agree.

Now the decision block (which is logically an OR ckt) looking at T2 and the "9 carry tgr". "T2 ON" would mean the $F_{MQ} \neq 0$; so in our case, it will be OFF and that gives us one vote for the "NO" side of the block. HOWEVER, the "9 carry trigger off" condition means the $F_{ACC} \neq 0$ and this is true! So, we will give the MQ a char even though its fraction part is zero.

This completes our 1st pass through the FAD flow chart. Our next example will pick up several points, purposely avoided this trip.

SECOND EXAMPLE

The points we will stress on this example are:

1. Unlike signs - so we will subtract (comp. add) the fractions
2. 7090 "Double Precision Handling" - for which unlike signs are necessary
3. Normalizing

Since "7090 DP" is probably the most confusing in this area, I would like to discuss logically WHAT it is before starting this second example.

The three conditions which must exist for this special handling are:

- a. Unlike signs (the SMALLER fraction must be subtracted from the LARGER)
- b. The SMALLER fraction must be in the ACC and MQ
- c. The ^FMQ must ≠ 0. (This is basically an extension of condition b).

To show an example of 7090 DP let's use the following numbers.

$$\begin{array}{l} \text{CLA} \quad \text{X} \quad \text{L}_X = -202.400000001 \\ \text{FAD} \quad \text{Y} \quad \text{L}_Y = +204.420000000 \end{array}$$

The small char is in the ACC so I will now show the register contents AFTER the char diff has been used to align the fractions.

$$\begin{array}{l} \text{SR} = +204.420000000_8 \\ \text{ACC} = -204.100000000_8 \quad \text{MQ} = +000.200000000_8 \end{array}$$

From this example I think you can see that mathematically we should subtract the ^FACC from the ^FSR and come up with a + answer. However, to be exactly correct we should also subtract the ^FMQ from the ^FSR. After all, the ^FMQ is a part of the ^FACC.

To do this correctly, we should "borrow one" from the ^FSR to subtract the ^FMQ from. This "longhand method" is not easily done in the machine so we take care of it in the following manner.

We subtract the ^FACC from the ^FSR. Now the resultant answer is one too high (we did not borrow one from the ^FSR first). To compensate for this we take the 2's comp of the ^FMQ. This gives exactly the same results as if we subtracted the ^FMQ along with the ^FACC in the adders (which we cannot do without many additional adder ckts).

Here is our same example showing what the machine does (only the fraction parts are shown for clarity).

$$\begin{array}{r} \text{F}_{\text{SR}} = + \quad .420000000_8 \\ \text{F}_{\text{ACC}} = - \quad .100000000_8 \\ \text{Subtract } \text{F}_{\text{ACC}} + \quad .320000000_8 \\ \hline \text{Sub "ONE" from} \quad 1 \\ \text{the ANS IN ACC} = \quad .317777777_8 \\ \hline \text{Now get 2's comp} \\ \text{of } \text{F}_{\text{MQ}} = \text{F}_{\text{ACC}} = \quad .317777777_8 \end{array} \quad \begin{array}{l} \text{F}_{\text{MQ}} = .200000000 \\ \text{F}_{\text{MQ}} = .600000000_8 \end{array}$$

Now the longhand way just to prove this correct--in case we have any skeptics (which I'm sure we don't).

$$\begin{aligned}
 F_{SR} &= + .420000000_8 \quad (\text{start by "borrowing 1" from .42}) \\
 F_{ACC} &= - .100000000_8 \quad F_{MQ} = 200000000_8
 \end{aligned}$$

$$\begin{aligned}
 \text{To Sub } F_{ACC} + \\
 F_{MQ} \text{ from } F_{SR} &= .317,777,777_8 \quad 600000000_8
 \end{aligned}$$

Let's now use the same example numbers to go through the flow chart for the second time. They are sufficient to give all three of the conditions I listed earlier.

The numbers are here again for easier reference.

$$\begin{aligned}
 (\text{CLA}) \quad ACC &= -202.400000001 \\
 (\text{FAD}) \quad SR &= +204.420000000
 \end{aligned}$$

We will start with L time, 1st step, on page 72, figure 5.3-23A. Review the objectives for 1st step and you will see there is little to do at this time using these examples.

The test to see which number has the smaller char will cause us to exit from the "yes" side of the Q carry decision block. The smaller is already in the ACC so we do not have to exchange them.

Next, we compute the char diff and test to see if it's greater than 77_8 . We can readily see that it isn't so we will take the "NO" leg from the "ADD 1 or 2 = 1" decision block. This results in our putting _____₈ in the shift counter.

We will breeze through 2nd step L time because all we do is shift the F_{ACC} and MQ right _____₈ places. This is to align the fractions so we can add time.

We are now about to enter the more interesting part--3rd step L time. Immediately we test the F_{MQ} for zero. This time the $F_{MQ} \neq 0$ so we will leave T2 tgr ON (it was turned on in 1st step unconditionally). This trigger should remind us sometime later that, should we have to subtract the F_{ACC} from the F_{SR} , not to forget the little old F_{MQ} .

Of course, the very next decision block asks if signs are alike or not (and they are not), which looks like we might just qualify for 7090 DP handling. We proceed merrily on our way to Fig. 5.3-23C on the next page.

Here the job is to subtract the smaller fraction from the larger and to do this properly we must know which fraction is smaller (we always comp the F_{ACC} ; if it was the larger, the answer would be in comp form. When the F_{ACC} is smaller, the answer is in true form but "one" too low). Let's take a closer look at the flow chart.

The SR is gated to the ADD with the Comp of the F_{ACC} along with it. We then check for a "col. 9 carry". Our example will give us one and this carry means the F_{ACC} was the smaller. This carry turns on the "9 carry trigger" and then goes to the "T2 trigger" decision block.

At the same time we must proceed down the lefthand side of the flow chart. The ADD is gated to the ACC at L5D1, and we test the Col. 9 carry tgr which was just turned "on". This tgr is saying the F_{ACC} was smaller (and it was the number complemented) so the answer is in true form (but "one" low because the "carry" was not added in). However, all we do for the time being is set the sign of the F_{ACC} equal to the sign of the F_{SR} , the larger, and continue.

Now back at the "T2 tgr" decision block we can finally say that "7090 DP" handling is necessary. So we want to "subtract 1" from the answer in the ACC and get the 2's comp of the F_{MQ} . However, since the answer (fraction) in the ACC is one to low anyway, subtracting the "1" from the F_{ACC} is not necessary. (Please note that the carry would be added in if T2 was off, $MQ=0$, however.)

The "ON" side of the T2 decision block leads to a group of 5 blocks. These jockey the MQ and ACC around so we can get the 2's comp of the MQ fraction. The "F. P. tgr" is turned on here also, to remind us what must be done when we get to 4th step L time (where we actually get the 2's comp of the MQ fraction and return it to the MQ). So let's proceed to the top of Page 75.

Here, in our previous example, we zero tested the F_{ACC} . This time, however, the F_{MQ} is in the ACC because of "7090 DP" conditions. This time these same two blocks which zero tested the F_{ACC} are going to provide the 2's comp of the F_{MQ} (by virtue of the fact it's now in ACC-9-35).

We continue down from these blocks to the "FP tgr" decision block. This tgr now tells us we started 7090 DP back in 3rd step so the output of the adders (right now) is the 2's comp of the F_{MQ} . The "ON" side says take it to the ACC, swap the SR and ACC, and then put the SR into the MQ. The F_{MQ} now equals the 2's comp of the original number and our "7090 D. P. handling" is completed.

At the bottom of Page 75, this time, we will exist from the "ACC-9=1" decision block on the "0" condition (our F_{ACC} is $.317777777_8$, remember). This means we will have to normalize our answer and must go to 5th step L time to do so. On to the top of Page 76!

The logic of normalizing is to shift the ACC & MQ fractions left until a bit appears in ACC pos. 9. For each place we shift the fraction, we must reduce the C_{ACC} . The shifting I'm sure you can easily follow in the first part of 5th step L time. Each time we shift left we step the SC and this procedure continues until $ACC-9=1$.

Once $ACC-9=1$ we turn on the FP tgr, which will then allow the "AD(Q-8) to ACC" block. The logic here is this: Since nothing was put in the SC, it started stepping DOWN from zero. This gives us the number of shifts necessary to normalize in 2's comp form. Therefore, to subtract the number of places it took to normalize,

from the C_{ACC} , we have merely to gate the SC and C_{ACC} into the ADD₈ (which is done all during 5th step). Then take the ADD output back as the new C_{ACC} after we stop shifting and stop stepping SC. The "ones - AD (Q, P, 1, 2)" block is just to maintain the 2's comp of the number of shifts in the ADD (the SC has only 8 positions).

The remaining part of this example is identical to the first. It would be to your benefit to continue through, however, and get the proper MQ character.

THIRD EXAMPLE

For this last example we will confine ourselves to a small area as the flow chart Fig. 5.3-23B, bottom of Page 73. The numbers we will use for this example are:

CLA	X	$L_X = +207.400000000_8$
FAD	Y	$L_Y = +207.400000000_8$

You will note the equal characters and like signs. This means no alignment of the fractions is necessary before adding them, and it will be true addition.

Let's add these two fractions ourselves to better see the problem prior to looking at the flow chart.

$$\begin{array}{r}
 .400000000_8 \\
 .400000000_8 \\
 \hline
 1.000000000_8
 \end{array}$$

carry out of position 9

Naturally the 7090 cannot represent this fraction as shown. It must recognize this "fraction overflow" condition and shift it one place to the right. If it does this, though, it had better increase the C_{ACC} by 1 to show the true position of the binary point. This is the purpose of the "Col. 9 carry tgr" decision block at the bottom right on page 73.

Coming down from this block (the ON side) the carry out of Col. 9 is allowed to carry in to ADD 8. This is allowing our "fraction overflow" to increase the C_{ACC} by 1 automatically (the char is in process of going from SR-ADD-ACC).

Next the F_{ACC} and MQ are shifted right one place and we force a "hot one" into ACC 9 to make up for the carry we let keep right on going to increase the C_{ACC} .

This concludes the text on FAD/FSB/UFA/UFS/FAM/UAM/FSM/USM. That's quite a list since all I've talked about is FAD. However, the differences are so minor that reading pages 78 and 79 (same manual) should answer any questions

you may have concerning their operation.

Review the terminal objectives for FAD; and, once satisfied you can perform them, continue on to FMP.

REVIEW QUESTIONS

FAD/FSB

1. How does the execution of FSB differ from that of FAD?
2. Why is "equalizing" the char. necessary before adding the fractions?
3. Will the fractions ever be comp added? If so, when and why?
4. Is normalizing always necessary on FAD? Why?
5. What is meant by "7090 double precision handling"?
6. What conditions cause "7090 DP" handling?
7. Can the MQ overflow on FAD? How?
8. Give examples which could cause the following on FAD.
 - a. ACC ovfl only
 - b. ACC ovfl and MQ ovfl
 - c. MQ underflow only

ANSWERS

FAD REVIEW QUESTIONS

1. Sign of fraction inverted
2. To "line up" the true binary point of the fractions
3. Yes, unlike signs, subtract smaller fraction
4. No, Addition of fractions may result in ACC 9 = 1
5. Getting the 2's comp of the contents of the MQ when it also must be subtracted from the ^FSR.
6.
 - a. Unlike signs
 - b. ACC fraction smaller
 - c. $^F MQ \neq 0$
7. No
8.
 - a. $SR + 377.4_8$
 $ACC + 377.4_8$
 - b. Impossible
 - c. $SR \ 023.5_8$
 $ACC \ 023.5_8$

07.03.02 FLOATING MULTIPLY

FMP (Floating Multiply) Overall logic

Multiplication of the two fractions is identical to MPY. The exception is that the size of the fractions is 27 bits as opposed to 35 in fixed point. To accomplish this, 27_{10} is placed in the shift counter during FMP instead of the 35_{10} as in MPY (33_8 and 43_8).

The characteristics are added and the "mid point value" is subtracted from the total. By "mid point value" I mean the 200_8 which we use as a reference to designate a positive or negative exponent. For example, if you were to add 212_8 to 207_8 (positive characteristics) the total would be 421_8 . Now we must subtract 200_8 to arrive at the correct characteristic of 221_8 . One further example:

Let's assume we had one characteristic equal to 203_8 and another characteristic equal to 176_8 . The values of these characteristics, exponentially, as you can see are $+3$ and -2 . If we were to add them, our total characteristic would be 401_8 . Then we subtract the 200_8 and we arrive at the characteristic of 201_8 or the equivalent of $+1$. This addition of the characteristics and the subtraction of the 200_8 is done during the multiplication of the fraction.

The product of multiplying our two fractions will be in the ACC bit positions 9-35 and in the MQ bit positions 9-35. In other words, two 27-bit numbers multiplied together will have a 54 bit answer. The most significant part of our product is in the ACC and receives its characteristic after they have been added. The fraction part of the MQ is removed from the FACC 27 $_{10}$ binary places. Therefore, the char. given to the FMQ is that of the ACC minus 27_{10} .

FLOW DIAGRAM TEXT

Floating Multiply

For our trip through the flow chart, a good understanding of what has to be done to multiply 2 floating point numbers is assumed; such as, adding of the char and multiplying of fractions, etc. The numbers we will use for this example are:

$$\text{LDQ} \quad \text{X} \quad L_X = 207.500000000_8$$

$$\text{FMP} \quad \text{Y} \quad L_Y = 175.400000000_8$$

Starting at the top of the flow chart, on page 80, figure 5.3-24A, carefully read the objectives. The objectives are areas I will specifically point out in the flow chart along with other areas which could be misleading or require comment. We will not discuss every block as many are self explanatory.

The first part of E time is used to zero test the multiplier (MQ) fraction. This is done by gating the MQ9-35 to SR and testing the SR input for no bits. (Check out this ckt in systems; you'll find it interesting.) If the $^F MQ$ is = 0, we will turn on the FP tgr. to remember this fact for later use. Our example numbers with the $^F MQ \neq 0$ would result in the "FP tgr" being left OFF.

Next, taking the right path, we find the SC set to 33_8 (27_{10}). This is the number of shifts needed to MPY the 27_{10} bit fractions. Immediately after we gate our multiplicand into the SR (E7D1) and gate ADD (Q=35) to ACC at E10D1. This now makes the $^F ACC = 0$ (so it can be used to develop our product) and at the same time puts the $^C SR$ into 1-8 of the ACC (SR 1-8 to ADD all during E time).

We must return to check the left path before proceeding to 1st step L time. The first decision block, "stg 1-35=0", is zero testing both the char and fraction of the multiplicand as it is brought into the SR. If both are zero, it is a "normal" F.P. zero. Our example will exit us from the "No" side of this block to check the "FP tgr". It is "OFF" because our $^F MQ$ was not = 0, and we can now go to L time, 1st step.

L time, 1st step, has one major objective and that is to ADD the characteristics of our multiplier and multiplicand. At the present time our multiplicand char is in the ACC (1-8). The multiplier char is in the MQ (1-8). The steps which are taken to perform this addition of characteristics and subtract the "200₈ midpoint" can be listed just as shown on the flow chart.

- a. Subtract 200_8 from the $^C ACC$ (instead of adding and then subtracting from the sum)
- b. Take diff ($^C ACC - 200_8$) back to ACC 1-8
- c. Zero out the SR characteristic positions so that MQ positions 1-8 will be cleared when we exchange $^C MQ$ with $^C SR$.
(The MQ will get a new char. upon completion of the MPY operation.)

- d. Exchange C_{SR} and C_{MQ} . C_{MQ} now is in position to be added with C_{ACC} .
- e. Add C_{ACC} and C_{SR} .
(Note:)
(1) The C_{ACC} = original multiplicand char - 200_8
(2) The C_{SR} = original multiplier (MQ) char.
- f. Put the sum back in ACC which now gives us the correct char for our product.

The above steps having been completed we can now proceed to L time, 2nd step, and take care of the fractions.

The first thing in 2nd step is to turn on "T2 tgr." This will indicate at a later time that multiplying of the fractions actually did take place. From this point you will find multiplying the 27 bit fractions almost identical to fixed point multiply. Let's quickly go through the chart, however, on page 82. (Fig. 5.3-24C)

With the $SC \neq 0$, we begin looking for bits in our multiplier, starting with MQ35, before doing any shifting. Our example causes us to exit from the "NO" side of the "MQ(35) presensed" decision block. Since our multiplier has no bit, we need not add the multiplicand into the ACC--just shift the F_{ACC} and F_{MQ} right every clock pulse and keep on looking by pre-sensing MQ pos. 35.

This continues until we finally encounter a bit going from MQ34 to 35 in our multiplier. This condition gives us the "YES" side of the "MQ(35) presensed" decision block, which causes us to stop shifting long enough to gate the "ADD 9-35 to F_{ACC} ". Then we resume shifting, pre-sensing MQ35, and stepping the SC.

The only difference here from fixed point is the fact we are now dealing with 27_{10} bit numbers instead of 35_{10} . This is the reason for using the "9 OV tgr." This will remember any carry we may get when adding the F_{SR} to the F_{ACC} (adding our partial products) when we find a bit in our multiplier. The fact that it comes on tells the machine to put a bit in ACC pos. 9 as we shift right. These "carries" represent the most significant digit of our partial products so we cannot ignore them.

The completion of multiplying the fraction is signalled by the SC going to zero, which turns on the "FP tgr", (top of page 83, figure 5.3 - 24D). Now we must normalize our product if ACC 9 does not already contain a bit. Since this is a normalized instruction, AND USE OF NORMAL NUMBERS IS EXPECTED, it will never be necessary to normalize more than one position. (ACC 9 or 10 will always contain a bit if NORMALIZED numbers were used).

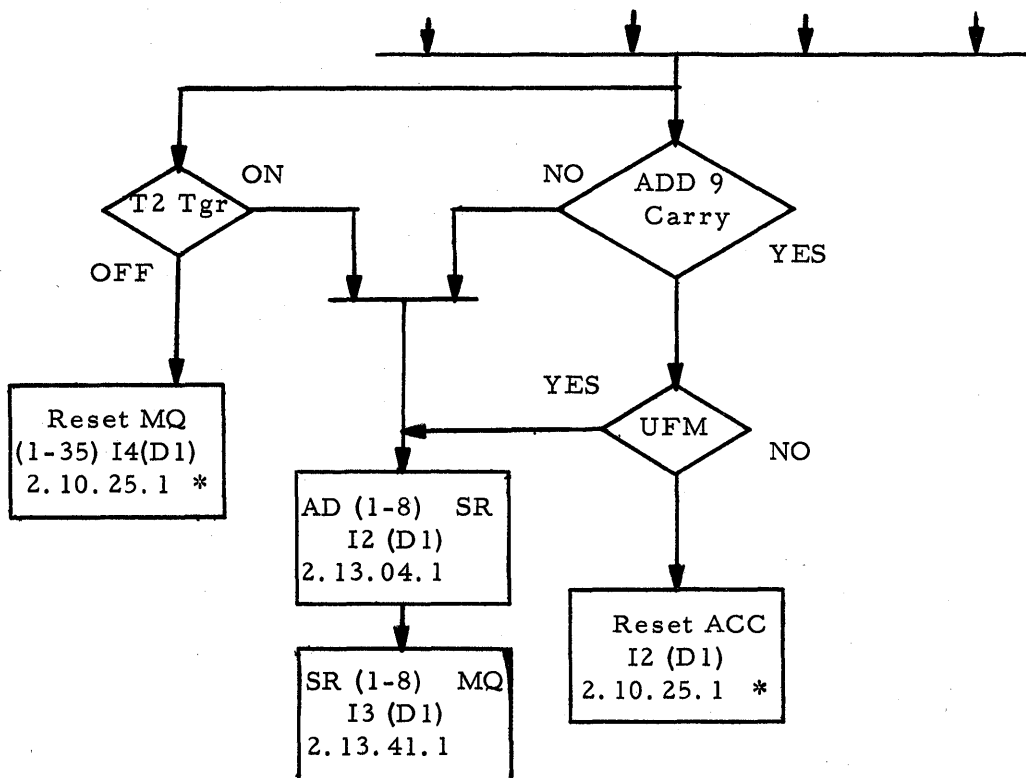
The logic for normalizing is the same as previous instructions. Shift the fractions left and subtract the number of shifts taken from the characteristic. You will note that the blocks in the flow chart which do the normalizing on FMP (between the $AC(9) = 1$ decision block and the "END OF" block) all contain an asterisk (*). At the bottom of the page the note indicates the logic of these blocks applies only to FMP--not UFM.

UFM being "unnormalized" floating multiply, it sounds reasonable that normalizing should not be attempted.

After normalizing (which would not be necessary for our example--right?), we end up and go to I time. There is little left to do here except set the signs of our product (ACC & MQ) and give the $^F MQ$ the proper characteristic. The signs should be self explanatory so let's concentrate on the $^C MQ$ being generated and the conditions which are needed to put it with the $^F MQ$.

We do agree that the number in the MQ is merely an extension of the ACC (?). Therefore, the F_{MQ} is 27_{10} binary places "further removed" from the binary point referred to by the C_{ACC} . If this is so, then to give the F_{MQ} its own characteristic, we have only to take the C_{ACC} and subtract 27_{10} (33_8). This is done on the flow chart directly under "I time next instr"--the two extreme right blocks. Whether or not we actually take the output of the ADD to the MQ, however, depends on several tests; one of which is being made in the ADD at the same time we are subtracting 27_{10} from the C_{ACC} .

Two blocks (just left of those used above) are "zero testing" the F_{ACC} . This is done by "comp AC9-35 to the ADD" and putting a "hot one" in ADD 35. An "ADD 9" carry at this point will indicate the $F_{ACC} = 0$. Before continuing on from this point, correct the flow chart (Fig. 5.3-24D, bottom left) as follows:



Getting back to our example numbers we can see we should not have an ADD 9 carry. This means, if T2 is ON (i. e. we did multiply), that we should give the MQ its proper characteristic. This would complete the logic of the example numbers we used. However, let's look further at other paths through the above logic.

If we had gotten an "ADD 9 carry" it means our $F_{ACC} = 0$. This could only happen on normalized FMP if the multiplier fraction had been zero OR the multiplicand a normal zero. In either case, T2 would have been off (i. e. we wouldn't have multiplied) so both the ACC & MQ would be reset. This gives us the CORRECT answer; a NORMALIZED ZERO.

One last condition is the $F_{ACC} = 0$ on UFM. In this case we give the proper characteristic to the MQ regardless; the reason being that, with unnormal numbers to start, all bits in our product may well be in just the F_{MQ} .

SUMMARY OF FMP

The basic concepts of multiplying two normalized floating point numbers are:

1. Add the characteristics
2. Multiply the fractions
3. Normalize one place if necessary
4. Compute proper characteristic for the F_{MQ}
5. Set the sign of the product (ACC & MQ)

These concepts, I'm sure, have been repeated several times previously. However, the machine does just these things and in this sequence. To review each briefly:

1. Adding of the characteristics takes place during 1st step L time. The "200₈ midpoint" is subtracted from the C_{ACC} (multiplicand) and the diff is then added to the C_{MQ} . This is then put in the ACC as the char for our product.
2. The multiplying of the fraction is done by looking at our multiplier (F_{MQ}) one bit at a time and adding in the multiplicand (F_{SR}) once for each bit found. The F_{MQ} is scanned by looking at MQ for 35 as we shift the MQ and F_{ACC} (partial product) to the right.
3. To normalize we check "pos 9" of the ACC after multiplying the fractions. If it is zero, the F_{ACC} & MQ are shifted left one and "one" is subtracted from the C_{ACC} . All this was done in L time, 2nd step.
4. The C_{ACC} is then used, during "I time next", to compute the proper Char for the F_{MQ} . This is done by subtracting the 33₈ (27₁₀) from it. This same procedure was used in FAD to get a char for the FMQ.
5. The signs of the product are also being set as we compute the MQ char. Like signs result in a + product; unlike signs give a - product.

REVIEW QUESTIONS

FLOATING MULTIPLY

1. How are the char's handled on FMP? The fractions?
2. Can ACC overflow occur on FMP? How?
3. What does "normalizing" consist of on FMP? How does this differ from FAD?
4. Why is 200_8 subtracted from the multiplicand char during L time, 1st step?
5. What function does the "FP tgr" serve on FMP?
6. Why does the $^F MQ$ get a char which is 27_{10} less than the char for the $^F ACC$?
7. If the $^F MQ$ is zero to begin with, the $^F MP$ will end op in E time. (T or F)

ANSWERS

FMP REVIEW QUESTIONS

1. They are ADDED, Multiplied
2. Yes--sum of char greater than 177_8 (with 200_8 midpt 377_8)
3. Checking ACC-9 of product and if = 0 shift one place and subtract one from ^CACC. FAD can normalize more than one place if necessary.
4. In preparation for ADDING char so only one 200_8 midpoint will be in our total.
5. Used to indicate if ^FMQ = 0
6. The ^FMQ is an extension of the ^FACC and is 27_{10} places further removed from the reference binary point just left of ACC 9.
7. True

07.03.3 FLOATING DIVIDE

FDH (Floating Divide) Overall logic

The logic of FDH is similar to that of DVH/DVP in that the fractions are divided in the same manner; the major difference again being that of the characteristic handling. The characteristics in divide have to be subtracted. The characteristic of the divisor (which is the SR) must be subtracted from the characteristic of the dividend (which is in the ACC). After subtracting the divisor characteristic from the dividend characteristic, now we must add the "mid point" back in to arrive at the correct characteristic.

For example, if our divisor characteristic is 201_8 and our dividend characteristic is 205_8 and we subtract them, the resultant characteristic will be 004_8 . This should, however, represent a characteristic of +4 (not an extremely negative characteristic). Therefore, we add in the " 200_8 mid point" and we now have a characteristic of 204_8 which truly represents our +4 exponent.

Our quotient will be in the MQ and it will receive a characteristic equivalent to the divisor characteristic subtracted from the dividend characteristic. In the ACC will be our remainder. The remainder will receive a characteristic equal to the dividend characteristic (that is the original dividend characteristic) minus 27_{10} . If you recall, the number of bit positions in the fraction is 27_{10} . Therefore, we are going to divide into the fraction 27_{10} places and this value must be subtracted from the original dividend characteristic to arrive at a characteristic for our remainder.

Another point of logic concerning division of the fraction merits mention before we go to the flow diagram. It is the "divide check test" logic for Floating Point divide. If you recall fixed point divide, the divisor (SR) could not be smaller or equal to the ACC on our initial attempt at dividing. This would have resulted in a Divide Check because the MQ cannot handle a 36bit quotient.

The same logic applies to the $^F SR$ which will be divided into the $^F ACC$. If the $^F SR \leq ^F ACC$, the fraction of our quotient (MQ) would be 28_{10} positions instead of 27_{10} which the machine can handle.

This possibility has been eliminated by machine circuitry to the following extent. On normal floating divide, if normalized numbers are used, the 7090 will divide and not give a divide check. Let me try to clarify this further by the following example:

Suppose we were to divide: SR = 204.430 000 000₈
 ACC = 206.630 000 000₈

As you can see, both are normalized FP numbers and the $^F SR$ is less than the $^F ACC$.

The machine tests for this condition and if it exists the 7090 merely re-scales the dividend (logically) like so:

$$\text{SR} = 204.430\,000\,000_8$$

$$\text{ACC} = 207.314\,000\,000_8$$

The ^CACC is "upped one" and the ^FACC shifted one place to the right. The NUMBER IN THE ACC IS MATHEMATICALLY THE SAME! But the ^FSR is now larger than the ^FACC ! What I have said here is what LOGICALLY takes place; it is ACTUALLY achieved in the following manner.

This condition when recognized, is accomplished by upping the ^CACC , stepping the SC, and NOT SHIFTING LEFT. (Normally on divide you would shift left then divide). If this is confusing perhaps the flow chart will help clear up this area.

One additional point; if normalized numbers are used it is IMPOSSIBLE for the ^FSR to be TWICE as small as the ^FACC . Therefore, it will never go into the ^FACC more than one time, initially. This is all the machine will take care of. For this reason, this condition is referred to many times as "quotient greater than one" (quo > 1). This is another method of looking at the same condition but perhaps not so easily understood.

FLOW DIAGRAM TEXT

Floating Divide

Starting at the top left on page 87, figure 5.3-26A, let's first go through the flow diagram without specific numbers. As different situations arise, I will give two numbers, divisor and dividend, which would utilize specific ckts.

After carefully reading the objectives of the beginning, the chart is self explanatory to the point of gating "SB to SR at E7(D1)". Immediately after are four blocks concerned with objective "3" above (preparation for the divide check test).

It is OK for the F_{SR} to be smaller than the F_{ACC} , but it cannot be twice as small. You are right--with normalized numbers this cannot occur; but whose to say all programmers follow the rules? So the comparison is to see if the F_{SR} is twice as small. To make this comparison easier the F_{SR} is going to be compared against ONE-HALF ($1/2$) the F_{ACC} .

If the F_{SR} is less than, or equal to, $1/2$ the F_{ACC} ; it is logically the same as saying the F_{SR} is twice as small as the F_{ACC} . Saying it another way, to add confusion; if $F_{SR} \leq 1/2 F_{ACC}$ then the $F_{ACC} \geq 2(F_{SR})$. Enough said; shifting the F_{ACC} right one place is halving the F_{ACC} in preparation for this test. Next, we proceed to L time, 1st step.

Again, carefully read the objectives. Objective "1" will be taken care of right away. By complementing the F_{ACC} into the ADD, with the F_{SR} , an "ADD 9 carry" tells us the F_{ACC} is smaller (remember, the F_{ACC} is $1/2$ its real value). This is the condition we must have to avoid getting a divide check. So the "FP Tgr" is turned "on" to say all is well and the division can now be done.

Continuing to the top right side of the flow chart now we set the sign of our quotient. If signs are alike we "seemingly" do nothing; but this means a plus answer and S position was reset during E time. After this, the F_{ACC} is "restored" by shifting left one so we can get on with the division.

Next on the flow chart the entire ACC gets complemented to the ADD with just the char. part of the SR. A "hot one" is also forced into ADD 35. Two things are taking place at once. The F_{ACC} is being "zero tested" and the C_{SR} is being subtracted from the C_{ACC} ; PARTIALLY. The subtract is not complete at this time, as you will see if you perform these steps with the following characteristics assumed.

$$C_{SR} = 203_8 \text{ (divisor)}$$

$$C_{ACC} = 207_8 \text{ (dividend)}$$

The FINAL char. answer should be 204_8 ; but at this point in time the "partial char. diff" is bits in Q and P with a char of 373_8 (this is really the char diff in comp form because we complemented the larger). However, as you can see, this partial diff is put in the ACC if the F. P. tgr in "ON" (i. e. this is a legitimate divide).

Meanwhile, the results of our "zero test" on the fraction part of the ACC would have been completed. The "col 9 carry" tgr would have been turned ON if the $F_{ACC} = 0$. This test is made to save time. If the dividend is zero; then, of course, the quotient will be zero so there is no need to take further L cycles. Let's assume an ACC fraction $\neq 0$ so we can do the divide process.

Proceeding on in the flow chart to the top of page 88, figure 5.3-26B, we can see the decision being made from our two previous tests.

1. Col 9 carry tgr off - means $F_{ACC} \neq 0$
2. F. P. tgr ON - means divide OK, F_{SR} is not twice as small as F_{ACC}

L time, 2nd step, is used to check for the condition referred to as "quotient > 1 ". If you recall our discussion, prior to this flow diagram text, concerned with the situation where the F_{SR} is smaller than the F_{ACC} (NOT twice as small, but smaller) this would mean that the first bit would be placed in our quotient before the dividend was shifted; and, after 27_{10} shifts, would actually represent a "quotient > 1 ".

An example, in the decimal system, would be dividing .5 by .3 (the answer is 1.66, etc.). However, the machine cannot represent improper fractions in F.P. format. So, we are going to "re-scale" the dividend by shifting and changing the char. Using our decimal numbers again, it would be like changing the .5 to .05 and then dividing. The answer is now .166 etc. The SAME answer if the char tells where the point really belongs.

Back at the flow chart, in 2nd step, the entire ACC is complemented; the fraction part so we can divide (same as fixed point divide) and the char part so we have the "char diff" now in true form.

After this the ACC and F_{SR} are gated to the ADD to see if the $F_{SR} \leq F_{ACC}$. If it is, then the dividend must be re-scaled. A "NO ADD 9 carry" means the re-scaling is necessary. Assuming the following F. P. numbers for L time, 2nd step, the ACC and SR would look like so:

Original Contents	SR	=	203.40000000 ₈
	ACC	=	207.60000000 ₈
	SC	=	33 ₈ (27_{10})
Just Prior to Re-scaling at L4 Time - 2nd Step			
	SR	=	203.40000000 ₈
	SCC	=	004.17777777 ₈
	SC	=	33 ₈ (27_{10})

Just After Re-Scaling (because of NO add 9 carry)

$$SR = 203.400000000_8$$

$$ACC = 005.177777777_8$$

$$SC = 32_8 (26_{10})$$

The C_{ACC} was increased by 1 and the F_{ACC} was effectively shifted right once because the SC was stepped and we did not shift left. We are going to be able to make a reduction of the dividend before it is shifted left--this is a special case and was not possible on the fixed point divide.

While still in this area, please take note that if the 9 carry had occurred (the $F_{SR} > F_{ACC}$), the C_{ACC} would not have been increased and the F_{ACC} would have shifted left as the SC was stepped.

We are now ready to proceed to page 89, figure 5.3-26C for 3rd step L time. Division of the fractions I also would like to discuss briefly before going further in the flow chart.

The F_{ACC} is in complement form throughout the L time 3rd step. This enables us to get the difference ($F_{ACC} - F_{SR}$) whenever we find the $F_{ACC} > F_{SR}$. It also must allow us to know when the $F_{ACC} > F_{SR}$ so we can put the diff back in the ACC. To make this comparison, we really need an "extra" ACC position. ACC 8 would be fine, but is used for char. handling and is not available.

For this reason the "9 OV Tgr" is used to "simulate" the ACC 8 position for handling the fraction. Therefore, by looking at the "9 overflow tgr ANDED with an ADD 9 carry, we can SIMULATE AN ADD 8 CARRY cond. This "simulated ADD 8 carry" will be used to tell us when the F_{ACC} is smaller than the F_{SR} .

Let's take an example to show this condition before going back to the flow chart.

$$F_{SR} = .400000000_8$$

$$F_{ACC} = .677777777_8 \text{ (remember--it's complemented)}$$

As we shift the F_{ACC} left, any bits shifted from ACC-9 turn on the "9OV Tgr" (just as if it were ACC-8). This is so we can remember what the high order bit of our dividend is.

After Shifting:

$$F_{SR} = .400000000_8$$

$$F_{ACC} = .577777777_8 \text{ (hot "one" to ACC-35, when shifting left, to maintain complement)}$$

and 9OV tgr "ON"

Now gate the two fractions into the ADD and add

```

                .4000000008
                .5777777778
output of adders = 1.1777777778

                ADD 9 carry
    
```

Had this ADD 9 carry been allowed to go into ADD 8 and had we been able to use the true ACC-8 pos to gate into ADD8, we would have gotten an ADD-8 carry. Hence, the name for this condition is "simulated ADD-8 carry", and it means the $F_{SR} > F_{ACC}$. I strongly suggest that, at this point, you take several example fractions for the SR and ACC and prove the validity of this logic. If it is clearly understood, it will make division of the fractions quite simple.

Returning to the flow chart you will see the F_{SR} and the ACC are gated to the ADD all during 3rd step. Therefore, when we decide the $F_{SR} \leq F_{ACC}$, we have only to take the diff (output of the adders) back to the ACC.

Since we have already shifted left (if the quotient < 1) or re-scaled the dividend (if the quotient > 1), we test immediately to see if the divisor fraction will go into the dividend fraction. The decision block is at the top right on page 89. It is checking for a simulated ADD-8 carry by looking at the 9 OV Tgr and the ADD 9 carry output.

At this time if the true value of the F_{ACC} is $> F_{SR}$, we will leave the "NO" side of this decision block. Here are 2 sets of example numbers which you can use to prove this point.

```

(1)  SR    = 201,500000008
      ACC   = 203.700000008

(2)  SR    = 204.600000008
      ACC   = 207.600000008
    
```

NOTE: Don't forget prior manipulation;
such as the F_{ACC} being complemented
and possibly shifted.

However, if the $F_{ACC} < F_{SR}$ at this point, it will result in our taking the YES path from this block. If doing the above examples did not also help show this point, here are two examples which might.

```

(1)  SR    = 202.600000008
      ACC   = 203.200000008

(2)  SR    = 204.500000008
      ACC   = 206.100000008
            152
    
```

NOTE: You will notice I had to resort to use of unnormalized F. P. numbers to achieve this result. This will also help prove another point later (I hope!).

As you further see, by looking at the flow chart, this decision is made to determine if the fractiondiff should be taken back to the ACC and a bit placed in our quotient. Then, if the FP tgr is off, (back at L2, 2nd step), we proceed to shift the F ACC left, pump a "hot one" into ACC 35 and step the SC. Now we are ready to check the F SR against the F ACC once more. (Note that the 9 OV tgr. will be turned ON if ACC-9=1 when we shift left, but OFF if ACC-9=0).

If the SC \neq 0, we will continue this "loop" until it is zero. This tells us when we have attempted to divide the F SR into all 27_{10} (33_8) places of the dividend. Once the SC goes to zero, we will turn ON the F. P. tgr and make ONE LAST attempt of dividing the F SR into the F ACC.

After this attempt, the FP tgr will be "ON" and cause us to proceed to the left and down the flow chart where an AND condition is needed for us to "End OP". Let's return to the top left of this chart and see what these six blocks were doing while we were busy dividing the fractions.

The first block, "1 to AD(1) at A4 (D3)", is adding back in the 200_8 midpoint which got subtracted out of our char at the beginning (Ltime, 1st step). You will recall that our "char diff" is in the ACC 1-8. In the case of characteristics such as:

$$SR = 203.50000000_8$$

$$ACC = 207.40000000_8$$

the difference of 004_8 would now be added to 200_8 and our correct quotient char. of 204_8 would result. However, let's assume the following characteristics (same fraction).

$$^C_{SR} = 207_8$$

$$^C_{ACC} = 203_8$$

Now the difference is a -4; not a +4 as in the previous example. The resultant quotient char. should be 174_8 . Let's go through the same process and see how we make out, step by step.

- (1) First, the $^C_{ACC}$ was complemented (along with bits Q and P) and added to the $^C_{SR}$. Result would be Q=0, P=0, 1 to 8= 003_8 .
- (2) This time our answer char diff came out in TRUE form, but one short. However, the very first part of L time, 2nd step, this char diff gets complemented along with the $^F_{ACC}$. Result: Q=1, P=1, ACC 1-8= 374_8
- (3) Now let's add to it the 200_8 at L4 time in 3rd step and see what happens!

$F_{ACC} =$	Q	P	1	-----	8
	1	1	1	1, 111,	100
Forced bit in ADD (1)				1	
ADD output	0	0	0	1, 111,	100
Result in octal				1	7 4 ₈

Very good! The machine works.

The logic here is really not too bad. It is simply the fact that if the larger char were complemented then the diff came out in true form originally, but one short. This is just perfect because the "diff" represents a "NEGATIVE diff" and must get subtracted from 200_8 --NOT ADDED. Therefore, to get the 2's comp of the true diff (so we can subtract it) all that is needed is to take the 1's comp of the computed diff which came out one to small (as it always will).

Agreed, this logic of why it always works may be confusing; but take the following six example characteristics and work them through. They should help clear up how the quotient char is generated if not the logic behind it (assume same fractions as previous example).*

<p>(1) $C_{SR} \quad 170_8$ $C_{ACC} \quad 172_8$</p> <p>(2) $C_{SR} \quad 204_8$ $C_{ACC} \quad 175_8$</p> <p>(3) $C_{SR} \quad 175_8$ $C_{ACC} \quad 204_8$</p>	<p>(4) $C_{SR} \quad 175_8$ $C_{ACC} \quad 170_8$</p> <p>(5) $C_{SR} \quad 050_8$ $C_{ACC} \quad 060_8$</p> <p>(6) $C_{SR} \quad 217_8$ $C_{ACC} \quad 210_8$</p>
---	---

* Suggested at this point that your answers be checked before continuing.

The next five blocks following "1 to AD(1) at A4(D3)" are primarily concerned with two things. Number one is the gating of the quotient char, just generated in the ADD, through the SR to the MQ. Second is saving the original divisor char (SR), while this is being done, so the original dividend char can be computed again.

You may recall the original dividend char (ACC) was lost during 1st step L time when we got our partial char diff. We must "regenerate" it, though, because our REMAINDER CHAR should be the original dividend char-- 27_{10} (33_8).

The gating of the quotient char to the MQ while saving the C_{SR} should be straight forward on the flow chart. The fact that it occurs each L cycle during 3rd step is not going to harm anything. However, the computing of the original dividend char is rather well hidden unless you look closely.

The last block in this group says "SR (1-8) to ADD at A8(D3)". This is gating the original divisor characteristic into the ADD at this time. At the same time the char

position of the ACC is still being gated into the ADDERS and this represents the difference between the divisor char and dividend char. It may represent a negative diff or a positive diff (complement or true form) but it is the diff. When this diff is added (or subtracted effectively, if it's in comp form) to the original divisor char, the result is the original dividend char.

Now, with the FP tgr "ON" we can End Op and gate this original dividend char into the ACC at A10(D1). Proceeding to I time at the top of the next page we have only to compute the char for our remainder (ACC) and restore our remainder fraction to its TRUE form. Proof of these two items will be left to you.

Divide Check Conditions

Let's return to the beginning of the flow chart; and, using the following examples, see how a divide check is handled on DVH.

Example Numbers

SR	203.200 000 000 ₈
ACC	204.600 000 000 ₈

NOTE: In order to get a divide check, the ^FSR must be twice as small as the ^FACC. To get this condition the DIVISOR must be an UNNORMALIZED number. Thus we can say: If normalized numbers are used, a divide check cannot occur on floating point divide.

At the bottom left of figure 5.3-26A the divide check test is made. Our example would cause an "ADD 9 carry" at this point and turn "ON" the following tgrs.

1. T2 Tgr
2. Divide check tgr
3. T1 Tgr

Each of these has a purpose, which I will mention but suggest you check out the logic in the flow chart.

1. Tgr. T2 turning "ON" will be used during "I time next" to tell the machine that the ACC must not be re-complemented. It will be the original dividend and that's the way we want it left.
2. The Divide Check tgr. is a "programmable" tgr, which is used by the DCT (divide check test) instruction. If using a FDP instr., a divide check would not stop the machine. Normally the DCT is used in conjunction with FDP.
3. Last is tgr. "T1", which is going to allow the master stop tgr to be turned "ON" during I time next of an FDH.

I realize we have not taken any one example all the way through the flow chart. However, if the logic covered has sunken in to some degree, it should be worthwhile for you to take through one, or more, of the following examples. The special conditions that each example should cause are given to the right of each example.

	<u>Example Numbers</u>	<u>Special Conditions</u>
(1)	SR = 204.400 000 000 ₈ ACC = 206.400 000 000 ₈	Quo. > 1 Char. diff. positive
(2)	SR = 204.500 000 000 ₈ ACC = 206.400 000 000 ₈	Quo. < 1 Char. diff. positive
(3)	SR = 207.600 000 000 ₈ ACC = 204.400 000 000 ₈	Quo. < 1 - Char diff will represent a negative char for the quo.
(4)	SR = 170.500 000 000 ₈ ACC = 173.500 000 000 ₈	Quo. > 1 - Char diff "positive" even though divisor & dividend char neg.
(5)	SR = 175.300 000 000 ₈ ACC = 204.600 000 000 ₈	Divide check condition ${}^F\text{ACC} > 2({}^F\text{SR})$

FLOATING DIVIDE SUMMARY

We have found that in Floating Divide the logic is much as we had expected. There are a few unusual ways of doing certain things, however, that can be confusing if it is not known "what" the machine is trying to accomplish.

The divide operation starts off by testing for a "divide check" condition. Since a divide check, on floating divide, means the $F_{ACC} \geq 2(F_{SR})$ the F_{ACC} is halved to make the test. This is done by shifting the F_{ACC} right one place. Of course, it is restored by shifting left after the test.

Next the test is made for a possible "quotient > 1 " condition. This occurs when the $F_{SR} < F_{ACC}$ initially. If this cond. does exist, the dividend (ACC) gets re-scaled (effectively). If this is not the case ($F_{SR} > F_{ACC}$), re-scaling is not necessary and the F_{ACC} is shifted left for our first attempt at reduction.

During 3rd step L time the fractions are multiplied. For the 7090 to compare the divisor (F_{SR}) to the dividend (F_{ACC}) and determine if the F_{SR} is smaller, requires an additional register position. However, we found ACC pos. 8 not available for use by the fraction; so we used the "9 OV tgr". With this was "anded" the ADD 9 carry output to "simulate" an extra ADD position. This "simulated ADD - 8 carry" served the same purpose as the "Q carry" on fixed point divide.

This about took care of our fractions. The subtraction of the SR char from the ACC char was also taking place during this time, however; in fact, during 1st step L time we immediately got the "partial diff".

This "partial diff" we found could be in comp form (if the C_{ACC} was the larger); but even this worked out during 3rd step L time--because if this diff was in comp form, it meant a neg char diff and the result was subtracting it from 200_8 .

Lastly, we looked at our divide check possibilities. We found that a divide check basically does these three things:

1. Stops the machine, if the instruction is a DVH, at I5 time.
2. Turns on a programmable tgr (with indicator) which can be tested with the DCT instruction.
3. Turns on one other tgr (T2) which says DON'T re-comp the ACC because it has never been comp!

REVIEW QUESTIONS

FDP/FDH

1. Give the use of each of the following triggers on an FDH instruction, if they turn "ON".
 - a. T1 tgr.
 - b. T2 tgr.
 - c. divide check tgr.
 - d. F.P. tgr.

2. Is a "neg char diff" handled differently than a "positive char diff"? (Example of each following)

<u>Neg Diff</u>		<u>Pos Diff</u>
Char of divisor	207	201
Char of dividend	201	207

3. What is meant by "partial diff" on page 87, Manual of Instruction, lower righthand corner?
4. What is meant by "quotient ≥ 1 " on F.P. divide?
5. How is a "quo ≥ 1 " handled on the 7090?
6. What conditions give a "divide check" on F.P. divide?
7. Why doesn't FDP/FDH attempt to normalize?
8. Why is the "ACC - 9 ovfl tgr" used?
9. What does "simulated ADD 8 carry" mean?
10. Why is the "original dividend" char computed in L time, 3rd step?

ANSWERS

FDP/FDH REVIEW QUESTIONS

1.
 - a. Halt CPU
 - b. Re complementing the ${}^F\text{ACC}$ is not necessary
 - c. Programmable indicator for DCT instruction
 - d. OK to divide

2. NO

3. If the divisor char was smaller, the diff at this point is in "comp" form. If the divisor char was larger, the diff at this point is in true form, but one too small (in either case it's only partially completed).

4. When the ${}^F\text{ACC}$ will go into the ${}^F\text{SR}$ before any shifting has been done.

5. The dividend (${}^C\text{ACC}$ and ${}^F\text{ACC}$) is rescaled effectively by upping the ${}^C\text{ACC}$ and "blocking" the 1st left shift of the ${}^F\text{ACC}$.

6. If the ${}^F\text{ACC} \geq 2({}^F\text{SR})$

7. If normal numbers are used it would never be necessary

8. To give us an extra ACC pos effectively since we cannot use ACC-8

9. The ${}^F\text{SR} > {}^F\text{ACC}$

10. It is needed to generate a char for our remainder (ACC) and was lost earlier. Char for remainder is original dividend char - 27_{10}

07.03.4 Floating Point UNDERFLOW and OVERFLOW

Logic

As you may already know, the floating point underflow/overflow conditions on the 7090 will cause a trap. This is assuming the 7090 is in its normal mode of operation (power just brought up; clear button depressed, etc.). Floating point trap (FPT) mode is considered a "normal mode of operation" for the 7090.

Also, the overflow/underflow is referring to the CHARACTERISTICS of our floating point numbers. Overflow means we have generated a char. too large (or positive) for the 7090 to represent in 8 bit positions (greater than 377_8). Underflow means we have generated a char. too small (or negative) for the machine to handle (less than zero).

I say generated because these conditions are continually tested for throughout execution of our FP instructions. During these instructions the char's get added, subtracted, shifted, etc. Let's look at a few very likely times when we might get underflow/overflow conditions.

FAD

During this instruction the characteristics are made "equal" so we may add the fractions. This would not result in an underflow/overflow cond., but how about when the $^CACC = 377_8$ and we have a "fraction carry"? Let's use the following example.

CLA X $L_X = +377.700\ 000\ 000_8$

FAD Y $L_Y = +377.700\ 000\ 000_8$

I think you can see that adding these fractions will result in a carry. The 7090 takes care of this by shifting the FACC right once, putting a bit in ACC-9, and letting the carry increase the CACC (to take care of shifting FACC right).

Increasing a char of 377_8 by "1" will cause an "ACC OVFL" condition because we cannot represent a char of 400_8 in the 7090. This condition will be recognized as this addition takes place. A bit in ACC-P only (no bit in Q) at this time signals the OVFL condition. (Check FAD flow chart, figure 5.3-23B, page 73 bottom righthand corner).

An underflow condition is also possible on FAD. Suppose we added the following 2 numbers.

CLA X $L_X = +010.000\ 000\ 040_8$

FAD Y $L_Y = +011.000\ 000\ 001_8$

The answer before normalizing would be:

ACC = $011.000\ 000\ 001_8$

Now we must shift the F ACC to the left 22_{10} (26_8) places to normalize. We must also subtract this number of shifts from the 10 (26_8) C ACC to show that this shifting took place. If we subtract 26_8 from the C ACC, which is 11_8 , we will be trying to generate a char more negative than the machine can represent. This condition will be recognized by bits in Q and P, of the ACC, as soon as the subtraction takes place (refer to page 76 figure 5.3-23E, middle right side of the page). This condition would be known as an ACC UNDERFLOW on FAD.

One additional condition possible on FAD is the "MQ UNDERFLOW". The last part of a FAD operation causes 27_{10} (33_8) to be subtracted from the C ACC to put in the MQ for its char. If the C ACC, at this time, is smaller than 33_8 (27_{10}); then we would be trying to generate a char. to neg. for the 7090 to represent. However, the char. generated was meant to be put in the MQ. Therefore, bits in Q and P of the ADDERS at the time we subtract this 27_{10} (33_8) will cause an MQ underflow to be recognized (refer to page 76, figure 5.3-23E, lower left corner).

Please note the diff between using the ACC Q and P positions to check for ACC underflow/overflow and ADDER Q and P to look for MQ underflow/overflow conditions. P only means overflow; Q and P means underflow. If this is not clear, use the previous example char. to prove the conditions mentioned.

Last on FAD is the fact that "two" types of underflow/overflow may occur during the execution of one FAD; such as ACC underflow and MQ underflow at the same time. Combine the logic of the last two examples and I think it will be clear. (Hint: If the F ACC underflows, then what should happen if 27_{10} is subtracted from that??)

We have discussed the 3 which are possible on FAD. For FMP and FDH I will only explain how each is possible and give one set of example numbers for each. The chart below will be used as reference.

Instruction	ACC	MQ	Spill Code Bits				OCT Code
			14	15	16	17	
FRN	Ovfl			*	*		06
FAD/FSB		Unfl				*	01
	Unfl	Unfl			*	*	03
	Ovfl			*	*		06
FMP		Unfl				*	01
	Unfl	Unfl			*	*	03
	Ovfl			*	*		06
	Ovfl	Ovfl		*	*	*	07
FDH/FDP		Unfl	*			*	11
	Unfl		*		*		12
	Unfl	Unfl	*		*	*	13
		Ovfl	*	*		*	15

FMP

The 1st of 4 possibilities is the MQ underflow. This will also be the result of subtracting 27_{10} (33_8) from the C ACC when generating a char for the MQ.

The second (ref to cond shown on chart under FMP) shows both the ACC and MQ underflowed. This would result when adding the char of our multiplier and multiplicand resulted in a char too neg. (less than zero). An example would be:

$$\text{LDQ X } L_X = 065.400\ 000\ 000_8$$

$$\text{FMP Y } L_Y = 040.600\ 000\ 000_8$$

Then, when generating the MQ char, it of course would also underflow.

There is also an ACC overflow possibility. This is the result of adding char that will total above 377_8 . An example would be:

$$\text{LDQ X } L_X = 301.400\ 000\ 000_8$$

$$\text{FMP Y } L_Y = 365.500\ 000\ 000_8$$

The resultant char would try to be 566_8 , which is much too big.

Since 27_{10} (33_8) is subtracted from that (and it would still result in a char above 377_8) the MQ is going to overflow as well. This one example actually shows ACC and MQ overflow conditions together. However, if the C ACC overflowed by 27_{10} (33_8) or less, you can see that only the ACC overflow cond. would exist.

FDH/FDP

FDP is different than the others in many respects (it even has a spill code bit especially for itself - 14). The 1st cond on the chart is an MQ underflow. Since this is the quotient char, this cond can arise when subtracting the divisor (SR) char from the dividend (ACC) char. Example numbers which could cause this cond. are:

$$\text{CLA X } L_X = 150.400\ 000\ 000_8$$

$$\text{FDP Y } L_Y = 374.600\ 000\ 000_8$$

The difference is $> 200_8$ and would represent a negative direction. Thus, an MQ underflow will result.

Since the third possible cond on the chart must have the same condition above to create the MQ unfl. let's take it next. The underflow in the ACC can occur at the same time if the original dividend char is $< 33_8$ (27_{10}). This ACC underflow cond. would occur then when the char for the remainder was being computed. (Original dividend char - 27_{10} (33_8)) Example numbers would be:

$$\text{CLA X } L_X = 031.400\ 000\ 000_8$$

$$\text{FDP Y } L_Y = 322.500\ 000\ 000_8$$

The second condition (ACC underflow only) could occur whenever the original dividend char is less than 33_8 (27_{10}) and the char diff (between C_{SR} and C_{ACC}) is less than 200_8 .

An MQ overflow on FDP can occur when the char diff (between divisor and dividend char) is greater than 200_8 , BUT REPRESENTS A POSITIVE DIFF. An example would be:

$$\text{CLA X } L_X = 370.400\ 000\ 000_8$$

$$\text{FDP Y } L_Y = 150.500\ 000\ 000_8$$

FLOATING POINT

UNDERFLOW/OVERFLOW SUMMARY

The logic of Underflow and Overflow conditions should be understood for each instruction. Since the conditions are recognized by testing specific results, when char's are being computed, it is imperative that the logic of "char handling" in the 7090 be understood. If it is known when char's are added, subtracted, etc., and how the char's get generated for remainders, the MQ, etc., the underflow/overflow possibilities can always be figured out.

As was mentioned at the beginning of this area, floating point trap will occur on any of the unfl/ovfl conditions. This "trap" causes the PC and a "spill code" to get stored at location zero. The spill code bits (14, 15, 16, 17 of the dec) tell what type(s) of underflow/overflow conditions occurred.

It is felt at this time that, if the logic of underflow/overflow conditions is understood, the FPT ckts can be self-analyzed with the aid of the flow chart on page 142. Manual of Instruction #223-6895. The trap ckts vary a little from each other in the 7090. Usually it is just the conditions which cause them and the "trap to" address; which is the case of FPT is 10_8 .

Overall Logic

Since the logic of ANDING, ORING, and even EXCLUSIVE ORING, should be familiar to you; it will not be discussed in great detail. This group of instructions are largely explained just by their name and are all quite similar. For this reason we will mention over-all logic of the entire group (ORS, ORA, ANA, ANS and ERA) and leave the details of individual instructions to the flow diagram text.

It should first be mentioned that these instructions, being of a "logical" nature, do not use the sign position to show "plus or minus". It is simply the high order digit of the number and is treated as any other bit in the ANDING or ORING process. This bit will have usage of the "sign" position of the SR and "P" position of the ACC.

The 7090 finds ORING of two numbers quite easily done. Either one input, OR the other input, to a register may set it. The two inputs need only be made available at the same time.

The ANDING of two numbers will also use the "ORING capabilities" mentioned above. Circuits to AND in the registers are not available; nor can we "AND" two numbers in the ADDERS. It just so happens, however, that the "OR of the comp of two numbers - re-complemented = the AND of those two numbers." Let's take a closer look by doing it.

Let's AND 631_8 to 567_8 the way the 7090 does.

$$631_8 = 110\ 011,001_2$$

$$567_8 = 101\ 110,111_2$$

$$\begin{array}{l} \text{The comp would be:} \quad 001\ 100\ 110_2 \\ \quad \quad \quad \quad \quad 010\ 001\ 000_2 \end{array}$$

$$\text{The OR of the comp is:} \quad 011\ 101\ 110_2$$

$$\text{Now we RE-comp:} \quad 100\ 010\ 001_2 = 421_8$$

$$\begin{array}{l} \text{Let's AND the numbers} \quad 110\ 011\ 001_2 \\ \text{visually to check our} \quad 101\ 110\ 111_2 \\ \text{answer:} \quad \underline{\hspace{1.5cm}} \\ \quad \quad \quad 100\ 012\ 001 \quad = \quad 421_8 \quad (\text{Looks good!}) \end{array}$$

One other operation we must do is an EXCLUSIVE OR. An example would be the EXCLUSIVE OR of 16_8 and 22_8 .

$$\begin{array}{rcl}
 16_8 & = & 001,110_2 \\
 22_8 & = & 010,011_2 \\
 \hline
 \text{The } \overline{\wedge} & = & 011,101_2 \\
 \text{In octal} & = & 35_8
 \end{array}
 \quad (\overline{\wedge} = \text{symbol for exclusive OR})$$

Again, as with ANDING, there are no circuits in the 7090 to perform this operation directly. EXCLUSIVE ORING is like ADDING the two numbers and IGNORING all carries, like so:

$$\begin{array}{rcl}
 & & 101_2 = 5_8 \\
 & & 011_2 = 3_8 \\
 \text{ADDED, but with all} & & \hline
 \text{carries blocked!} & & 110_2 = 6_8
 \end{array}$$

This is the basis for the derivation of the formula used to get the EXCLUSIVE OR on the 7090. The formula is: $2(A \text{ or } B) - (A + B) = \text{the exclusive OR}$. This formula is equal to "ADDING the two numbers and subtracting out an amount equal to the carries".

These are the methods by which the 7090 does all of the logical AND/OR instructions. The "Why do they work?" question may not be clear or easy to understand except to an accomplished mathematician in some cases. However, after taking examples through, with the flow diagram text, most of your questions should be answered.

FLOW DIAGRAM TEXT

ORS (OR to Storage)

For discussion of this instruction refer to Figure 5.3-67 of Page 129, Manual of Instruction, Ref #223-6895. If you have just finished FAD or FDH, this flow diagram should bring a sigh of relief.

The instruction is to OR the contents of ACC (P-35) with the contents of the core location specified. This is accomplished by making core storage think it is to take a STORE type cycle. The contents of the ACC are gated to the SB and MDR.

At the same time, however, the ORS instruction sends a line to core which allows the strobe pulses. This means that the MDR sees the SB as input and the sense amps as another input at the same time. A bit from either one OR the other will put a bit in the corresponding pos. of the MDR. The "ORING" of the two numbers actually takes place in storage in the MDR.

ORA (OR to ACC)

Refer to Figure 5.3-68 on Page 129, same manual.

The logic of this instruction is like that of ORS with the exception that the OR'd result should be in the ACC. This ORING of the two numbers takes place early in the instruction as shown on the flow chart.

As one of the numbers is brought from storage into the SR (at E7D1), the ACC is also gated into the SR. Now, with the numbers already OR'd together in the SR, all that is left is to place the result in the ACC.

ANA/ANS (AND to ACC/AND to STORAGE)

The operation of these two instructions is somewhat different and more difficult than the OR type instruction. This is due mainly to the fact that no machine circuits are available to AND together two numbers. However, since "ORING" can be done very easily with the ckts we do have, we will find that ANDING can be done by ORING!

It very conveniently happens to be a fact that the ANDING of two numbers can be accomplished by "ORING" them in complement form and re-complementing the result. For example, let's AND together the following two numbers:

$$\begin{array}{rcl} 11_8 & = & 1001_2 \\ 15_8 & = & 1101_2 \\ \hline & & 1001 \text{ (ANDED result)} \end{array}$$

Now, let's do it the way the 7090 would.

$$\begin{array}{rcl}
11_8 & = & 1001_2 \quad \text{complemented} = 0110_2 \\
15_8 & = & 1101_2 \quad \text{complemented} = \underline{0010_2} \\
& & \text{the OR'd result} = 0110_2 \\
& & \text{Recomp \& ans. is} = 1001_2 \\
& & \text{(It works every time--try it!)}
\end{array}$$

Once convinced that this "method" works, let's look at Figure 5.3-69, Page 130 and see how this is accomplished. The only difference between ANA and ANS is that for ANS an extra E cycle is required to store the ANDED result and restore the original contents of the ACC. (The original contents of the ACC should not be changed whenever avoidable.) For this reason much "jockeying" of the two numbers is required to save the original ACC contents of BOTH instructions; even though it is only necessary on ANS.

In the flow chart the first 3 blocks after E time complements the ACC contents and saves ACC Q pos in SR-Q. The next block brings the 2nd number into the SR from storage. Then we must exchange the ACC and SR so that we can get the complement of the 2nd number we just brought out (the SR has no "complementing" ability). At this point the original ACC contents are in the SR in comp form and the number from storage is in the ACC.

Next, in L time, we will comp the present contents of the ACC and EXCHANGE the two registers once more. The reason for this is that the SR will be used to "OR" these two numbers together. When this is done, it's present contents will be lost and we do not want to lose the original ACC number (in case the instruction is ANS).

At the top right of the chart the ORING takes place. The ACC is gated into the SR and the SR is gated into the SR. One of the desirable abilities of shift cell ckts makes the latter gating possible.

Now that we have the OR of the two numbers complemented, we must exchange the registers once more so the ACC can be used to re-complement the result. With this accomplished, ON ANA ONLY, we can end op in L time and re-comp the ACC during I time next.

However, should the instruction be ANS, we must request an E cycle, re-comp the ACC and get the result up to the SR so that it can be stored. It will end op in E time and we have only to restore the original ACC number by recomplementing.

ERA (Exclusive OR to ACC)

We have seen that ORING two numbers is not difficult in the 7090. Also, by using an ORING operation, we were able to AND two numbers. However, getting the EXCLUSIVE OR is another story. It can be done in the 7090 by using the following formula.

$$\begin{array}{rcl}
2(A \text{ or } B) & - & (A + B) = \text{The exclusive} \\
& & \text{OR or numbers} \\
& & A \& B
\end{array}$$

(2 times the OR, minus the sum, equals the exclusive OR)

The logic behind the derivation of this formula is covered on Page 132 for those interested. We will not, at this time, try to prove its validity other than to take an example through the flow chart.

The formula for doing this exclusive OR will, of course, be broken down into the following steps:

The formula is: $2(A \text{ or } B) - (A + B)$

The steps used by the 7090 will be:

1. OR A and B
2. ADD A and B
3. Get the comp of the sum of A and B
4. Double the OR of A and B (by shifting left 1 place)
5. Subtract the sum of A and B from twice the OR of A and B

These steps are well laid out in the flow chart on Page 131, Figure 5.3-70. Let's take the contents of the registers through the flow chart using the following example--assuming 5 position registers.

CAL	A	$L_B =$	$L_B = +22_8$
ERA	B		$L_B = +22_8$

6 pos regs assumed, plus sign pos
(We know by just looking that the answer should be 113_8 --right?)

On our flow chart the value of -31_8 and $+22_8$ is ADDED at E9 (D3) (LOGICALLY, signs don't count) and the SUM of A and B is put in the ACC at E11 (D1). AT THE SAME TIME the sum of A and B is gated to the ACC, the value "A or B" is being formed in the SR. This is done by "ACC (P-35) to SR" at the same time the SR is gated to the SR. So, by the end of the E cycle we would have the following register values (in binary and 6 position regs assumed).

SR	1 011 011	=	(A or B)
ACC	1 101 011	=	(A + B)

(P of ACC shown; and sign pos of SR)

During L time we next comp the sum of A and B so it will be ready for subtraction. The result is put in the ACC at L2(D1) and at L6(D1) we switch the registers so we can shift the OR of A and B (now in the ACC) one place left. With this accomplished, the contents of the ACC and SR would look like so:

SR	0 010 100	=	ones comp of (A + B)
ACC	0 110 110	=	2 (A or B)

The instruction then ends operation and these two registers are sent to the ADDERS (along with a bit in 35 to get the two's comp of A + B) and the output of the ADDERS is placed in the ACC. This should represent the EXCLUSIVE OR of A and B. Let's do it and see.

SR	0 010 100
ACC	0 110 110
<u>Hot Bit</u>	<u>1</u>

Added Result= 1 001 011 = 113_8 (perfect!)

SUMMARY

LOGICAL AND/OR INSTRUCTIONS

Three main points of the logical instructions that we have just discussed are:

1. The SIGN position is just another bit--it does not determine any more or less than the other 35 positions how the operation is to be performed. For this reason, we would expect other types of logical instructions to be used in conjunction with those mentioned, such as: CAL, ACL, SLW, etc.
2. ORING is easily done on the 7090 by giving two inputs to a register position, at the same time, and letting either input turn it on. However, ANDING is not so easily done. The answer to this problem was to COMPLEMENT the two numbers, OR them together and then RECOMPLEMENT the result. This we found was an effective method of ANDING.
3. Last was the good old EXCLUSIVE OR. Neither ANDING, nor ORING, seemed to solve this problem. However, someone noticed that if two numbers were ADDED and all CARRIES suppressed, the result would be the EXCLUSIVE OR of the two numbers. From this was derived the formula which is EFFECTIVELY ADDING the numbers and SUBTRACTING the CARRIES after. This formula is: $2(A \text{ or } B) - (A + B) =$ the EXCLUSIVE OR. The 7090 found the exclusive OR by solving this formula in a thorough series of sequential steps.

REVIEW QUESTIONS
LOGICAL AND/OR INSTRUCTIONS

1. Where does ORING take place on ORS? On ORA?
2. How does the 7090 "AND" two numbers?
3. What is the formula for EXCLUSIVE OR used by the 7090?
4. Give the contents of the ACC, or storage, depending on the instruction for each of the following.

a.	CLA	Y	L	
	ORS	Z	X	-301440216
	HTR		L	Y +200770351
			L	Z -172063314

Loc Z = ?

b.	CAL	X
	ANA	Z

Acc = ?

c.	CLA	X
	ERA	Y

Acc = ?

- | | | | | |
|----|-----|-----|-----|--------------|
| 5. | 000 | CAL | 100 | 100 = +37700 |
| | 001 | ERA | 101 | 101 = -12345 |
| | 002 | SLW | 102 | |
| | 003 | HPR | | |

Give contents of 102 after the above program if the +A0 (L2D1) at 02A2E04 page 02.12.30.1 has no output. Assume a 15 bit storage (sign and 14 places).

6. Why is the contents of the ACC and SR exchanged so often on ANA and ANS?
7. The sign position of the SR is compared to the P position of the ACC on an ERA instruction. (T or F)

ANSWERS - LOGICAL AND/OR INSTRUCTIONS

1. MDR, .SR
2. By complementing them, Oring the comp and then recomping
3. $2(A \text{ or } B) - (A + B)$
4. a. -672773355_8
b. $+301\ 040\ 214_8$ (bit in ACC P)
c. $+0101\ 330\ 147_8$
5. Loc 102 =
6. To save original contents of ACC in case of ANS
7. True

Over-all Logic

Convert Instructions can best be described as "table look-up" instructions. If they are thought of in this manner, it will greatly simplify the logic of how they are performed.

We have three such instructions on the 7090: CVR, CRQ and CAQ. They are all similar in that to perform any useful purpose a table must already exist in core storage before they are executed. This table may consist of "BCD equivalents", "binary equivalents" or other pertinent data which the convert instruction CAN "LOOK UP".

To know where the instruction must look in these tables to find what it wants, part of the ACC or MQ is used in conjunction with its address (21-35 of the convert instruction). The convert instruction also is able to reference the table several times, if desired, by use of a "count field" in its decrement. A count larger than 6 is not normally desired however (as you will see later). Each time a reference to the table is made, an E cycle is required to bring out the contents of that location.

The Manual of Instruction has an excellent write-up of each convert instruction. A detailed example of how each of the three may be used is also given. For these reasons, no "flow diagram text" will be given in this area. In its place is the suggested reading and study of pages 132 to 140 (inclusive) (manual 223-6895) with concentrated efforts toward understanding the "table look up" concept.

SUMMARY OF CONVERT INSTRUCTIONS

Some interesting points should have been discovered while studying convert instructions.

1. They can be used to convert binary numbers to BCD.
2. They can be used to eliminate "leading zeros" in a BCD field.
3. They can convert BCD numbers to binary.

If, AND ONLY IF, the proper tables are available in core storage.

Not only are the "equivalent" numbers in these tables important, but the ADDRESS portion of numbers in these tables is used to determine the NEXT table look up address.

Another interesting point is the use of the index register. Say, for example, on CAQ we had a BCD number which took 8 BCD characters to represent it (instead of 709542, the number was 84709542 - page 139). Since we can only do 6 BCD char at a time, we would want to remember our last table look up address when we did the last two digits.

One last comment is to note the ADDITIVE accumulation of address on this CAQ instruction. I'm sure you can see that if our table was in upper core, it would be possible for a carry to "goof up" our answer. This is only because addresses, like 77700_8 , added six times will cause a carry into position 19--which is the low order part of our answer in the example given.

REVIEW QUESTIONS FOR CONVERT INSTRUCTIONS

1. Why is a count field greater than 6 usually desirable?
2. What tells CAQ to End Op?
3. Use of any index register is allowed on convert instructions. (T or F)
4. Why isn't the contents of the MQ lost during execution of CAQ?
5. No E cycle is necessary in order to develop the first table address in a convert instruction. (T or F)
6. The sum of SR(21-35) and MQ(S-5) is set into XRA each E1(D1) during a CRQ operation if the instruction contained a bit in 20. (T or F)
7. If CVR with a count of 6 is given and "shift ACC Right" fails to come up, the machine will make six references to the same address in storage. (T or F)
8. The original contents of the MQ are lost during the execution of a CAC instruction. (T or F)
9. When a CAQ with a count of seven is given, the seventh reference to storage will always be to the same address as the first. (T or F)
10. When the count field exceeds a given amount, the converts may be I. A. (T or F)
11. A convert cannot take more than six E cycles. (T or F)

ANSWERS - CONVERT REVIEW QUESTIONS

1. The resultant answer would be meaningless since the "correct answer" would be used, in some cases, to determine further table look up.
2. SC = 0
3. False - C only
4. The MQ is ring shifted MQS to MQ35
5. True
6. False
7. False (The 1st and 2nd references MAY be different)
8. False
9. False
- 10.
11. False

08.00 DATA CHANNEL

08.01 INTRODUCTION AND I/O PROGRAMMING

Objectives: Without reference we must be able to accomplish the following:

1. Define the need for Data Channels
2. List the capabilities of all models of the 7607.
3. List the nine select instructions including a description of each.
4. Define the purpose of the load channel instructions
5. Differentiate between the two load channel instructions by using them in an example program.
6. Draw the format that all I/O Commands use.
7. List the seven I/O Commands including a description of how each operates for both Read and Write operations.
8. Define the use of indicator 19.
9. Given an I/O program following a WRS we must be able to draw the resultant tape including number of records, record size, gap size, and where tape will stop.
10. Given an I/O program following a RDS and a description of the record configuration on a previously written tape, we must be able to list the number of words that will be read, what records these words will be read from, and where tape will stop at the completion of the program.
11. Define the conditions that will allow the following instructions to transfer.
 - a. TCO
 - b. TCN
 - c. TRC
 - d. TEF
12. Define the conditions that will allow the following instructions to skip.
 - a. BTT
 - b. ETT
 - c. IOT
13. List the registers that will be stored following the execution of a SCH. Also define the bit positions where these registers will be located in the stored word.
14. Describe the Data Channel conditions that will allow the SCH to be executed.
15. Define the three PSE instructions associated with Data Channel giving an example of use for each.

Reading Assignment: 7607 Data Channel Instruction Reference Manual (IRM) Pages 5-16
7090 Data Processing System Reference Manual (SRM)
Pages 57-64, 90-96, 40-44

Review Questions:

1. What output units are normally connected to the 7090?
2. What input units are normally connected to the 7090?
3. Why must these devices be connected to the system through a 7607?
4. How many channels can be connected to the 7090?
5. What is the difference between a 7607 MI and 7607 M2?
6. What causes an output device to go into operation?
7. Basically, how does a "Write Select" select the channel and unit desired?
8. What are the objectives of a Read Select Instruction?
9. RDS and WRS are Data Select instructions. (True or False)
10. The "Op code" for select instructions remains the same regardless of which channel is selected. (True or False)
11. Describe what will take place if a WTDC 5 is encountered in CPU and there is no channel C connected to the system.
12. What is the purpose of a RCH instruction?
13. What is an I-O command?
14. What is the format of an I-O command?
15. Are I-O commands decoded in CPU?
16. In the following examples indicate what will be written on tape.

WTBA 4
RCHA X

- | | | | |
|----|---|------|------|
| a. | X | IOCD | 0,,3 |
| b. | X | IOCP | 0,,2 |
| | | IOCD | 2,,1 |
| c. | X | IOSP | 0,,2 |
| | | IOCD | 2,,1 |

- d. X IORP 0,,3
IOCD 0,,3
- e. X IOCP 0,,1
IOSP 1,,1
IORP 2,,1
IOCD 3,,3

17. Indicate what will be written for each of these examples.

Note: Use for A and B Only

WTDA 4
RCHA X
LCHA B

- a. X IOCT 0,,1
Y IORP 5,,2
Z IOCD 0,,4
B IOCD 1,,2

- b. X IORT 0,,3
Y IORP 5,,1
Z IOCD 0,,2
B IOCD 1,,3

Use for "C" WTDA 4
RCHA X
HTRH

18. The following will pertain to WC=zero situations. Indicate what will be written on tape for each example.

WTBA 4
RCHA X

- a. X IOCD 0,,0
- b. X IOCP 0,,3
IOCD 0,,0
- c. X IOCP 0,,3
IORP 0,,0
IOCD 0,,3
- d. X IORP 0,,0
IOCD 0,,3

19. Name the three cycles associated with channel and define them.

20. What will happen if a LCHB is encountered in CPU and Channel B is not Data Selected?

21. What will happen if a RCHB is encountered and Channel B is not Data Selected?

22. Explain what will happen during the execution of the following program when the IOCT does not find LCH decoded in CPU.

```

      WTBA    4
      RCHA    X
      HTR
X IOCT      0,,1

```

23. What is the difference in the operation of an IOCT and IOST while writing?
24. Referring to the following programs, describe how much information will be read into storage. NOTE: Tape contains 10 records of 3 words each.

```

      RTBA    5
      RCHA    A
      CLA     X
      LCHA    B
      HTR

```

- a. A IOCP 0,,5
 IOSTN 5,,1
 B IOCD 15,,5
- b. A IORP 0,,5
 B IOCD 0,,3
- c. A IORP 0,,1
 B IOCD 0,,2
25. Write a program to solve this problem:

Tape contains a 100-word record. Using a LCH and IOCT delay the CPU program until the first 50 words are read into storage from tape; then continue with the CPU program.

08.02 FUNCTIONAL UNITS

Objectives: With the use of Figure 3 in the IRM, we must be able to accomplish the following:

1. Define the purpose of each functional unit located in the 7607.
2. Describe the logic block makeup of all registers and counters located in the 7607.
3. Trace the data flow for a read or write operation from or to any I/O device.
4. Trace the command word path for core to the proper registers in data channel.

Reading Assignment: IRM Pages 11-14, 17-20

Review Questions:

1. Name three possible inputs to the Data Register.
2. The _____ and _____ are count up counters. The _____ is a count down counter. The _____ and _____ are stepped every BDW cycle while the _____ is stepped every BCW cycle.
3. What are the four positions of the Op. Reg. and what do they control?
4. What does the contents of the Location counter indicate?
5. What is the purpose of the Channel Address counter?
6. What controls the outgating of the Tape Register to the R/W Register?
7. What is the purpose of the Delay Counter?
8. How is the L. C. loaded?
9. What does a position of the LRCR being left on indicate?
10. How many outputs are available from the Final Amplifiers? Each position.
11. Trace the path of command from core storage to the proper place in Data Channel. Name all registers and switches included in this path.

08.03 TAPE WRITE SELECT OPERATION

Due to the length of the lecture associated with the Tape Write Select Operation, this section of the Guide will be broken into four parts. The four parts are as follows:

08.03.01	WRS RCH BDW
08.03.02	Writing
08.03.03	Disconnect
08.03.04	Read Checking and Miscellaneous

Each part will have objectives, reading assignment, and review questions.

08.03.01 WRS RCH BDW

Objectives: Without reference we must be able to accomplish the following:

1. Given a channel and tape drive number, we must be able to determine the bit configuration of the address portion of a WRS instruction that would select this channel and tape drive for either BCD or Binary mode.
2. List the major objectives in the selection of a tape drive.
3. Referring to Objective 2 we must be able to define the need or the "why" of each of these objectives.

4. List the five operations that bring up channel interlock.
5. Define the purpose of the lines going to CPU and MPLX from the End Op Control and Channel In Use triggers.
6. Define why the channel registers and counters are reset by a WRS instruction.
7. Describe the circuitry that gives channel the ability to stack instructions.
8. List and define the purposes of the delay counter including the modes of operation.
9. Explain the purpose of the Tape Disconnect tgr.
10. Define the purpose of the Proceed to E circuit.
11. Describe the reason for holding the data register reset during the E cycle of an RCH instruction.
12. Draw a logical AND-OR representation of Channel Input Switch bit 3. Include the input circuits to both the data register and word counter.
13. Using Fig. 10 & 11 of the IRM draw a timing chart depicting the reason we take a dummy B cycle following the E cycle of a RCH.
14. List the three ways that B cycle demands can be satisfied by the cycle timer.
15. Describe the principle of operation of the data channel priority circuits.
16. Define why the channel priority tgr is turned on during E time of an RCH instruction.
17. List the conditions required to request a BDW cycle during a WRS operation. Exclude time.
18. List the three requirements to turn on the Channel Priority tgr. Exclude time.
19. List and define the major operations that are gated by the BDW tgr. being on.

Reading Assignment: IRM Pages 29-34

Review Questions:

1. Name four conditions needed before a Write Select Instruction can End Op.
2. What is the purpose of the End Op. Cntl. Tgr. in channel?
3. What conditions in the tape unit must be satisfied before the GO tgr. is turned on?
4. Why does Busy coming up initiate a Non Data Disconnect?
5. What mode is the D.C. running in for Write Delay?

6. Is it possible for an RCH to hang up in L time?
7. Is it possible for an RCH to turn on the I-O check indicator?
8. What are the objectives of an RCH?
9. What conditions must exist to enable channel to request a BDW cycle to load the Data Reg. with the 1st data word?
10. How do we establish which channel will use a B cycle?
11. What must be accomplished during the BDW cycle?
12. Which of the following are reset as a result of a data select instruction?
 - a. WC
 - b. LC
 - c. CAC
 - d. Indicator tgr. 3, 1, 2, 18, and 19
 - e. DR
13. What are the three ways the B cycle demands can be satisfied by the cycle timer?
14. If the _____ trigger in data channel may be turned on, the data channel will accept a select instruction.
15. Channel A has been given an RCH to a TCH; during E time of the RCH, Channel C, which is more remote, requests B time.
 - a. Why does Channel A get the next B time rather than Channel C?
 - b. Show the system page number, the input, and list the logic blocks (example 2C, 5D, etc) which will block the setting of the priority trigger in Channel C.
16. What determines whether or not a particular channel may get B time priority?
17. Why does selecting a card machine interlock the channel?
18. Work Simulated Machine Error Analysis Problem #1. Located at the end of the Data Channel section of this Guide.

08.03.02 WRITING

Objectives: Without any reference we must be able to accomplish:

1. Define the purpose of the following triggers
 - a. Write Condition Trigger
 - b. Write Trigger Release Trigger
 - c. Sync Trigger
 - d. Sync Gate Trigger
 - e. Write Clock Control Trigger
 - f. No Echo Trigger
 - g. Demand Gate Trigger

2. Define the number of write oscillators required and purpose of each for all models of data channels.
3. Describe the purpose in developing the "Write First Word Test" line.
4. Define the two ways a Tape Demand is requested during a WRS operation.
5. List the sequence of events that occur from Tape Demand to the turning on of the BDW Required Trigger.
6. List the write clock pulses and the purpose of each.
7. Define the count sequence of the Group Counter and what bits are transferred to the R/W register for each count.
8. Define how a "C" bit is generated while writing.
9. Describe the operation and function of the Write Translators.
10. Describe how noise is checked for in the interrecord gap.

With the use of all reference material we must be able to accomplish the following:

1. Trace the sequence of events, as shown on figures 14, 18, and 23, through the ALD's to determine the circuits involved in the development of each logical function.

Reading Assignment: IRM Pages 35 - 42

Review Questions:

1. How does channel know when to start writing on tape?
2. What is the purpose behind having Write First Word Test generate a tape demand?
3. Is Write Condition used to control the Write Clock?
4. When we check for Echos, how many do we check for?
5. How do we recognize that a complete word has been written?
6. What must be accomplished now that a complete word has been written?
7. "Write Demand" (60.36.06.1, 1C) gates the first word from the DR to the TR. True or False.
8. The drive to the write clock stops when the "Write Clock Control" tgr. is reset. True or False
9. On a Model II data channel why do we need 4 oscillators for driving the write clock?

6. What must be accomplished now that the last character has passed the read head?
7. Work Simulated Machine Error Analysis problems 5, 6, 7 and 8 which are located at the end of the Data Channel section of this Guide.

08.03.04 READ CHECKING & MISCELLANEOUS

This section covers read checking, commands other than IOCD, and "Space Tape" operations.

Objectives: Without reference we must be able to accomplish the following:

1. List the six error conditions that are checked for during a Write Operation.
2. Give an example describing how each of the six errors could be caused.
3. Define why different clip levels are applied to the final amplifiers.
4. Describe why the read clock can be started from any bit position of either Skew A or B registers.
5. List the read clock outputs that are used for a read check operation and the major function of each.
6. Define how the machine determines which skew register to gate to the LRCR.
7. Define a condition of when the RDD tgr. would not be set at RC7 time.
8. Define when Control Word Gate line is developed relative to tape Demand and Word Count for an IOST, IOSP, IOCP and IOCT.
9. Describe what causes an IRG to be generated for an IORP or IORT command.
10. Define when an IORP or IORT will proceed or transfer.
11. List the objectives for a BCW cycle that is initiated by a proceed type command.
12. List the objectives for an ECW cycle that is initiated by a "T" type command.
13. Define what initiates a disconnect if an LCH is not waiting in L time when a transfer pulse is developed.
14. Define the conditions that determine whether or not a "Space Tape" will occur.
15. Define how an IOCD WC=0 is simulated when a WRS is not followed by an RCH.
16. Define why it is impossible for an LCH to be waiting in L time if an IOCT or IOST WC=0 is loaded by an RCH.

10. On a WRS operation, what line simulates the first demand to bring the first word to be written from the DR to TR?
11. How is the out gating of the TR controlled so that one character at a time is sent to the R/W register?
12. Work Simulated Machine Error Analysis Problems 2, 3, and 4. Located at the end of the Data Channel section of this Guide.

08.03.03 DISCONNECT

Objectives: Without any reference we must be able to accomplish the following:

1. List the conditions required to bring up the Data Disc. Pate line for a WRS operation using a ICOD Command.
2. Describe why the "no-name" trigger on 60.36.01.1 (3A and 3B) is required to reset the WR Clock Ctrl. Tgr.
3. Name the trigger that initiates the disconnect in channel.
4. Name the trigger that initiates the disconnect in TAU.
5. List the modes the Delay Counter runs in during the disconnect.
6. Define the conditions that allow the Delay Counter to run in each of its modes.
7. Describe how a check character is written on tape.
8. Describe how we can continue to read tape after WDD 20 tells the tape drive to stop.
9. Identify the trigger which when on indicates the last character of the record has been read.
10. Describe what a LR CR trigger that is on at RDD 136 time indicates.
11. Describe the operations that are initiated by the RDD 144 pnke.

Review Questions

1. The first "Write Demand" following WC=O develops "Disc. Call". True or False
2. Why do we turn on the Data Disc. Gate Tgr. ?
3. What must we accomplish during the disconnect procedures?
4. When is the check character written?
5. When is the GO tgr. reset?

17. Describe why an IORT or IORP WC=0 generates an IRG instead of spacing tape when it follows an IOCP with a word count.

Reading Assignment: IRM Pages 42-51

Review Questions:

1. On a read check operation, an error in the Skew A VR checker will turn on the TAU error trigger. True or False?
2. Explain why the +P RC4 Tr is OR'ed with +P 1st bit on page 61.10.10.1.
3. On a read check operation, a Skew B VRC error will turn on the TAU error trigger. True or False?
4. WD Noise error could be caused by missing oxide on tape. True or False?
5. Match column B with the statements of column A.

- | | |
|---|----------------------------|
| (1) Sample write skew gate error. | a. RDD 128 |
| (2) Resets RDD trigger | b. RC 4 |
| (3) Run Delay Counter in micro-second mode | c. RDD trigger on |
| (4) Gate Skew B register to the LRCR | d. Skew A VRC error RC 10 |
| (5) Start Read Clock | e. RC 10 |
| (6) Will turn on redundancy trigger | f. RDD 128 |
| (7) Samples LRCR | g. Skew A VRC error |
| (8) Resets Read Condition | h. Write Compare |
| (9) Prevents reset of RDD by RC4 | i. WDD 20 |
| (10) Indicates next character is check character | j. RDD 10 |
| (11) Forces even redundancy for check character VRC check | k. LRCR error |
| | l. RDD 136 |
| | m. Check Character trigger |
| | n. RDD 36 |
| | o. WD Noise error |

6. With the following program, how many tape demands will it take to disconnect?

WRS	1201	100	IOCP	WC = 5
RCHA	100	101	IORP	WC = 5
		102	IOCD	WC = 5

7. How is a "Data Disc" accomplished when the channel is loaded with an IOCD with a WC = 3, but the channel cannot get B time from CPU?
8. Draw a sketch showing how the following program would write the information. Give a command number under corresponding information. (Tape is moving left to right)

WTBA	1	(1)	CW - IOCP	(WC = 20)
RCHA	CW	(2)	IORP	(WC = 30)
LCHA	CW + 7	(3)	IOCP	(WC = 25)
		(4)	IOSP	(WC = 15)
		(5)	IORT	(WC = 30)
		(6)	IOCP	(WC = 5)
		(7)	IOCT	(WC = 15)
		(8)	IORP	(WC = 25)
		(9)	IOCP	(WC = 20)
		(10)	IOCD	(WC = 15)

9. What is the pulse that causes a proceed on an IORP and how long is the pulse?
10. What prevents more than one Record Control pulse in an EUR gap?
11. What "AND" circuit in systems brings up the line that will cause a disconnect on IORT when no LCH is waiting in CPU?
12. B output of C block located at 3G is shorted to plus P level. 61.20.30.1
On tape write select operation the following will be noted:
 - a. Nothing will be written on tape
 - b. Operation will be normal for write although read checking will give errors.
 - c. Data will not be gated to tape register since we are unable to receive.
 - d. Record will be written correctly except failing to write check character.
13. H output of DE block 3H is shorted to a plus P level. 61.60.20.1. On a tape write operation the following will occur:
 - a. Operation is normal in all respects.
 - b. The record will be written correctly but will not be read checked.
 - c. The record will be neither written nor read checked.
 - d. Record will be written properly, not read checked, and tape will fail to stop.
14. C output of Driver Indicator block located at 2B is shorted to minus N level. Page 61.60.32.1. On Write Tape Select operation, the following will be noted:
 - a. Record will be written in normal operation in all respects.
 - b. Characters of record will be written normally but check character will not be written and tape will not stop.
 - c. Check character will be written but tape will not stop.
 - d. Tape will stop at end of record but a check character will not have been written.
 - e. On records that should have a check character of no bits (no check character) operation will be normal.
15. Solve Simulated Machine Error Analysis problems 9, 10, 11, 12 and 13; which are located at the end of the Data Channel Section of this Guide.

08.04 TAPE READ SELECT OPERATION

Objectives: Without reference we must be able to accomplish the following:

1. Define the bit positions, of the address portion of an RDS instruction, that are decoded to determine which channel.
2. Define the purpose of the End Op Control trigger.
3. Define the function of the channel interlock line. Also list the five operations that bring up channel interlock.
4. List the major objectives in the selection of a tape drive for an RDS operation.
5. List the three major functions of TAU busy.
6. List the major objectives for the ECW cycle of an RCH instruction.
7. Define the purpose of the first character trigger.
8. List the two conditions that can activate the first bit line for an RDS operation.
9. Referring to objective 8, define the purpose of having two ways to develop the first bit line. Also give an example of each.
10. List the Read Clock outputs and the objective of each for an RDS operation.
11. List the conditions required to turn on the BDW Req trigger for an RDS operation. (Exclude time)
12. List the sequence of events (objectives) for a TAU disconnect using an IOCD WC = RL.
13. List the sequence of events (objectives) for a channel disconnect using an IOCD WC = RL.
14. Differentiate between the operation of an IOCD WC = RL and the following:
 - a. IOCP or IOSP WC < RL
 - b. IOCP or IOSP WC = RL
 - c. IOCP WC > RL
 - d. IORP WC < RL
 - e. IORP WC = RL
 - f. IORP or IOSP WC > RL
 - g. IOCD WC < RL
 - h. IOCD WC > RL
15. Define where the data transfer is blocked for an RDS operation with a bit in IND 19.
16. Define the two purposes of the character counter.

17. List the four error conditions that are checked for during an RDS operation.
18. Referring to objective 17, give an example of how each of the four errors could be caused.

Reading Assignment: IRM - Pages 51-67

Review Questions:

1. What conditions must be present before a Read Select instruction can End Op?
2. What conditions in the tape unit must be present before the Go tgr can be turned on?
3. How long does tape move before we start to read?
4. Why must the DR be reset during the ECW cycle of the RCH?
5. In the following example, what would appear in the LC upon completion of the ECW cycle?

RCHA 500
6. How many outputs are available from each position of the Final Amplifiers?
7. What is used to start the Read Clock?
8. What can be used to start the Read Clock if a character does not get into Skew A?
9. What will result if Skew A VRC is incorrect?
10. When is the 1st Character tgr. turned on? Off?
11. List five things that will occur at RC7.
12. How can we tell the difference between an RC cycle for the 1st character and others?
13. When is the 1st character gated to the TR?
14. What occurs at RC 4? RC 6?
15. How do we realize that we are in the 6th RC cycle?
16. At the end of the 6th RC cycle, where is the 6th character?
17. How do we gate the 6th character to the TR?
18. What gates TR to DR?
19. What turns on the DR LD tgr?

20. What turns on BDW Req?
21. What main objectives must be accomplished during the BDW cycle?
22. How do we recognize the end of a record?
23. What are some items that must be accomplished with an "RDD 36"?
24. When do we sample the LRCR for errors?
25. When is the Go tgr. reset?
26. How many errors are possible while reading? List them.
27. When will an IOCP proceed when used in conjunction with a read operation?
28. In the following examples, is it possible that the IORP command will be skipped?

a. IOCP 0,, 3
 IORP 0,, 3**
 IOCD 0,, 3

There are many 3 word records on tape.

b. IOSP 0,, 3
 IORP 0,, 3**
 IOCD 0,, 3

c. IORP 0,, 3
 IORP 0,, 3**
 IOCD 0,, 3

29. When using an IOCDN, how is the WC reduced since there are no BDW cycles needed for "N" type commands?
30. Can an IORP be used to read more than one record? An IOSP?
31. What happens when an EOF is read?
32. How does channel recognize an EOF?
33. Solve Simulated Machine Error Analysis Problems 14, 15 and 16; located at the end of the Data Channel section of this Guide.

08.05 TCH-IA OPERATION

Objectives: Without reference we must be able to accomplish the following:

1. Define the purpose of the look-ahead circuits employed by a TCH or IA command.
2. List the major objectives that occur after a TCH is decoded on the MX SB.

3. List the major objectives that occur after an indirectly addressed command is decoded on the MX SB.
4. List the cyclic make-up of an indirectly addressed TCH command.

Reading Assignment: IRM - Pages 67-73

Review Questions:

1. What is the function of a TCH command?
2. Where is the TCH decoded and why?
3. What is the purpose of activating Retain Priority with the TCH, ADR, CNTL, TGR?
4. During the cycle that a TCH is brought out of core, what address is gated to MAR and how is this accomplished?
5. How does channel recognize that a command is indirectly addressed?
6. In the following example, what will be the address of the first data word?

0	IOCD*	1,, 3
1	IOCD*	2,, 3
2	IOCD	3,, 3

7. Can a TCH be indirectly addressed?
8. In the following example, which command will be finally loaded into channel?

0	TCH*	1
1	IOCD*	2,, 3
2	IORP	3,, 3
3	IOSP	4,, 3

9. What effect will execution of this command have on channel operations?

10	TCH	11
11	TCH	10

10. How many BCW cycles are required to execute an indirectly addressed TCH?

08.06 NON-DATA SELECT OPERATIONS

Objectives: Without reference we must be able to accomplish the following:

1. Define the purpose of the following instructions:
 - a. Rewind
 - b. Rewind - Unload
 - c. Backspace Record

- d. Backspace File
- e. Write End of File
- f. Set Density

2. First, illustrate a tape showing groups of records separated by end of files. We must then be able to indicate on this illustration the final position of the R/W head, relative to tape, after a Rewind, Rewind - Unload, Backspace Record, or Backspace File has been executed. This must be done for all possible positions of the R/W head prior to the execution of these instructions.
3. Illustrate a tape showing the final result of a Write End of File instruction.
4. Define why Write End of File and Set Density instructions bring up channel interlock.
5. Define why it is possible to change density in the middle of a Backspace Record or Backspace File operation.
6. If a tape drive is in write status, define why we must move tape forward before we go backward during a Rewind, Rewind - Unload, Backspace Record, Backspace File instruction.
7. Define the purpose of the erase trigger during the execution of a Write End of File instruction.
8. Define the purpose of running the delay counter in milli-second mode for an SDN instruction.

Reading Assignment: IRM - Pages 74-81

Review Questions:

1. What conditions must be present before a Rewind Instruction can End Op?
2. How does channel know that the Rewind instruction has been accomplished?
3. Is there any difference in the execution of a Rewind instruction that selects a tape unit that is in write status?
4. During the execution of a backspace record, how do we recognize that tape has reached the beginning of a record?
5. What should occur if a BSR is given to a tape unit that is positioned at LP?
6. What will happen if a BSR is given to a tape unit that is positioned between Load Point and the 1st record?
7. What is the major difference between the execution of a BSR and BSF?
8. Briefly explain the execution of Write End of File.

9. What is the difference in the OP Code of a SDHA 5 and SDLA 5?
10. Can a SDN instruction be used to run the delay counter constantly?

08.07 DATA CHANNEL TRAP

Objectives: Without reference we must be able to accomplish the following:

1. Define the purpose of the Data Channel Trap feature.
2. List the three courses of traps.
3. Define the two CPU instructions associated with Data Channel Trap.
4. Referring to Objective 3, define the purpose of each instruction.
5. Define the terms "disabled" and "Inhibited" as associated with Data Channel Trap.
6. Describe the priority sequence that controls what channel can trap first.
7. Describe how TEF and TRC instructions operate on a channel that is enabled to trap.
8. Define the storage addresses that are normally reserved for Data Channel Trap. Consider a eight channel system.
9. List the conditions required to turn on the Channel Trap trigger on 02.10.56.1.
10. Referring to the Channel Trap Timing chart on page 91, figure 58, of the Instruction-Reference manual, we must be able to define the purpose (objective) of each line on the chart.

Reading Assignment: IRM Pages 87 - 92
7090 Reference Manual Pages 15 - 17

Review Questions:

1. What is the advantage of using Data Channel Trap?
2. What are the conditions that may initiate a trap?
3. How do we establish what conditions on which channels can cause traps?
4. What is the purpose of an Enable instruction whose operand is zero?
5. Can you enable for an EOF without also enabling command word trap?

6. Can a trap signal that occurs while a channel is inhibited be remembered?
7. What happens when more than one channel wants to trap at the same time?
8. True or False? If an ENB instruction is given to enable a channel for EOF trap after an EOF has been read, the channel will not perform the trap.
9. What conditions may cause a command word trap?
10. In addition to the EOF indicator, the _____ trigger, the _____ trigger, and the _____ trigger must be on to request a channel trap priority.
11. True or False? If channel A were enabled to trap on EOF and an EOF condition arose, the EOF enable trigger would be reset.
12. What prevents a Data Channel Trap from preventing the completion of a MF instruction?
13. When the Restore tgr on 2.10.56.1 goes off at E8 of the STR cycle, it disables trapping. True or False?

08.08 CONSOLE OPERATIONS

Objectives: Without reference we must be able to accomplish the following:

1. With the 7617 in manual - off line, describe the effect(s) that a depression of each of the following keys has on the Data Channel. Include status of registers, counters, and control triggers.
 - A. Load Command
 - B. Load Data Register
 - C. Load Location Counter
 - * D. Store Data Register
 - * E. Display Storage
2. * Also describe the effect(s) if the console was in manual-on line.
2. List the key(s) and switch(s) set up that you would use if you wanted to scope all circuitry associated with writing ones on a tape drive.
3. List the key(s) and switch(s) set up that you would use if you wanted to scope all circuitry associated with reading fours (4) from a tape drive.
4. One of the uses of the CSRI and CSRO is to cause repetitive BDW cycles, so list the key(s) and switch(s) set up that you would use to allow scoping of the BDW cycle circuitry.
5. Define what circuit requests a B cycle every fourth cycle when operating with the CSRI or CSRO.

Reading Assignment: IRM Pages 14-16, 20-23, 92-94

Review Questions

1. How many keys must be used to initiate a Write Tape Cycling operation?
How many switches?
2. If console entry keys 3-17 are depressed and load location counter is then depressed, what will be in the location counter?
3. Why must you load commands before loading the Data Register?
4. To punch the locations 1000 - 1027 on a card, give the proper sequence of keys, switches, etc., that must be used.
5. If position 18 of the entry keys is depressed and then you load command, where will this bit appear?
6. What is the purpose of the Reset Key?
7. On a Read Tape operation with the stop on error switch on, tape will stop at the end of the record in which the error occurred. (True or False)
8. Does the tape cycle switch have any affect on operations such as BSR, BSR, and REW?
9. When operating in "Write Cycling", describe what occurs when tape reaches the end of tape marker.
10. Is Stop On Error effective during a write operation?

08.09 Data Channel Miscellaneous

This section will consist of objectives, reading assignment, and review questions for the list of topics.

- A. Transfer and Skip Instructions
- B. I/O Checks
- C. Store Channel Instruction

Objectives: Without reference we must be able to accomplish the following:

1. Define the purpose of the following instructions:

A. TCO	E. ETT
B. TCN	F. TRC
C. TEF	G. IOT
D. BTT	H. RTT

2. Referring to objective one use each instruction in a program example. Exclude RTT.
3. List the conditions that will cause a I/O check for a RCH instruction.
4. List the conditions that will cause a I/O check for a LCH instruction.
5. List the conditions that will cause a I/O check for a Write Select operation.
6. List the conditions that will cause a I/O check for a Read Select operation.
7. Define what will prevent a SCH instruction from operating if the registers and/or counters are changing.

Reading Assignment: IRM Pages 81-84, 86-87

Review Questions

1. What conditions will cause an I-O check during a Read Select operation?
2. Does an RCH, given when the channel is not Data Selected, cause an I-O check?
3. Does a BTT skip when it finds the beginning of tape indicator on?
4. A channel must be selected before a SCH can operate correctly. (True or False)
5. When executing IOT, is there any way of determining which channel caused the I-O check?
6. On a SCH the Word Counter is routed to positions 3-17 of the Channel storage bus switches. True or False?
7. On a SCH the location counter is routed to positions 21-35 of the channel storage bus switches. True or False?

08.10 Data Channel Examination

08.11 9T55

Objectives: Without reference we must be able to accomplish the following:

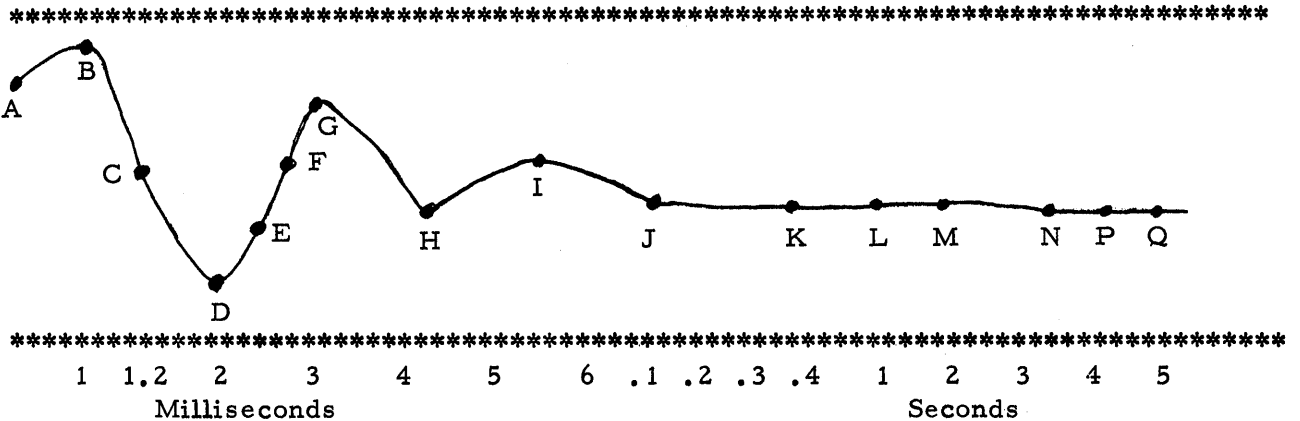
1. Define the purpose of the 9T55 diagnostic program.
2. Describe the general operation of the program.
3. Given a 9T55 printout, which includes a graph, we must be able to determine the condition of the tape drive that the test was run on. That is, we must be able to determine whether or not the drive needs maintenance and if so, what maintenance.

Reading Assignment: *9T55 Supplement

*This supplement is located after the review questions of this section. It consists of a sample 9T55 printout followed by an explanation.

Review Questions:

1. After studying the graph, answer the questions that follow.

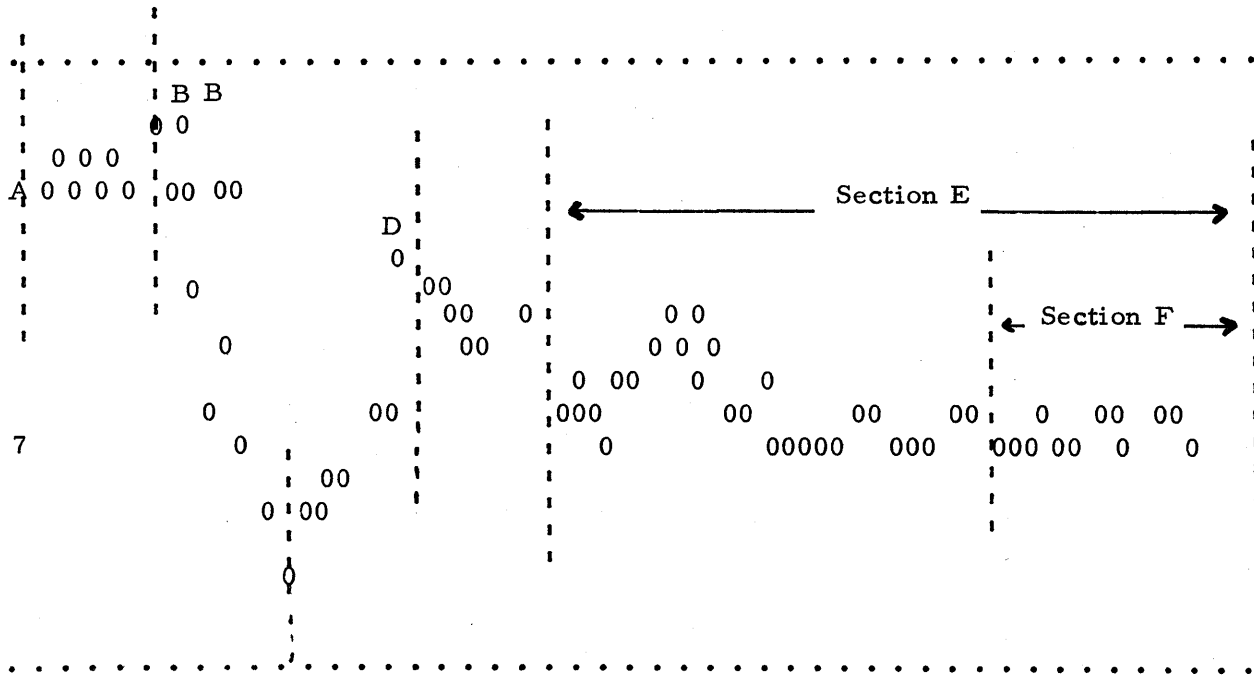


- (1) What point on the graph indicates minimum go down time?
- (2) Which point indicates where the longest record gaps are written?
- (3) Which point indicates where the shortest record gaps are written?
- (4) From this point on a sloping line would indicate a "count of five" problem.
- (5) The point representing max. overshoot away from the drive capstan is.....
- (6) The point representing max. overshoot toward the drive capstan is.....
- (7) _____ point is higher on the graph than _____ point because at this time "forward coast" is approximately equal to "go down time".

9T55 Supplement

TF A2 MOD4-HI INTER-RECORD GAP TEST WRITTEN 70 RECORD GROUPS

TF A2 MOD4-HI PLOT OF RECORD GAP READ TIME VS WRITE GO DOWN TIME
 READ MILSEC



6
 MIN 1 1.2 2 3 4 5 6*.1.2 .3 .4 1 2 3 4 5
 WRITE GO DOWN TIME MILLISECONDS * SECONDS

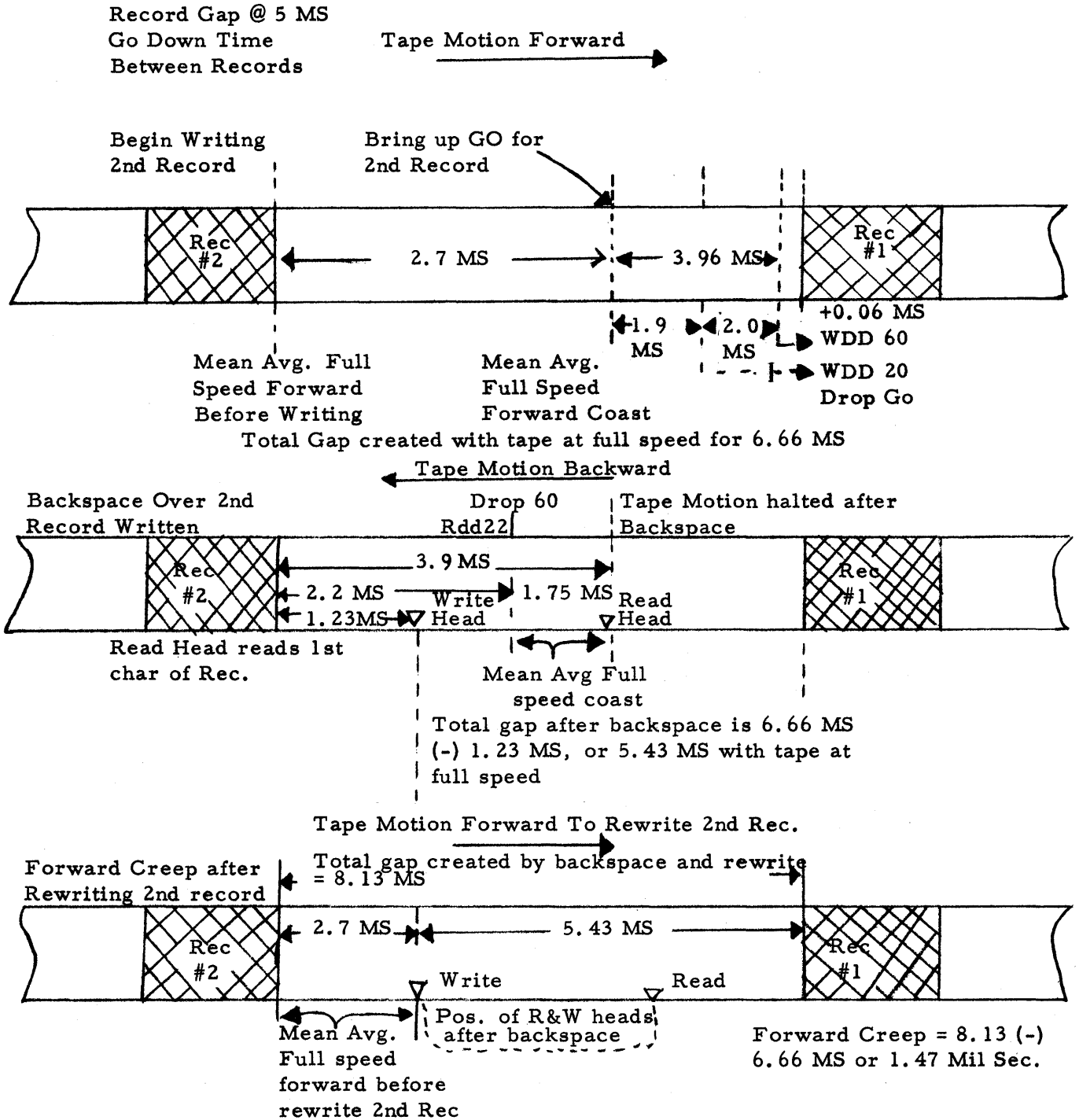
TF A2	MOD4-HI	70 REC GPS READ	LOW	RANGE	AVERAGE
		MINIMUM GO UP TIME	6.88	.40	7.13 MSEC
		VAR GO DOWN MIN TO 1.20 MSEC	7.74	.24	7.83 MSEC
		VAR GO DOWN 1.40 TO 5.90 MSEC	6.54	1.09	7.17 MSEC
		VAR GO DOWN 6.00 TO 430.00 MSEC	6.94	.24	7.02 MSEC
		VAR GO DOWN .50 TO 5.00 SECS	6.94	.13	7.01 MSEC
		MINIMUM GO DOWN TIME	7.66	.15	7.72 MSEC

TF A2	MOD4-HI	INTER-RECORD GAP TEST COMPLETE	LOW	RANGE	AVERAGE
		30 BKSP-RDS COMPLETE	4.22	.31	4.36 MSEC
		BKSP START TO CHK CHAR LOADED RH COL	3.97	.34	4.12 MSEC
		RD SEL TO FIRST BIT LOADED RH COL	6.16	.15	6.23 MSEC
		RD SEL TO FIRST BIT LOADED LIICOL	6.38	.19	6.46 MSEC

TF A 2	MOD4-LO	WR-BKSP-WK CREEP	LOW	RANGE	AVERAGE
		VAR DEL BEFORE WR 09 OPNS FWD CREEP	.98	.26	1.11 MSEC
		VAR DEL BEFR BSKP 09 OPNS FWD CREEP	1.00	.31	1.20 MSEC
		DEL-WR-DEL-BKSP 09 OPNS FWD CREEP	1.11	.17	1.19 MSEC
		5 OPNS WORST CASE GDN MIN MS, BKSP, GDN	1.02 SC, WR		
		05 OPNS FWD CREEP	1.09	.17	1.18 MSEC

OK

Figure Page #1



B. Highest plot on the graph.

This point occurs almost always during the low go down time range (min. to 1.2 milsec). The diagnostic increases the go down time for each variable record gap 0.052 milsec from min. to 1.2 milsec go down time. Note that the first ten plots on the graph represent the total change of only approximately 0.5 milsec. Since each succeeding plot from the first is using more of the full speed forward coast (which is at least 0.9 milsec and is usually about 1.2 milsec) the record gap length in milsec should increase as go down time is increased. The highest point will be reached when the go down time is equal to full speed forward coast. At this point a record gap of maximum length has been created which should not exceed the maximum length.

C. Shortest record gap throughout the range of the graph

It should not be less than specified in the diagnostic. As the go down time is increased beyond the full high speed coast time, record gaps become shorter. This is caused by the mechanical delay in the prolay to again move tape once the nylon idler has allowed tape to break physical contact with the drive capstan. The point at which the minimum length record gap will be created is when the driving prolay arm assembly is magnetized toward forward go after it has attained maximum inertia moving from forward go to neutral status caused by a previous impulse to stop tape motion. This point is generally reached at between 2 and 3 milsec of go down time on a model four tape drive.

D. As go down is further increased from point C the driving prolay has more time to reach neutral status before being impulsed toward go. The record gap length in milsec will increase as go down time is increased until point D is reached (usually between 3 and 4 milsec go down time). The driving prolay arm assembly has a tendency to overshoot neutral on its travel from go to neutral. The peak usually associated with point D is caused by a magnetic impulse to attract the prolay arm assembly to go when it has already attained maximum inertia toward neutral (and thus the go position) from the stop direction. This is caused by the overshoot resulting from a previous order from the data channel to stop tape movement.

The motion control diagnostic 9T55B prints a graphic representation of the record gap length in milliseconds (vert. axis) and go down time variable from minimum to 5 seconds (horizontal axis). For the Customer Engineer to correctly evaluate tape motion from this graph he must first know how the graph for a well adjusted tape drive should look. He should also understand why this is so.

Let us first understand how long minimum go down time actually is. For reference a model four tape drive at 556 BPI will be used. This gives a tape speed of 0.1125 in. per millisecond and delay counter timings of 1 microsecond per step in microsecond mode, or .1 millisecond per step in millisecond mode. The physical distance between the write and the read heads on this tape drive is 0.3 in. With the information given, the minimum go down time may now be computed. For the last character of a record on a write operation until the go line drops will be WDD 60 in microsecond mode for a check character plus WDD 20 in millisecond mode for a total of 2.06 milliseconds. From writing to read checking the last character of a record is 0.3 in. times 0.1125 in. per millisecond or 2.67 milliseconds. Add to 2.67 milliseconds the time necessary to run the delay counter to RDD 144 plus the 10 microsecond EOR single shot and it is noted that from writing the last character of a record to disconnect is 2.82 milliseconds. Since go is not dropped until 2.06 milliseconds after the last character written this factor must be subtracted from 2.82 milliseconds. The result equals 0.76 milliseconds, or the minimum actual time that the go line is held down.

Now that minimum go down time is established there are several other areas of the graph which should be examined.

A. Record gap in milliseconds at minimum go down time.

From write to read checking the last character of a record equals 2.67 milliseconds on a model four tape drive. Full speed forward coast is 0.9 milliseconds minimum. (Usually about 1.2) From computing minimum go down time it may be noted that minimum forward full speed coast exceeds minimum go down time. Hence when go is brought back up tape is still moving at full speed. Tape will move at full speed from last character of the last record written for 2.67 milliseconds, plus RDD144, plus the microsecond EOR single shot, plus WD 50 for a total of 7.824 milliseconds. There for the record gap in milliseconds at minimum go down time equals 7.8 milliseconds.

E. Go down time 4 milsec. to maximum go down of 5 sec.

As go down time is increased from approximately 4 milsec. to the maximum go down time of 5 sec. tape is allowed to come to a complete stop in the variable gap before being ordered to go. These succeeding gaps may be readily calculated. From min. stop envelopes of 0.9 milsec. full speed coast and complete stop by 3.0 milsec. with 50% amplitude of 2.1 milsec. there is a mean average forward coast of 1.9 milsec. at full speed. From go up until WD50 is a constant of 5.0 milsec. The forward start envelope must reach full amplitude by 3.3 milsec. With the mechanical delay in prolay motion until tape starts moving the mean average full speed forward motion until WD50 is approximately 2.7 mil. Add the time from the last character of a record until go is dropped for WDD 60 plus WDD20 or 2.06 milsec. plus mean average forward coast of 1.9 milsec at full speed plus 2.7 milsec. forward motion at full speed to the first character of the next record. The sum total is a record gap of 6.66 milsec. duration. The 6.66 milsec record gap length multiplied by .1125 milsec. per in. equals a physical gap length of 0.749 inches. The total range from min. gap length to max. gap length should be quite close with possibly some oscillations noted in the 4 milsec. to 6 milsec. go down time range. The variable gap length during the portion of the graph from 4 milsec. to 5 sec. go down time is usually closer to 7.1 milsec. duration on a well-adjusted tape drive. This is dependent on the forward full coast time being greater than the min. of 0.9 milsec. (It usually is closer to 1.2 milsec) and/or the forward start envelope reaching full amplitude in less than the maximum allowable time of 3.3 milsec. from start of go.

F. Long Go Down times of .5 sec. to 5 sec.

Tape motion becomes more critical of prolay and capstan operation as the go down time increases into the .5 sec. to 5 sec. range. As the prolay arm is moved less and less frequently anything which might tend to decrease the force with which the idler strikes the capstan or the penetration of the idler into the driving capstan has an even more adverse effect on tape operation than at lower go down times. Some of the things which might cause this are improper prolay lubrication drive gaps so small that the armature to core clearance becomes too large for the armature to seal which may allow tape to slip; capstans which have become hardened through use and do not allow proper penetration of the idler into the capstan may also allow tape to slip.

One other factor which should be examined is forward creep. (See Figure Page #1) From previous sections it has been shown that a record gap on a model four tape drive with go down of something greater than 4 milsec should be of 6.66 milsec duration at full tape speed. This is broken down as follows: from last character of last record there is 2.06 milsec. till go is dropped plus 1.9 milsec. mean average full speed coast time and 2.7 milsec. full speed forward motion till first character of next record. Assume that a second record is written and then backspaced. For the tape drive to exhibit the characteristic of forward creep the backspace operation must stop tape with the write head closer than 2.7 milsec. to the first character of the second record written and the one just backspaced over. The amount of forward creep will be 2.7 milsec. minus this distance. When the second record is backspaced and the first character written is read in a backward direction the RDD will start timing out and will drop go at RDD22 with the delay ctr in millisecond mode. This results in the read head going backward into the record gap for 2.2 milsec. at full speed. With the specification for backward stop of at least 0.9 milsec. full speed coast (usually about 1.2 milsec.) and down to 50% amplitude by 1.7 milsec. and at full stop within 3 milsec. the mean average full speed backward coast is approximately 1.7 milsec. A delay of dropping backward go of 2.2 milsec. plus coast time of 1.7 milsec. equals the total distance the read head has moved backward into the record gap. Since the distance between the read and write heads is 0.3 in., or 2.67 milsec., the write head moved backward into the gap 2.2 plus 1.7 minus 2.67 or 1.23 milsec. If this second record were now to be rewritten, the record gap would be increased by the mean average value of 2.7 milsec. for forward tape motion at full speed before writing the first character minus 1.23 milsec. for the distance the write head had previously moved into the gap on the backspace operation. This results in a forward creep value of 1.47 milsec. The motion diagnostic 9T55B causes an OK to be printed over a wide range in the computed forward creep value. Since the right stop capstan should be adjusted on line for forward creep, a computed value within the OK range of the diagnostic is usually quite readily obtainable.

08.12 Introduction to CAU

Objectives: Without reference we must be able to accomplish the following:

1. Define the gate and panels of the 7607 that CAU is packaged in.
2. Describe in general the powering of CAU.
3. Define the following three main functions of CAU.
 - A. Level Letter
 - B. Synchronizer
 - C. Data Assembler
4. Draw a block diagram of CAU, and machines on Data Channel. On this diagram show the logical flow of data and control instructions for the following operations:
 - A. Read Card Reader
 - B. Write Card Punch
 - C. Write Printer
 - D. Write Printer Binary
 - E. Read Printer

Reading Assignment: IRM Pages 94-103
7090 Reference Manual 96-107

1. What model(s) is the 7607 that has a CAU.
2. How is CAU powered in the 7090 system?
3. What is meant by, a function of CAU is to act as a level setter?
4. How many operations can be performed by CAU?

08.13 Basic CAU

Objectives: Without reference we must be able to accomplish the following:

1. Define the functions of the following circuits:
 - A. CB Trigger
 - B. Card Control Trigger
 - C. Card Sync. Trigger
 - D. Card Ring
 - E. CB Counter
2. Draw a block diagram of the Basic CAU.

Using only ALD's we must be able to accomplish the following:

3. Define the circuit which allows the Card Control Trigger to come on only when the CB trigger comes on.

4. Define why the rise of the card drive pulse and the rise of the card sample pulse occur 25 usec apart.
5. Define the circuit which prevents CB Counter from being stepped with the first card sample pulse.
6. Define what signifies when an entire row of a card has been read.
7. Define what determines when an entire card has been read.

Reading Assignment: IRM Pages 103-105

Review Questions

1. What initiates the Basic CAU operation?
2. What length pulse is generated by the turn on of the CB Set trigger?
3. What drives the Card Ring?
4. How many drive pulses does it take to step the Card Ring from all triggers off back to all triggers off?
5. How many times would the Card Ring run to completion for a card reader operation?
6. What steps the CB counter? What is a full count for a card reader operation?
7. What resets the CB trigger?
8. What three conditions cause the Card Ring to be reset?
9. Which Card Ring trigger determines the duration of the Write Right, Write Left, Read Left, Read Right gates?

08.14 Card Machine Power Distribution

Objectives: Without reference we must be able to accomplish the following:

1. Define the purpose of the 55V supply.
2. Define the purpose of the 46V supply.
3. Describe how the 55V and 48V voltages are developed and distributed.
4. Describe the power interlocking that is associated with card machines and channel.

5. Describe the 208V 3Ø 60 cycle distribution from the 7618 to all card machines.

Using ALD's we must be able to accomplish the following:

6. Trace through the ALD's the circuits necessary to get 55V and 46V to the printer, reader, and punch. Include interlocking.

Reading Assignment: 711, 716, 721 Manual of Instruction
Pages 12, 76

Review Questions:

1. 55V and 46V are sent from the printer to Data Channel. What are these voltages used for in channel.
2. Dropping power to data channel drops all power to the card machines. True or False?
3. HD relays 1, 2, 3, and 4, located in section 17B of the printer wiring diagram, control the distribution of only the 46V to the card machines. True or False?
4. Turning off the main line switch on the printer drops till DC power to the card machines. True or False?

08.15 Read Card Reader

Objectives: Without reference we must be able to accomplish the following:

1. List the major steps for a Read Card Reader operation.
2. Define why a RDS 1321 holds up channel interlock.
3. Define what circuit initiates the card reader to start moving.
4. State the cards per minute speed of the card reader.
5. State the number of cycle points the card reader has.
6. Define the number of CB set and CB reset pulses generated by the card reader for one card.
7. Define the purpose of the Read Pulse Control circuit.
8. Describe how 72 lines from the calc entry hubs can be converted into 36 lines without the use of triggers.
9. Define what the "After 12 CB" line signifies.

10. Given a word count less than 24, be able to state when data disconnect will occur.

With reference we must be able to accomplish the following:

11. Trace the circuitry that is used for the normal run-in of cards on the 711 wiring diagram.
12. Trace all circuits shown on Figure 71 of IRM through the ALD's.
13. Trace the circuitry that is used to develop a card EOF on the 711 wiring diagram.

Reading Assignment: IRM Pages 105-106
711, 716, 721 Manual of Instruction Pages 17-24

Review Questions:

1. What brings up read pulse control for a Read Card Reader operation?
2. How many CB reset pulses are sent to CAU from the card reader?
3. How many CB set pulses are sent to CAU from the card reader?
4. Why does "card machine selected" bring up the lines "Data selected" and "Channel Interlocked" in Channel?
5. What is the purpose of the Record Control trigger?
6. What generates a "demand" to be sent to channel to signal that a word has been sent to the DR?
7. If the command in channel is an IOCPN does the information from the card get to the DR? Why or why not?
8. The record control trigger is turned on every row. What makes it active only on the 13th row?
9. What resets the CB counter?
10. If the CAU is disconnected after the 8th row what stops information from being received at Calc entry?

08.16 Write Punch

Objectives: Without reference we must be able to accomplish the following:

1. List the major steps for a Write Punch operation.
2. Define why a WRS 1341 holds up channel interlock.

3. Define what circuit initiates the card punch to start moving.
4. Define the conditions necessary to request a BDW cycle for a Write Punch operation.
5. Define the number of CB sets and CB resets that will be developed by the card punch for one card cycle.
6. Define the purpose of the Write Pulse Control circuit.
7. Define the circuitry that allows 72 columns to be punched at one time when the maximum amount of data we can transfer is 36 bits.
8. Trace the circuitry that is used for the normal run-in of cards on the 721 wiring diagram.
9. Trace all circuits shown on Figure 72 of IRM through the ALD's.

Reading Assignment: IRM Pages 106-108
711, 716, 721 Manual of Instruction Pages 124-131

Review Questions:

1. What brings up write pulse control for a Write Card Punch operation?
2. What generates a "demand" to be sent to channel to signal that the word in the DR has been used?
3. What holds the information in Calc Exit left while Calc Exit Right is being set up so that a whole row can be punched?
4. What resets the CB Set Trigger after it has been set by the EOR set pulse for the 13th row?
5. On a write punch operation, what causes the first reset of the CB trigger?
6. On a write punch operation, what causes the reset of the CB trigger after we have run the Card Ring for our EOR sequence? Be specific!
7. Briefly, what is the purpose of R-31 on the card punch?
8. If input D to plusTA block at 3G (Syst. 80.50.01.1) were shorted to -N, what would be the effect(s) if you gave a Write Punch?

08.17 Printer Operations

Objectives: Without reference we must be able to accomplish the following:

1. Define how a Printer Write Select operation differs from a Punch Write Select operation.
2. Describe an example of use of the Printer Binary operation.
3. Define what allows information transfer to the printer at "1" time only for a Printer Binary operation.
4. Define when End of Record is recognized on a Printer Binary operation.
5. List the major objectives for a Printer Read Select operation.
6. List the three triggers that allow information transfer for a Printer Read Select operation.
7. Define the number of CB sets and CB resets that occur for one print cycle of a Printer Read Select operation.
8. Define when End of Record occurs for a Printer Read Select operation.

With reference we must be able to accomplish the following:

9. Trace the circuitry that is used to place the printer in ready status on the 716 wiring diagram.
10. Trace all circuits shown on Figure 73, 74, and 75 of IRM through the ALD's.

Reading Assignment: IRM Pages 108-111
711, 716, 721 Manual of Instruction
Pages 84, 85, 97 - 100

Review Questions:

1. On a Printer Binary operation, how is printing controlled for just the "1 row"?
2. Does Basic CAU operate normally on a Printer Binary operation, with the exception of Write Pulse Control?
3. On a Read Printer Operation, how many times is the Read Cards trigger turned ON?
4. On a Read Printer operation, how long does the Write Cards trigger stay on after it is turned on by printer read select?

5. On a Printer Read Select operation, when is the Channel Write Cards trigger turned off?
6. Referring to question 6, why is the trigger turned off at that time?
7. When does the EOR pulse occur for a Printer Read Select?
8. What prevents Channel write cards and Write Cards triggers from being turned ON during the last 7 echo times?
9. When will the word used to set up the analyzer at 11 left time of a Printer Read Select be loaded into the DR?
10. An RDS printer operation was initiated using the following I/O program.

```

1000  IOCP   100,, 22
        "    200,, 2
        "    122,, 2
        "    202,, 2
        "    124,, 2
        "    204,, 2
        "    126,, 2
        IOCD  206,, 20

```

The card image locations (100-127) contain the following phrase:

```

THE  LITTLE  RED  FOX

```

The following was printed:

```

727  697767  +72  23-

```

The echo image locations (200-225) contained only the following:

```

Loc. 212  000  000000  000  001
Loc. 216  000  000000  100  000
Loc. 222  000  010000  000  000

```

All other echo image locations are blank.

NOTE:

1. All numeric values except those printed on the 716 are octal.
2. All print magnets are wired to Calc exit left hubs and the Echo hubs to Calc entry left hubs.
3. An analyzer mechanical timing chart is located on page 49 of the 711, 716, and 721 Manual of Instruction.
4. The following indications were noted on the DC console:

```

Op Reg. - Zero
WC      - Zero
CAC     - 226
LC      - 1010

```

5. If at this time you are not screaming and pulling your hair out, you apparently don't understand the problem.

What machine failure could cause this problem? (Limit answers to logic contained in CAU.)

5.18 Sense Operations

Objectives: Without reference we must be able to accomplish the following:

Define the purpose of the sense instructions.

Define the number of sense entries that are available to CPU.

Define the number of sense exits available to the Punch and to the Printer.

Describe an example of use for the sense exits to the punch, to the printer, and to the sense entry from the printer.

With reference we must be able to accomplish the following:

Trace the circuitry used for the sense instructions through the ALD's and card machine wiring diagrams.

Reading Assignment: IRM Pages 111-113
7090 Reference Manual Page 40-41

Review Questions:

- What is the use of sense operations on card machines?
- How could a sense instruction cause an overflow in the printer?
- How many sense exits for the Card Punch?
- How many sense entries to CPU?
- What are some of the uses of the printer sense exits?
- When picked, the Sense Punch will continue to emit a pulse every cycle until the punch is disconnected. Explain how this is accomplished.
- Using the following program, explain the purposes of the +A at 5A (Extender at 5B included) and the +A at 5C, 60.36.02.2.

```
WRS  PU
RCH  Y      Y = IOCD, 30, 100
CLA  X
NOP  Z
PSE  1342
ADD  X + 1
RDS  CR
RCH  K      K = IOCD, 30, 1000
HTR
```

8. How does a PSE 1341 instruction end operation when no card machine is selected (ie. Punch or Printer)?

08.19 Manual Operations

Objectives: Without reference we must be able to accomplish the following:

1. Define the use of any switch or group of switches on the 7617 that are associated with card machine manual operations.
2. Write the program that the Load Cards key simulates.
3. Describe the operation of CAU circuitry when using the hand key with the card cycle switch off. With card cycle switch on.

With reference we must be able to accomplish the following:

4. Trace the circuits associated with any manual key through the ALD's.
5. Trace the Load Cards key operation in the ALD's using Figure 6 in the IRM.

Reading Assignment: IRM Pages 27, 111-113

Review Questions:

1. With card cycle switch on, depressing and releasing the hand key causes the card ring to step far enough to get one gate. What stops the card ring from stepping after we get the gate?
2. On a load cards operation, how does the system know when location one contains the first instruction?
3. On a load cards operation, how do we get the Location Counter to Zero?
4. What is the purpose of the "print binary" switch?
5. Do we set a Channel Unit Select Trigger when the Read Card Reader key is depressed? Why?

SIMULATED MACHINE ERROR ANALYSIS PROBLEMS

On the following pages you will find simulated data channel "bugs". These are depicted via a picture of the data channel console on which is recorded the status of the indicators. The following convention will be used to indicate the status of the indicators.

Indicator appears to be on solid -



Indicator appears to be flickering -

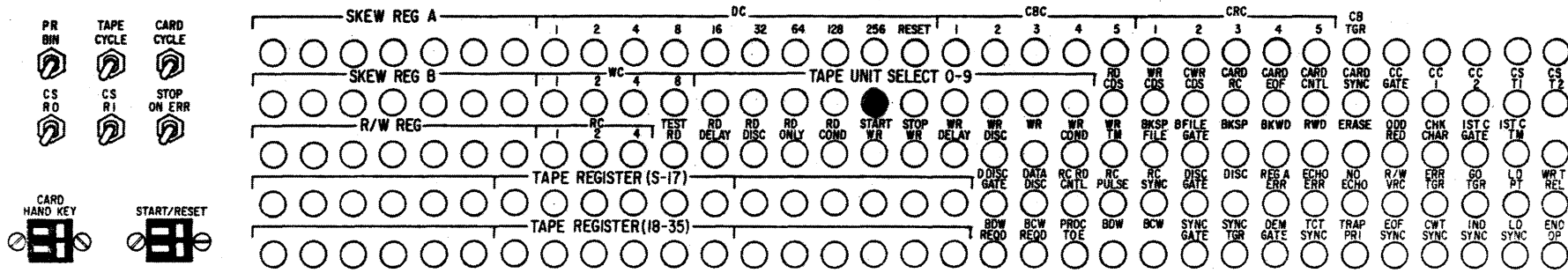


When working with the problems, your objective will be to:

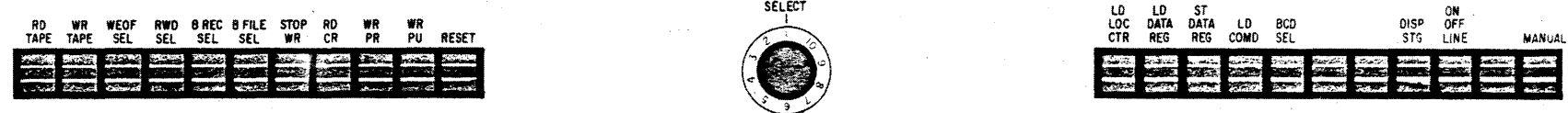
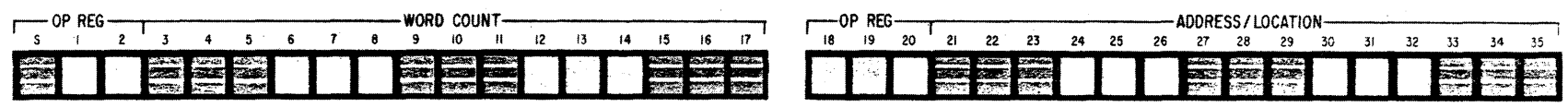
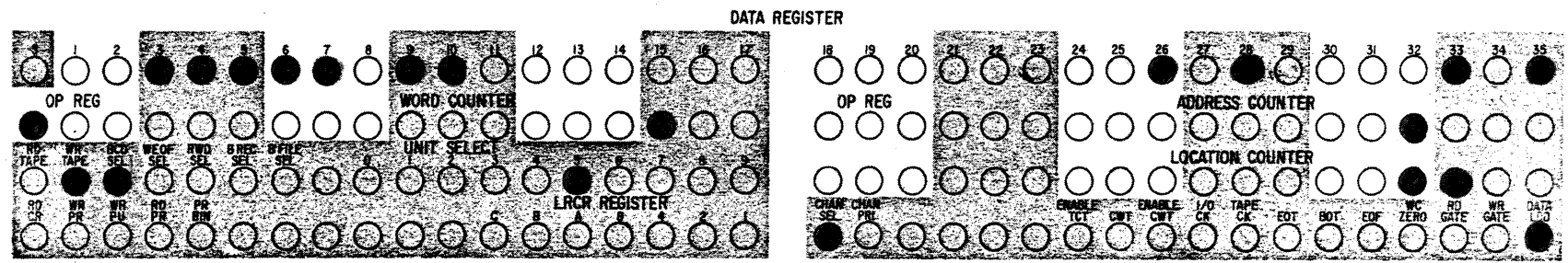
1. Determine what is wrong with the operation described.
2. Be sure you are aware of how it should have operated.
3. Being as specific as possible, pick a trigger or control line which, if failing, could cause the indications shown.
4. Be sure your solution will satisfy all the indications.

The answers, including a brief explanation, can be found at the end of the review question answers for Data Channel.

The sequence for working these problems will be controlled by the review questions.



216



Auto-On Line Status

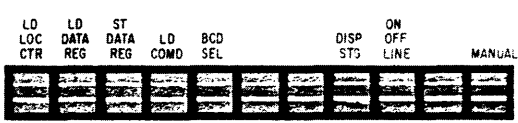
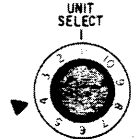
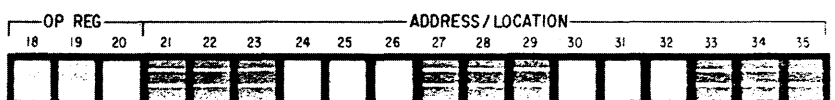
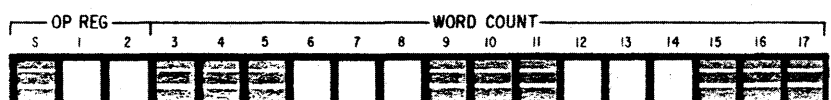
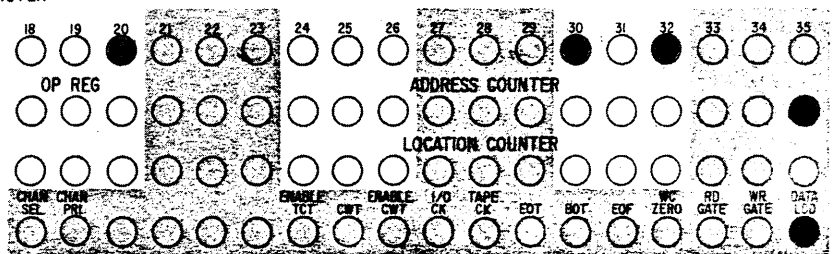
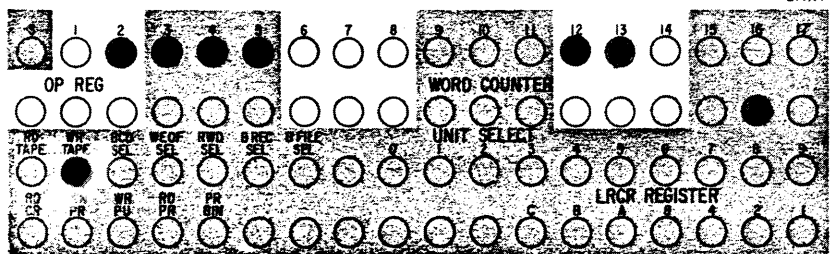
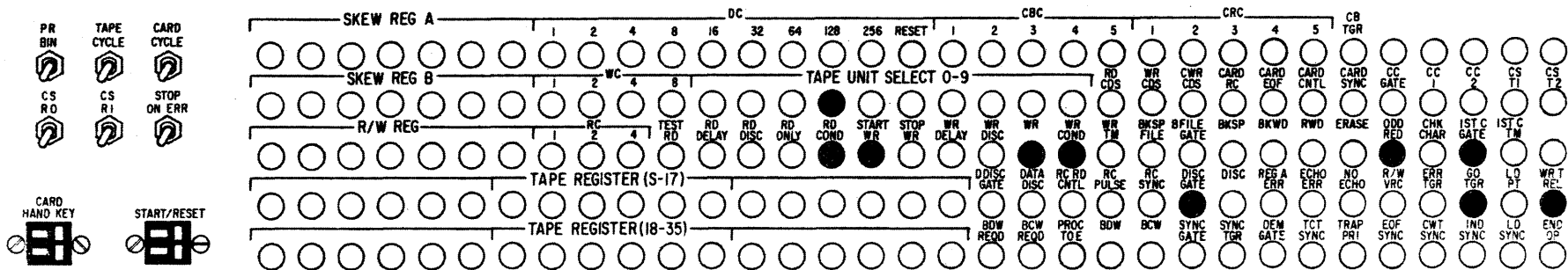
Program:

- ORG 7
- GO WTDA 5
- RCHA X
- REWA 4
- HPR
- X IOCP GO,,5
- IORT X,,2

Symptoms:

- CPU hangs in L time
- Prog. Reg. = +772
- Prog. Ctr. = 12₈

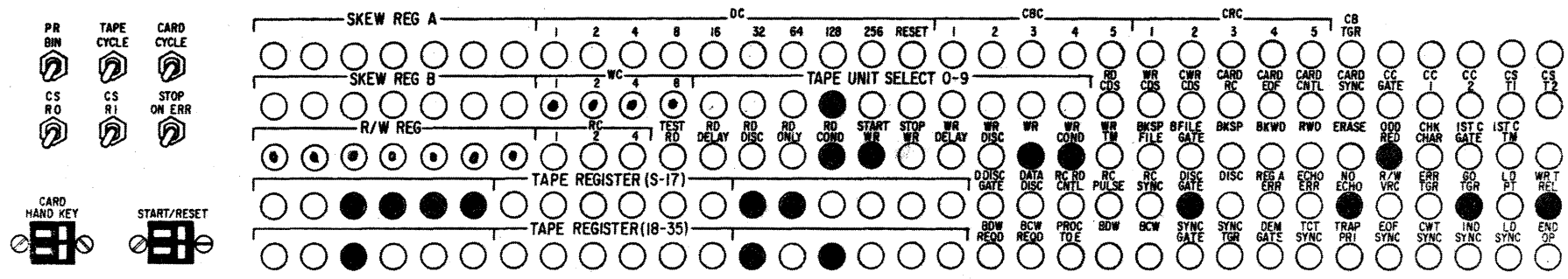
What is the trouble?



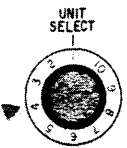
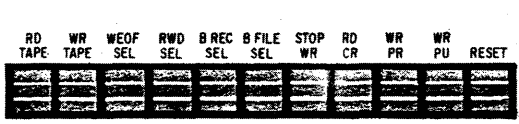
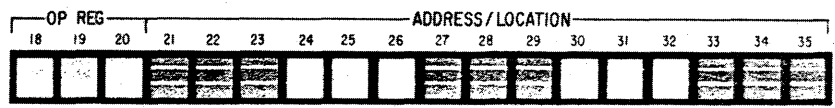
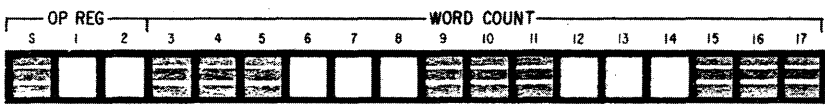
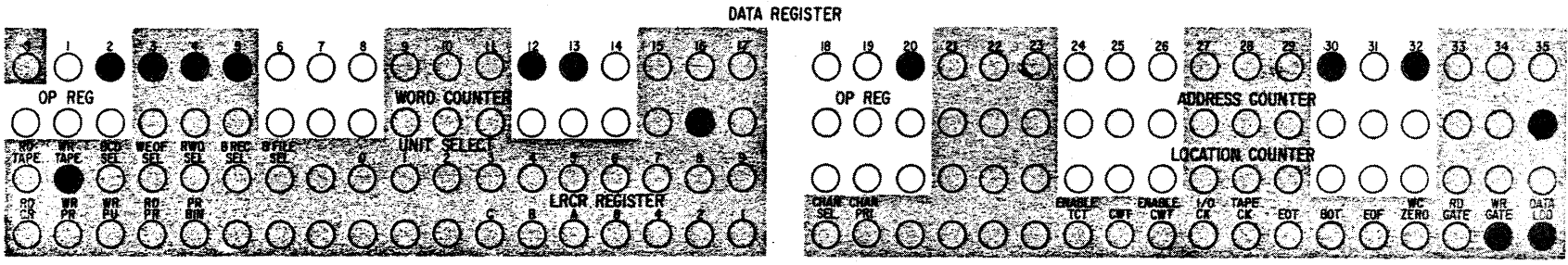
Operation:
 Manual ON LINE
 Load IOCD 0, , 3 octal
 Depress Write Tape

Other Symptoms:
 Tape is moving constantly

217

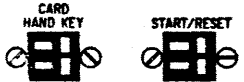
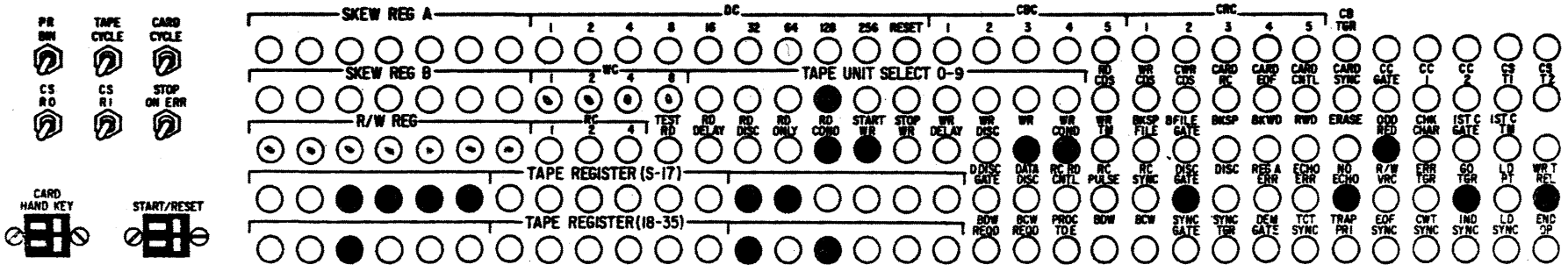


218



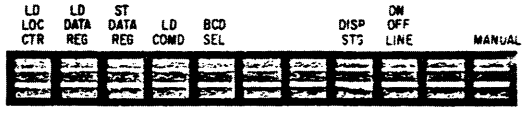
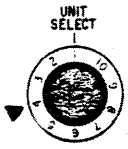
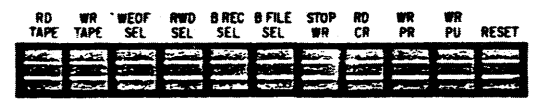
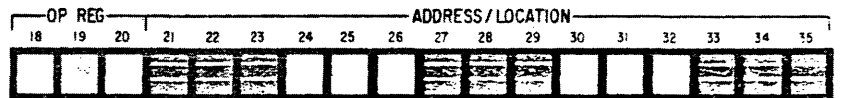
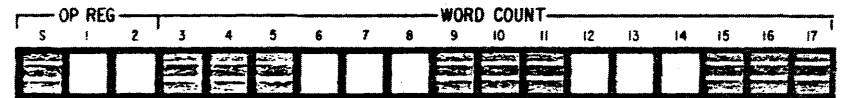
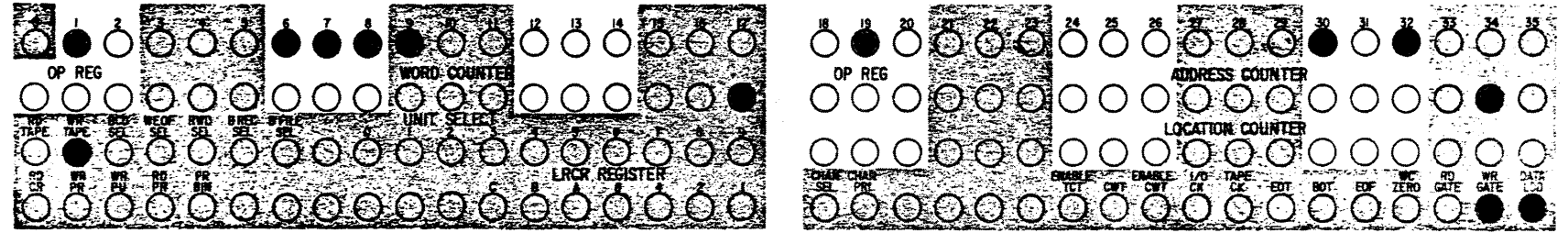
Operation:
 Manual ON LINE
 Load IOCD 0, , 3 octal
 Depress Write Tape

Other Symptoms
 Tape moves constantly
 WC Running
 R/W Reg Indicators blinking



219

DATA REGISTER

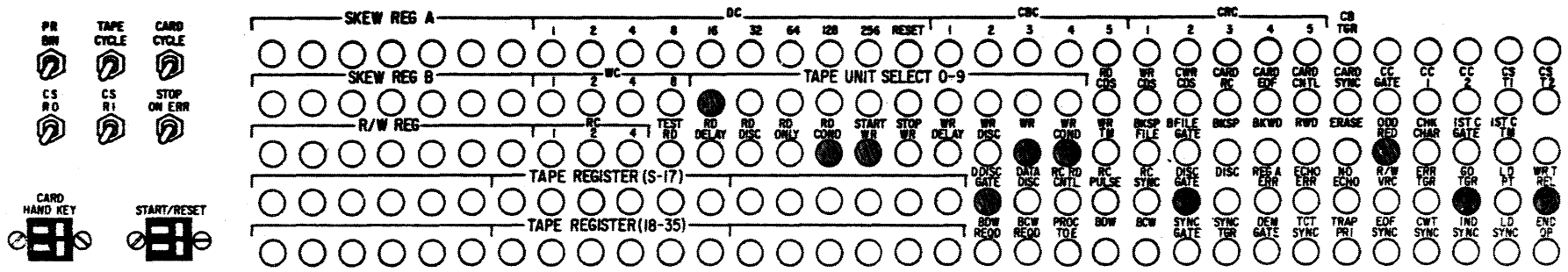


Operation:

- Manual ON LINE
- LOAD IOCD 0, , 3 octal
- Depress Write Tape

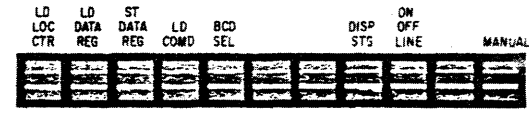
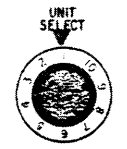
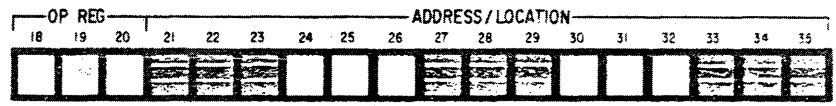
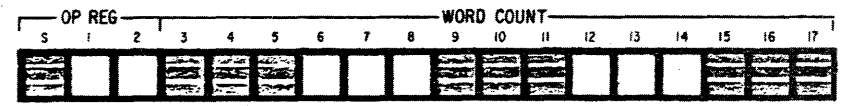
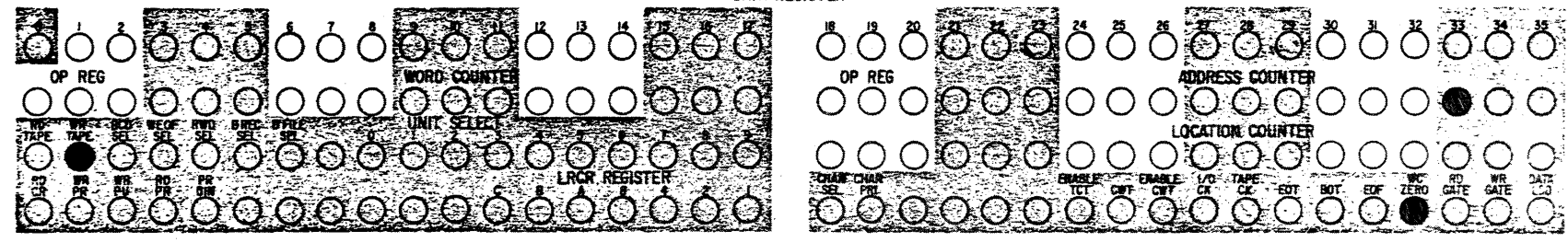
Other Symptoms:

- Tape moving constantly
- WC Running
- R/W indicators blinking



220

DATA REGISTER

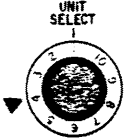
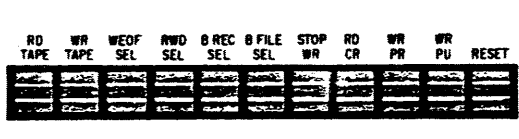
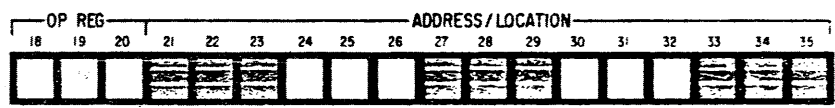
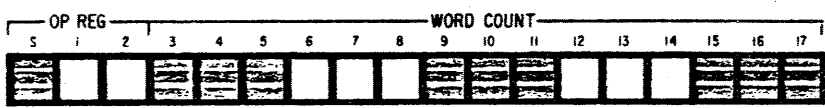
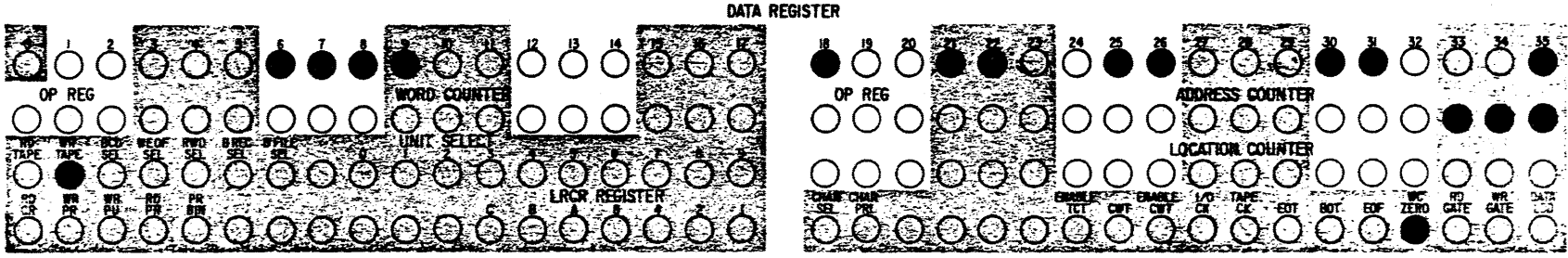
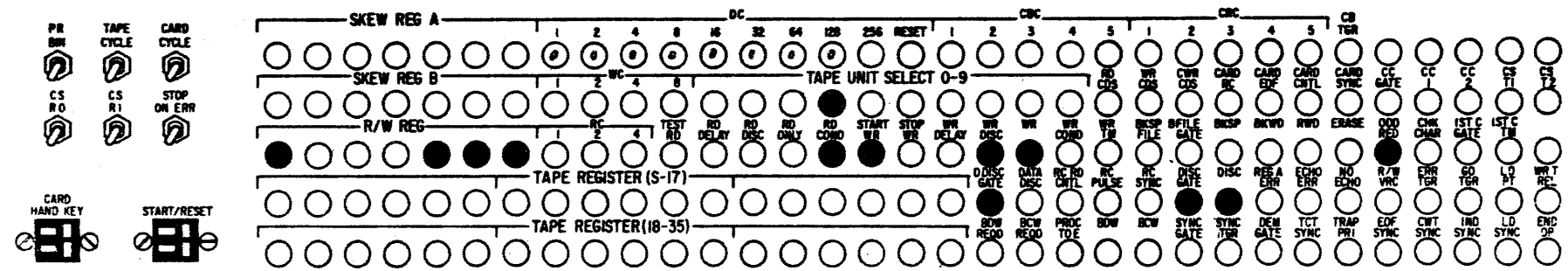


Operation:

- Manual/on line
- Load - IOCD 0,, 4
- Depress WR. Tape

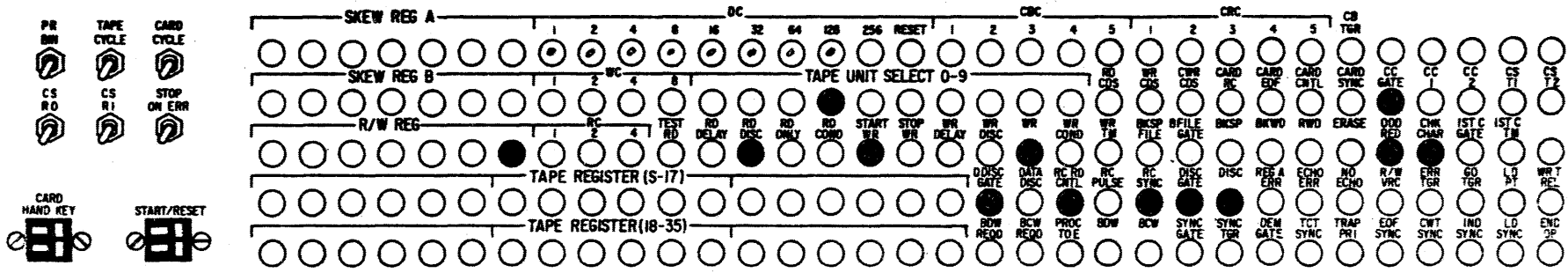
Other Symptoms:

- Tape runs away
- W. C. Ran
- R. C. Ran
- D. C. Ran

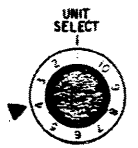
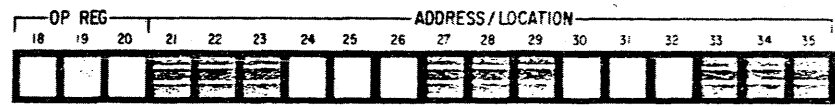
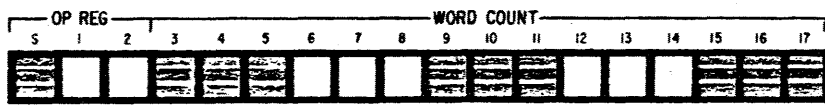
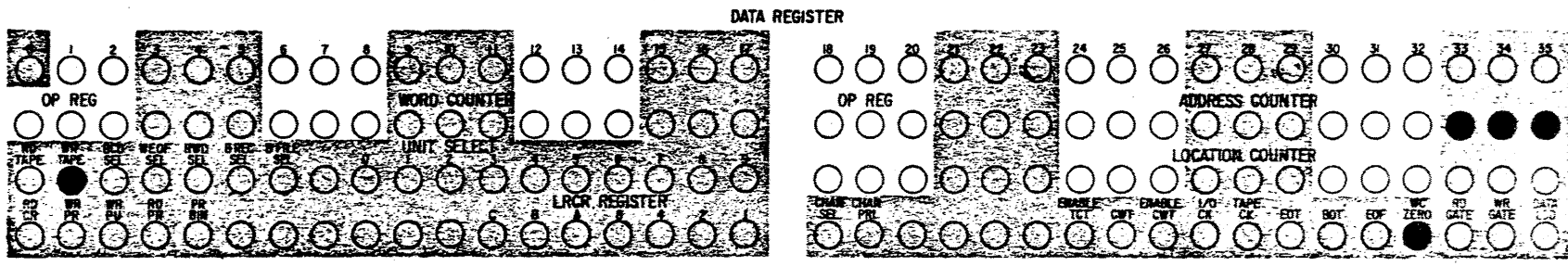


Operation:
 Manual ON LINE
 Load IOCD 0,, 7
 Depress Write Tape

Other Symptoms:
 Tape has stopped
 DC is rippling



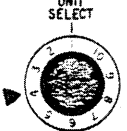
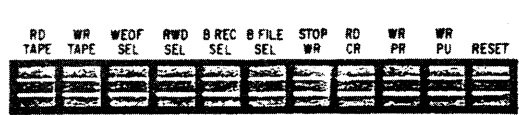
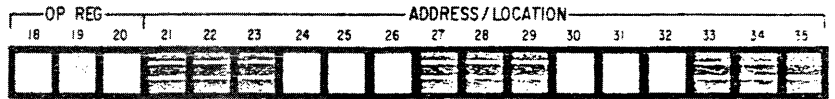
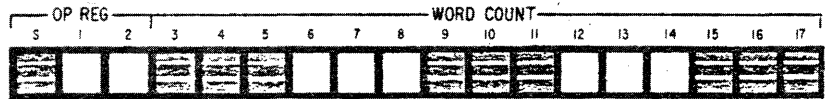
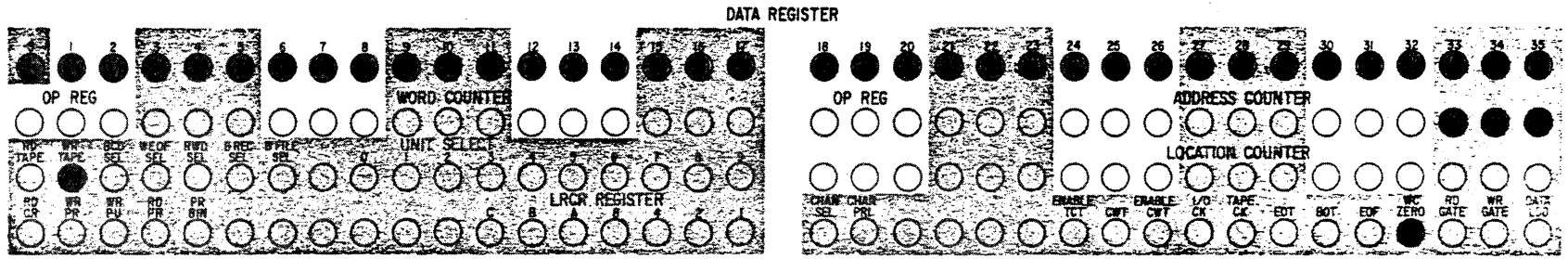
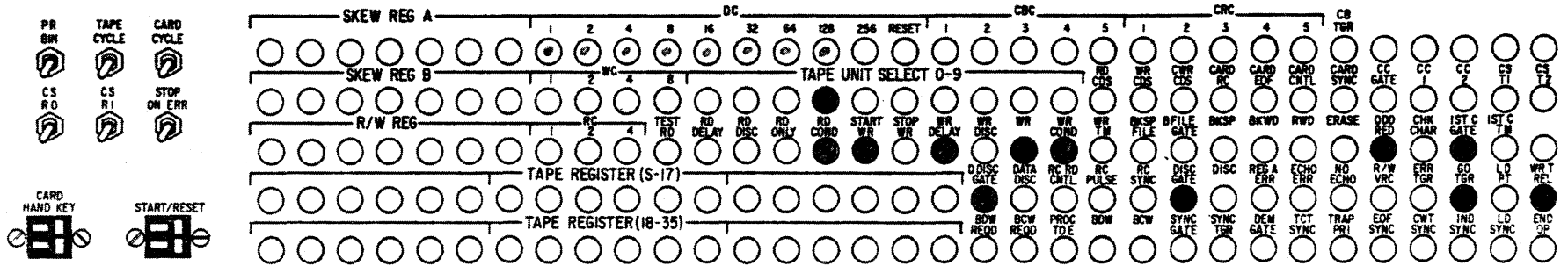
222



Operation :
 Manual On Line
 Load IOCD 0,, 7
 Depress Write Tape

*** Words written were composed of zeros.in storage.

Other Symptoms :
 Tape has stopped
 D. C. is rippling

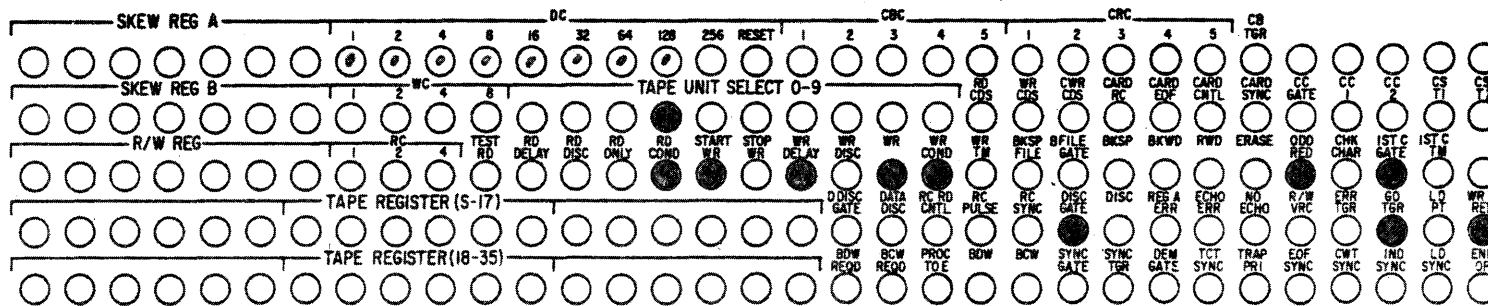
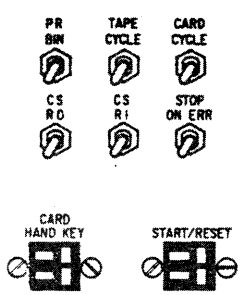


Operation:
 Manual ON LINE
 LOAD IOCD 0, 7
 Depress Write Tape

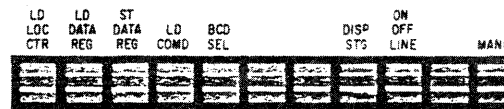
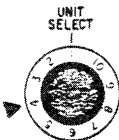
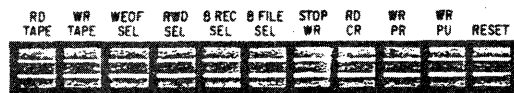
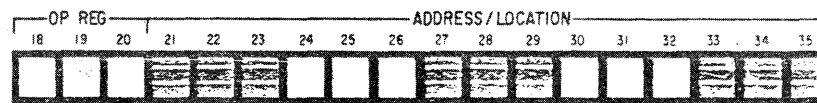
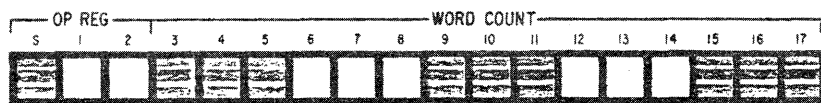
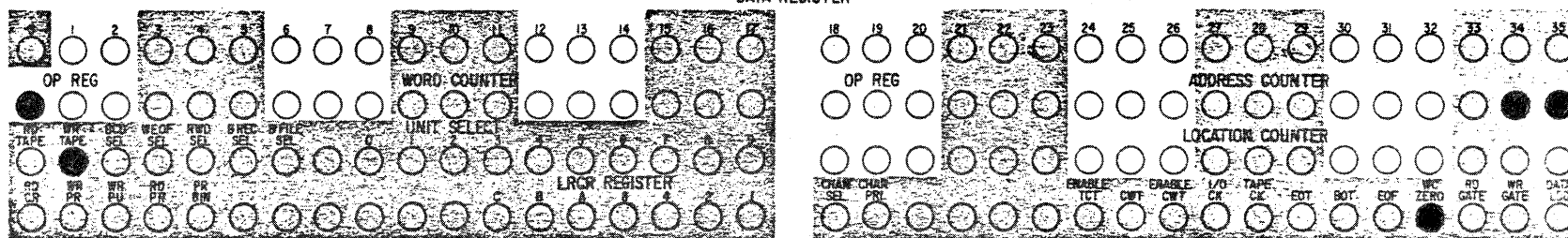
Other Symptoms:
 Tape moving constantly
 D. C. rippling

*Stg. Loc 0-7 contain all ones.

223



DATA REGISTER



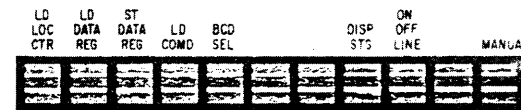
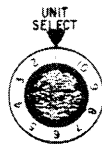
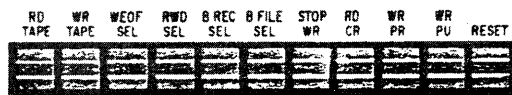
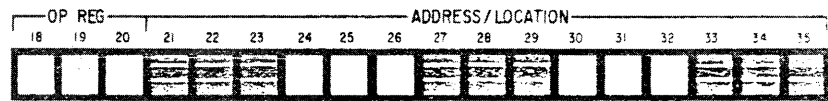
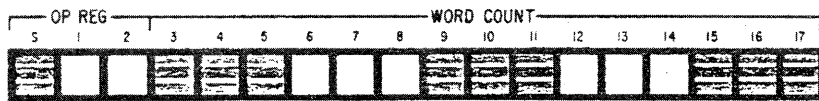
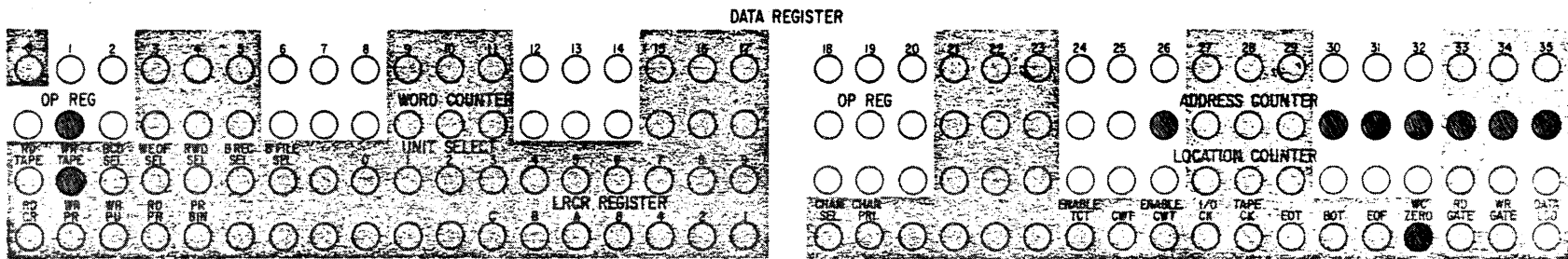
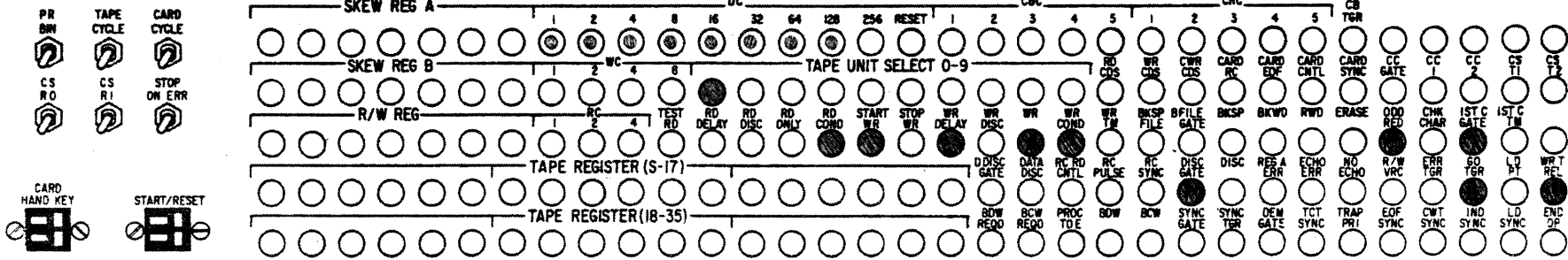
Operation:

- Manual ON LINE
- Load Command - IOCP 0, , 3
- Stored in Loc. 0 IOCD 4, , 3
- Depress Write Tape
- Stg. Loc. 1-7 are blank

Other Symptoms:

- Tape running away
- D. C. rippling

224



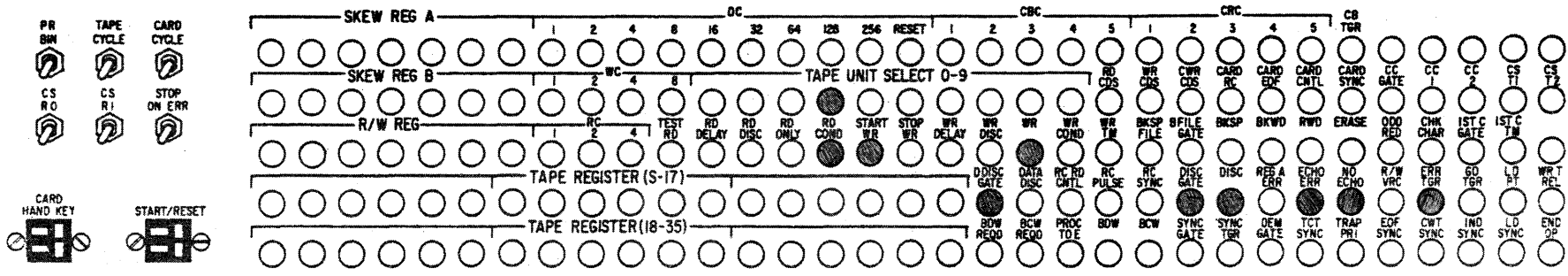
Operation:

- The following program is in storage
 77776 IOCP 100, 0, 2
 77777 IORP 1000, 0, 77
 00000 IOCD 0, 0, 0
- LOAD IOCP 1000, 0, 77 into channel
 LOAD Location counter with 77776
- With manual - on line, press write tape

Other Symptoms:

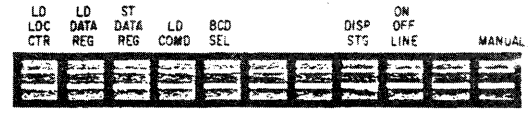
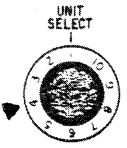
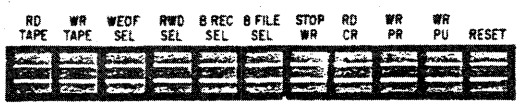
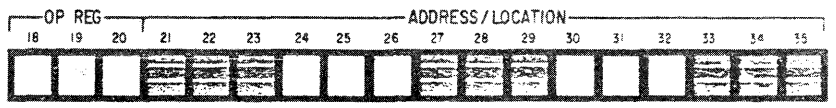
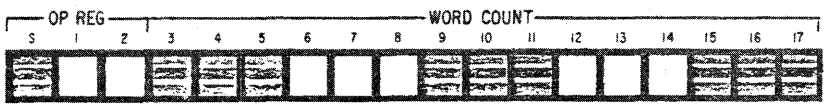
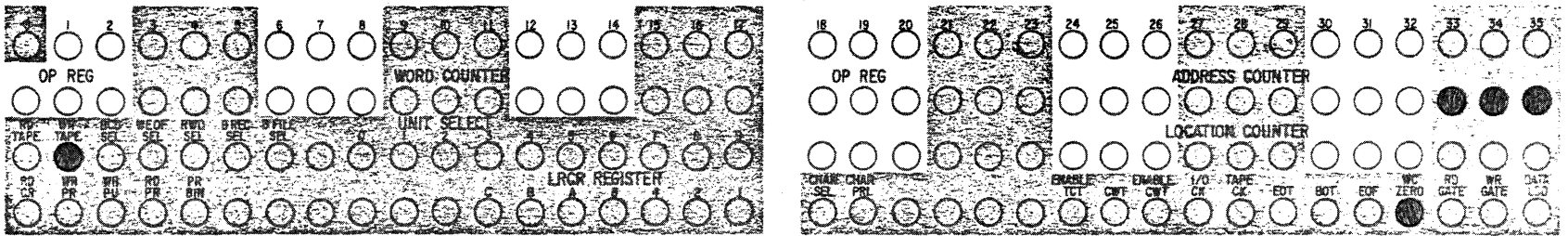
- Tape runs away
- WC Ran
- RC Ran
- DC is rippling

Note: Read operations also fail



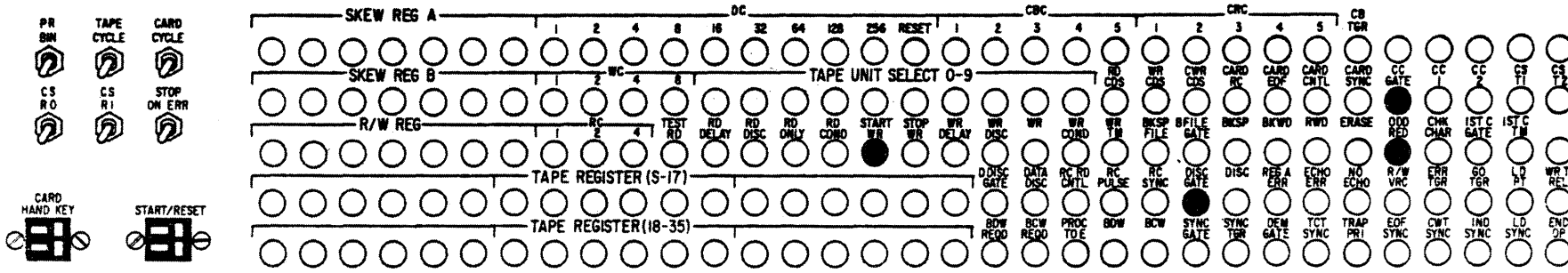
226

DATA REGISTER

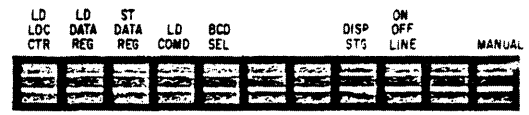
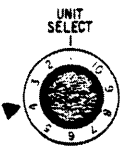
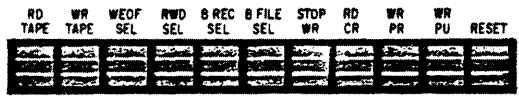
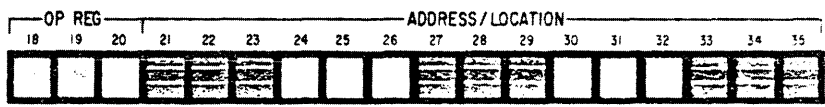
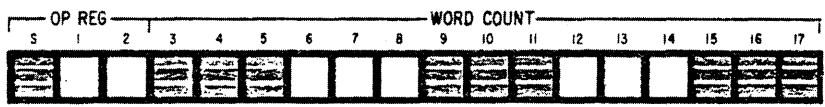
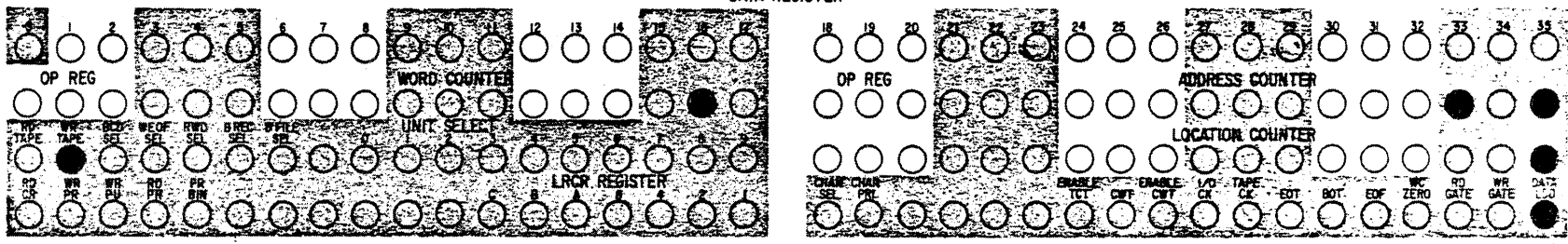


Operation:
 Manual ON LINE
 Load Command IOCD 0, , 7
 Depress Write Tape
 Stg. Loc 0-7 are zeros

Other Symptoms:
 Delay ctr. rippled and stopped
 Wr. clock ran
 Tape stopped



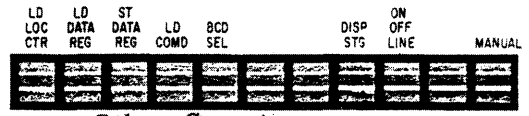
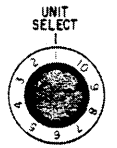
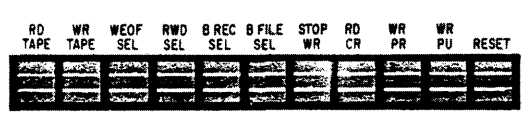
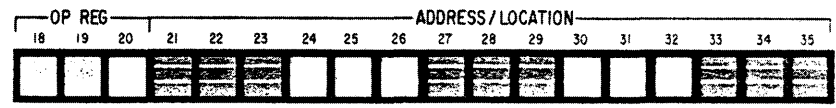
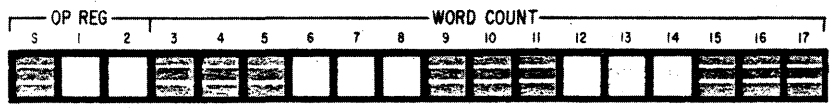
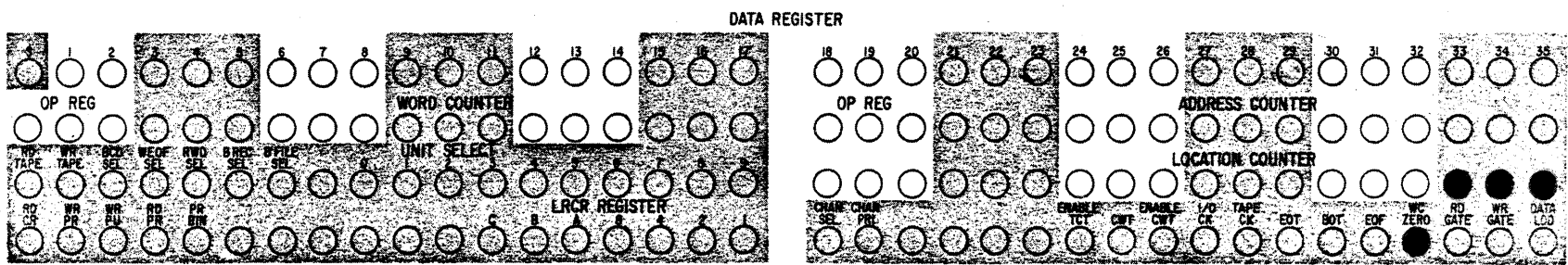
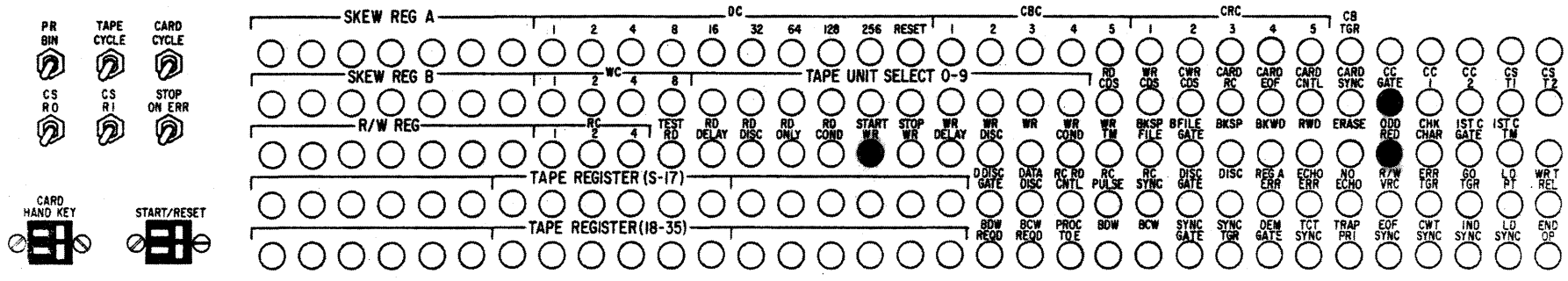
DATA REGISTER



Operation:
 Manual ON LINE
 Load Command IORP 0, , 3
 Stored in Loc 0 IOCD 4, , 3
 Depress Write Tape

Other Symptoms:
 Tape has stopped

227



Operation:

Manual ON LINE

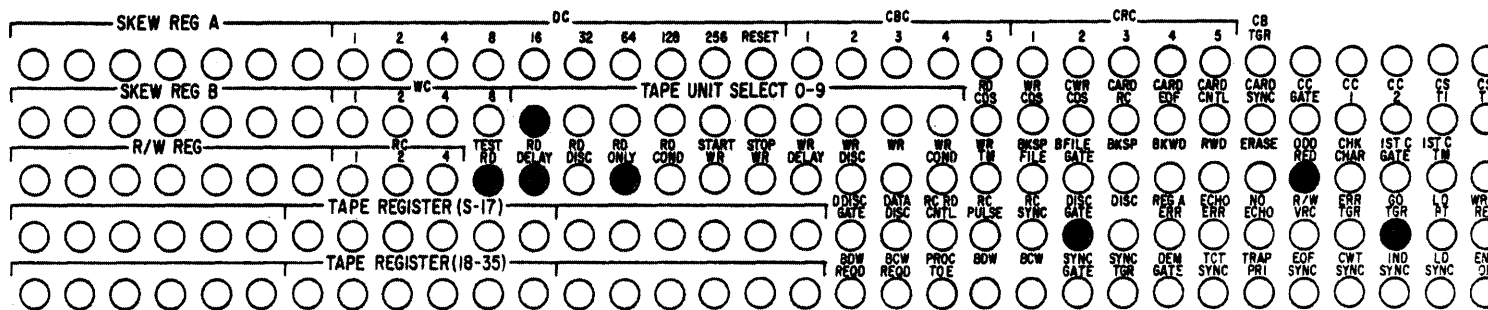
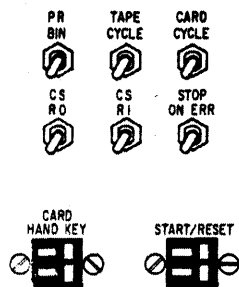
Load Command	IORP	500, , 77777
Stored in Loc	0	IORP 500, , 77777
	1	IORP 500, , 77777
	2	IORP 500, , 77777
	3	IORP 500, , 77777
	4	IORP 500, , 77777
	5	IORP 500, , 77777
	6	IOCD 0, , 0

Depress Write Tape

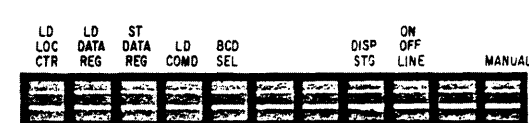
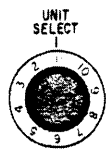
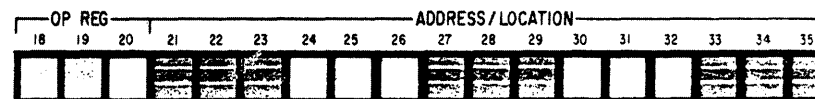
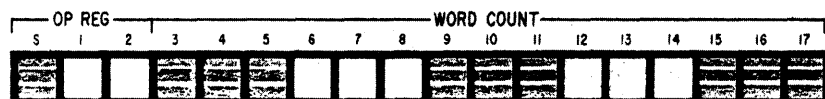
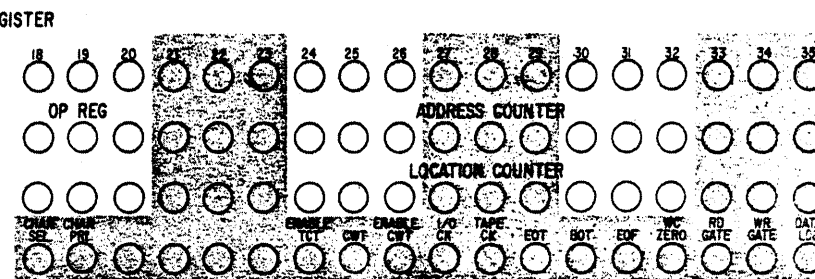
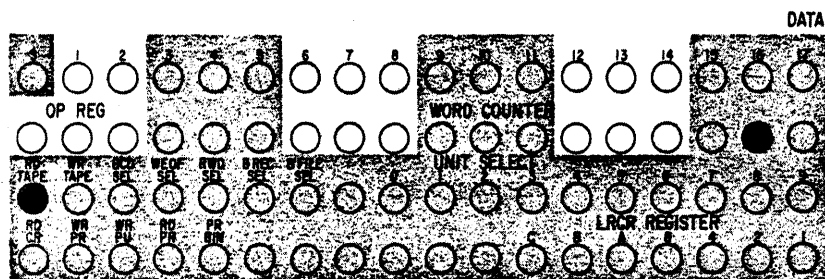
Other Symptoms

1. Further investigation reveals only 2 records on tape.

228



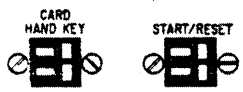
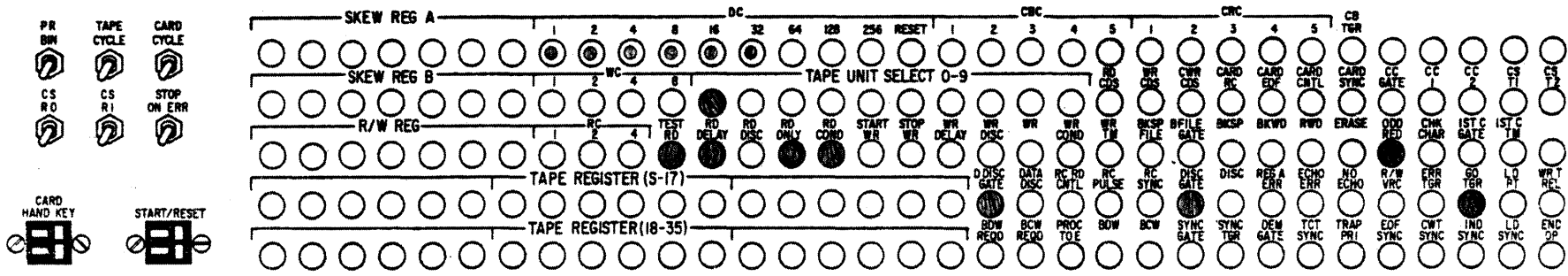
229



Manual ON LINE
Load IOCD 0, , 2
Depress READ TAPE

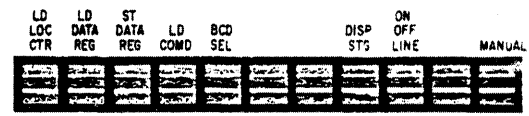
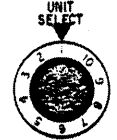
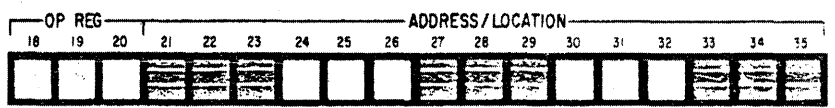
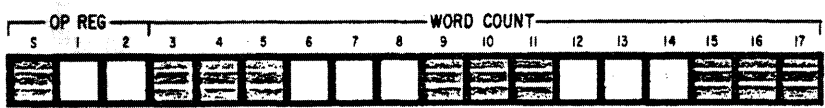
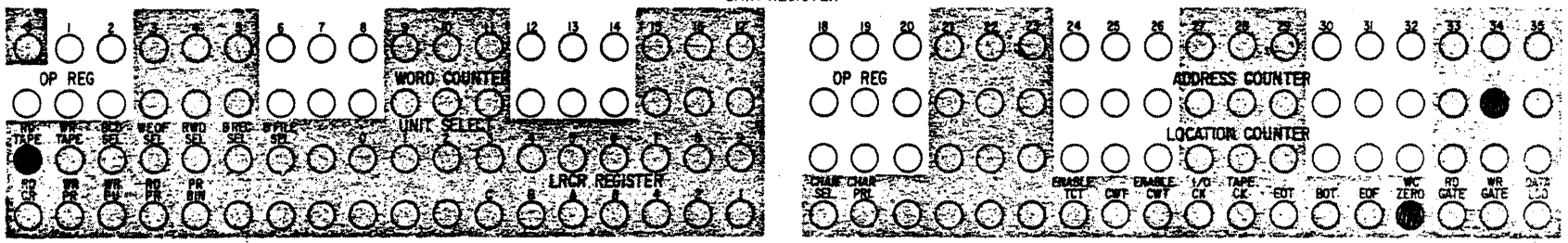
Other Symptoms
1. Tape runs away
2. D. C. is rippling

There are many 5 word records of all bits on tape.



230

DATA REGISTER



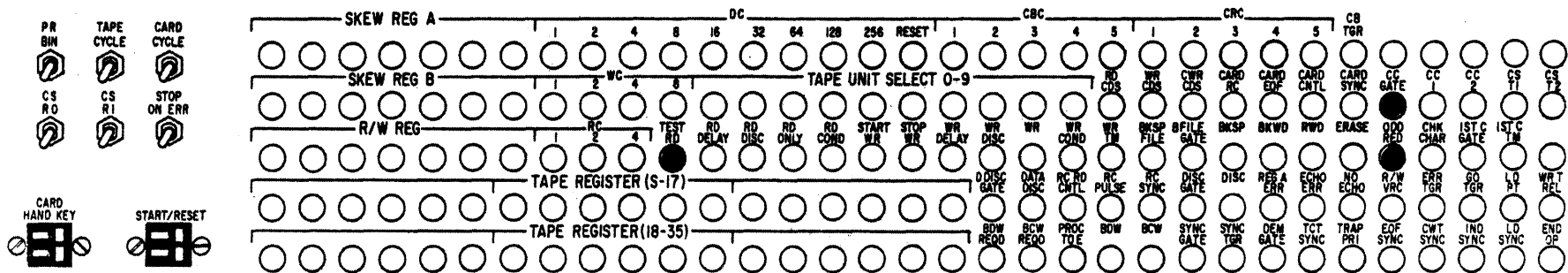
Operation:

Manual ON LINE
 Load Command IOCD 0, , 2
 Depress Read Tape

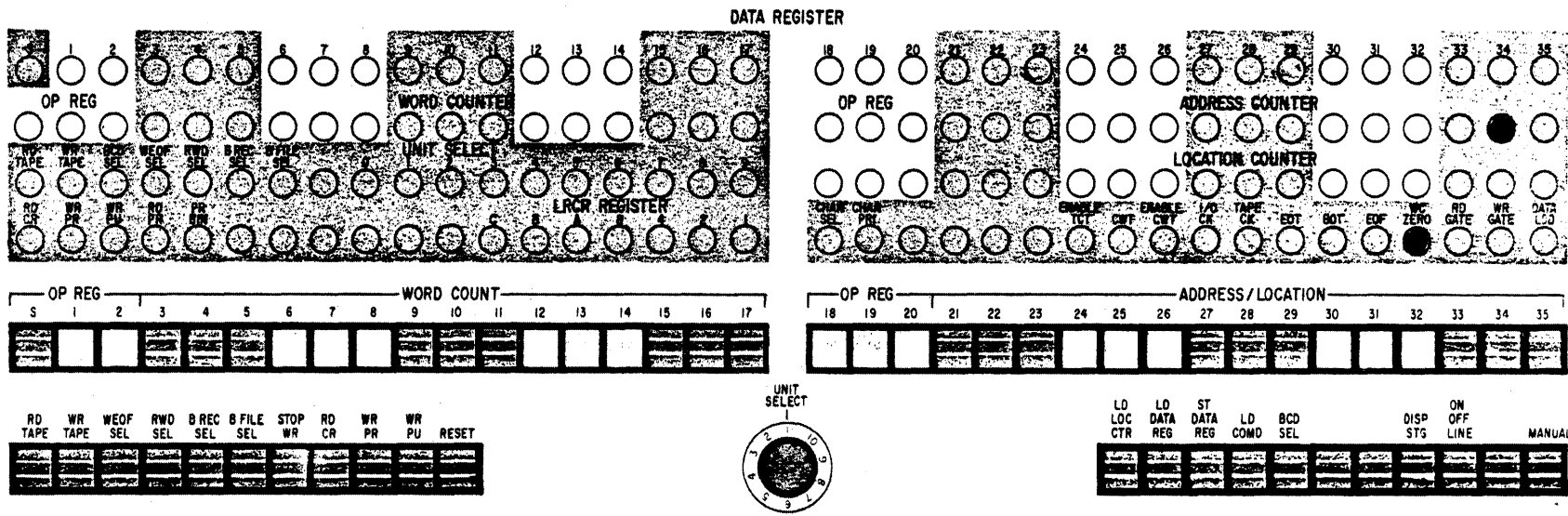
*There are many 5 word records on tape.

Other Symptoms:

D. C. Rippling
 Tape running away
 There is nothing in Storage
 Location 0 and 1.



231



Manual On Line
Load IOCD 0, , 2
Depress Read Tape

Other Symptoms :

1. Loc 0 contains the last word of the 1st record.
2. Loc 1 contains the last word of the 2nd record.

* There are many 5 word records on tape.

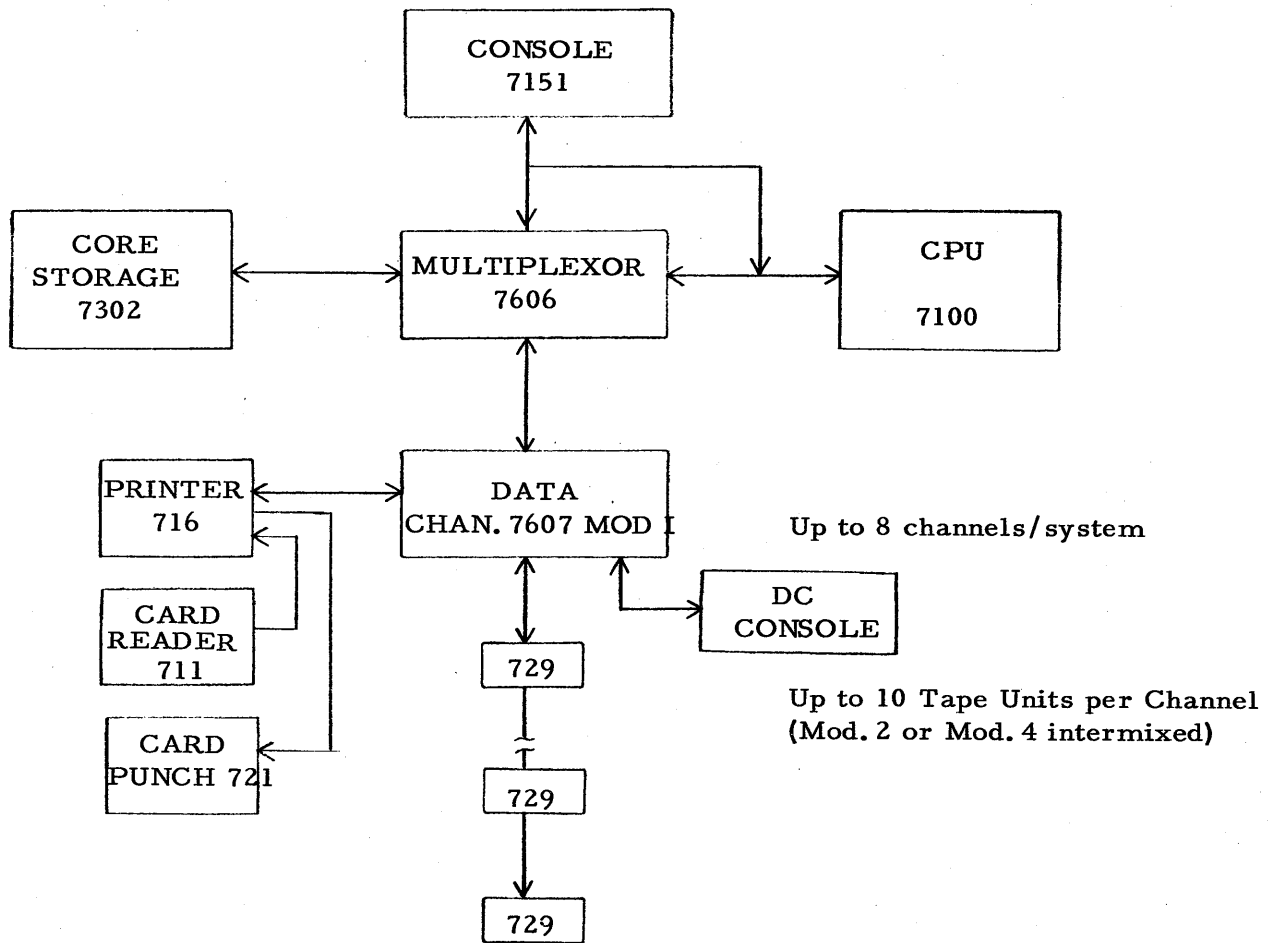


FIG. 01.01

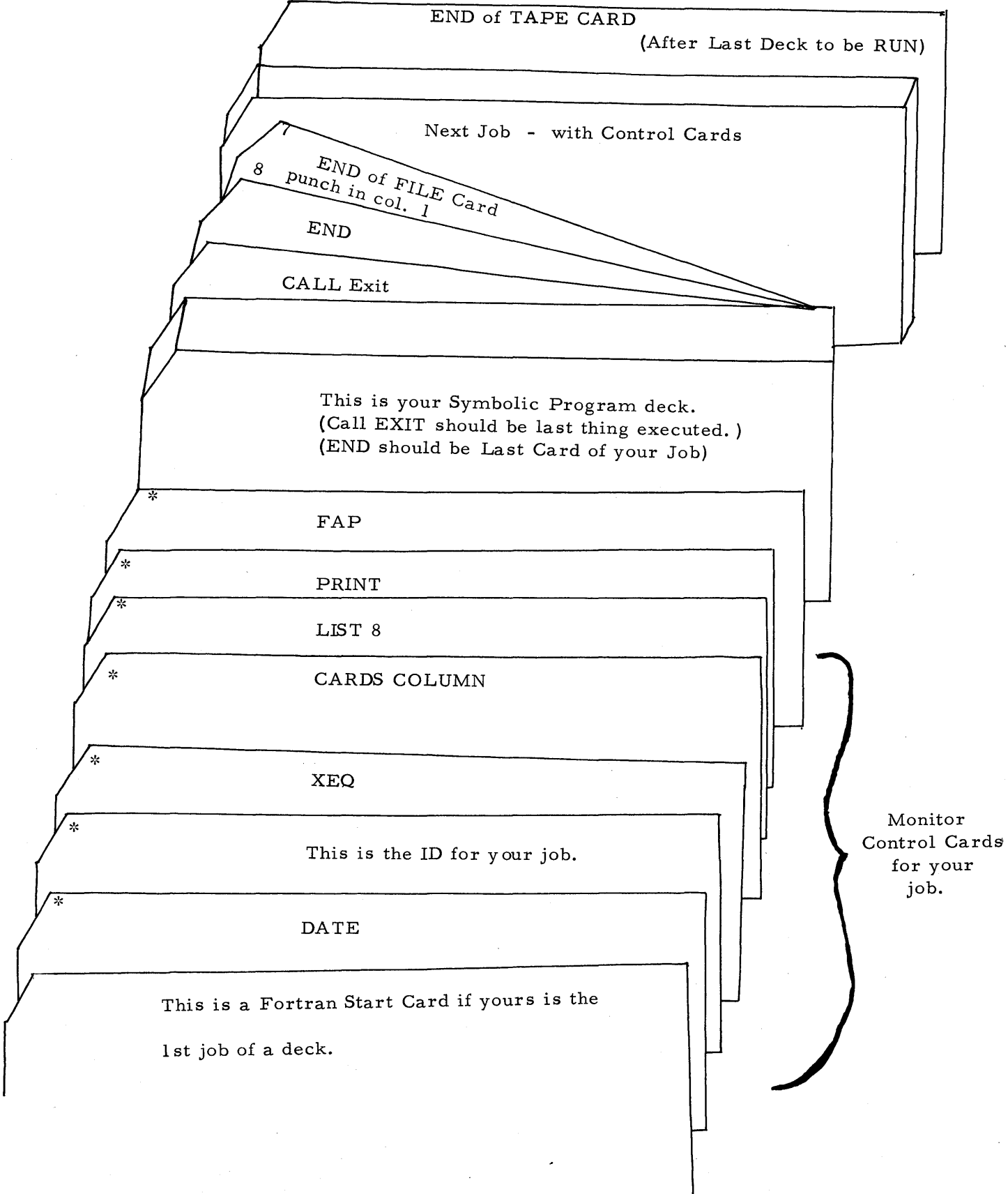


Fig. 01.02

Problem				MONITOR CONTROL CARD SEQUENCE					
Coder		THE LITTLE OLE WINE MAKER		Date	Page	of			
Location		Operation		Address, Tag, Decrement/Count		Comments		Identification	
1	2	6	7	8		72	73	80	
	?	FORTRAN START CARD SHOULD PRECEED THE FIRST JOB							
*		DATE							
*		THIS IS THE I. D.		FOR THIS JOB.	(YOUR NAME, YOUR JOB NAME)				
*		XEQ			(USE TO EXECUTE YOUR JOB IF DESIRED)				
*		CARDS COLUMN			(USE TO GET ON LINE COL. BIN. OBJECT DECK)				
*		LIST 8			(USE TO GET 2 COL. LISTING WITH OCTAL AND RELOC. BITS)				
*		PRINT			(USE TO PRINT ON LINE)				
*		FAP			(TELLS MONITOR TO CALL IN FAP AND GIVE IT CONTROL)				
		}			(YOUR SYMBOLIC PROGRAM CARDS)				
		CALL EXIT			(CALLS FOR MONITOR CONTROL WHEN JOB HAS FINISHED)				
		END			(TELLS MONITOR THIS IS LAST CARD TO READ FOR JOB)				
		END OF FILE (7 & 8 IN COL.		ONE)	SHOULD FOLLOW YOUR JOB ON CARD INPUT				
		END OF TAPE CARD SHOULD		FOLLOW	LAST JOB.				
					(Typical layout for object deck to be punched on line in col. bin. which can be later loaded and run under monitor control.)				

234

Fig. 01.03

Problem				MONITOR CONTROL CARD SEQUENCE				
Coder		THE PHANTOM		Date		Page of		
Location		Operation		Address, Tag, Decrement/Count		Comments		
1	2	6	7	8	72	73	80	
FORTTRAN START CARD SHOULD PRECEED THE FIRST JOB								
*		DATE						
*		THIS IS THE I. D.		CARD FOR THIS JOB.				
*		XEQ					(EXECUTE JOB)	
*		CARDS ROW					(ROW BINARY OBJECT PUNCHED ON LINE)	
*		LIST 8						
*		PRINT						
*		FAP						
		ABS	}				(ABSOLUTE OBJECT PROGRAM DECK)	
		ORG		1000				(BEGIN PROG. IN ACTUAL ADDRESS 1000)
	START							(YOUR SYMBOLIC PROG.)
		CALL EXIT						
		END		START			(WITH ABSOLUTE, VARIABLE FIELD OF "END" MAY HAVE TRANSFER TO ADDRESS... START IS EXAMPLE)	
							(END OF FILE CARD should follow each JOB)	
							(END OF TAPE CARD should follow last JOB)	
							(Typical layout for row binary object prog. to be loaded by SE9AP 9 LEFT loader and run WITHOUT monitor.)	

235

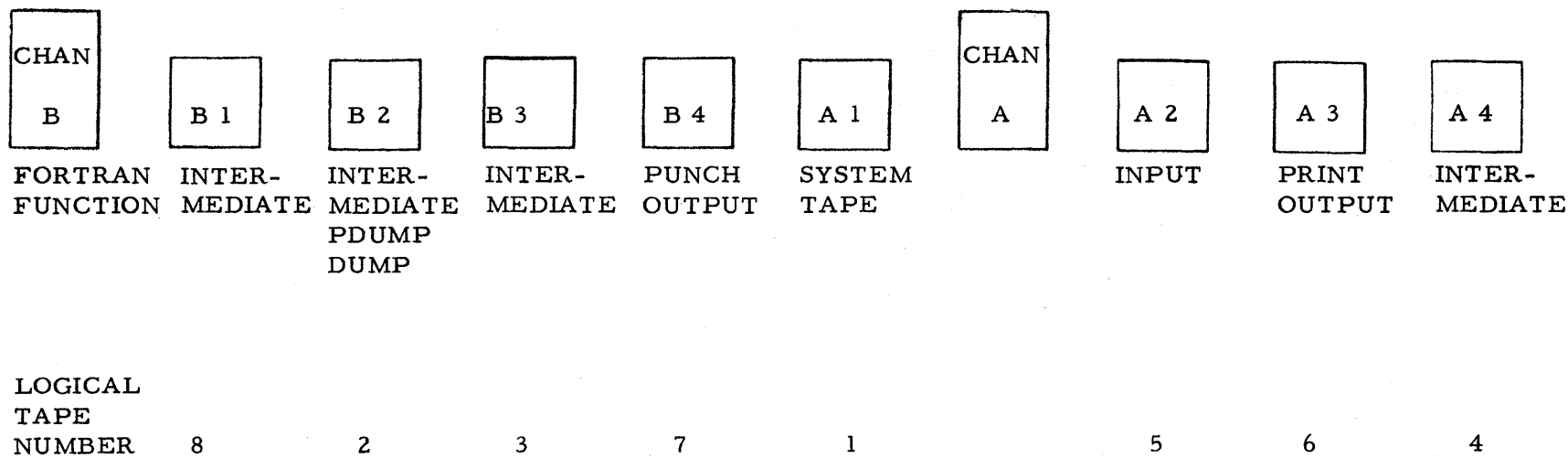
Fig. 01.04

DUMMY	SOURCE ERROR	MACH ERROR	LIBRY SEARCH	BSS CNTL	DEBUG	SCAN	FAP II	FAP I	SIGN ON	DUMP	C-T SIM.	IOP	
12	11	10	9	8	7	6	5	4	3	2	1	0	

	E O F	GENERAL DIAGNOSTIC	E O F	LIBRARY	E O F	FORTRAN COMPILER RECORDS 13 - 34	E O F	MONITOR RECORDS 0 - 12	
--	-------------	-----------------------	-------------	---------	-------------	--	-------------	------------------------------	--

FORTRAN SYSTEM TAPE

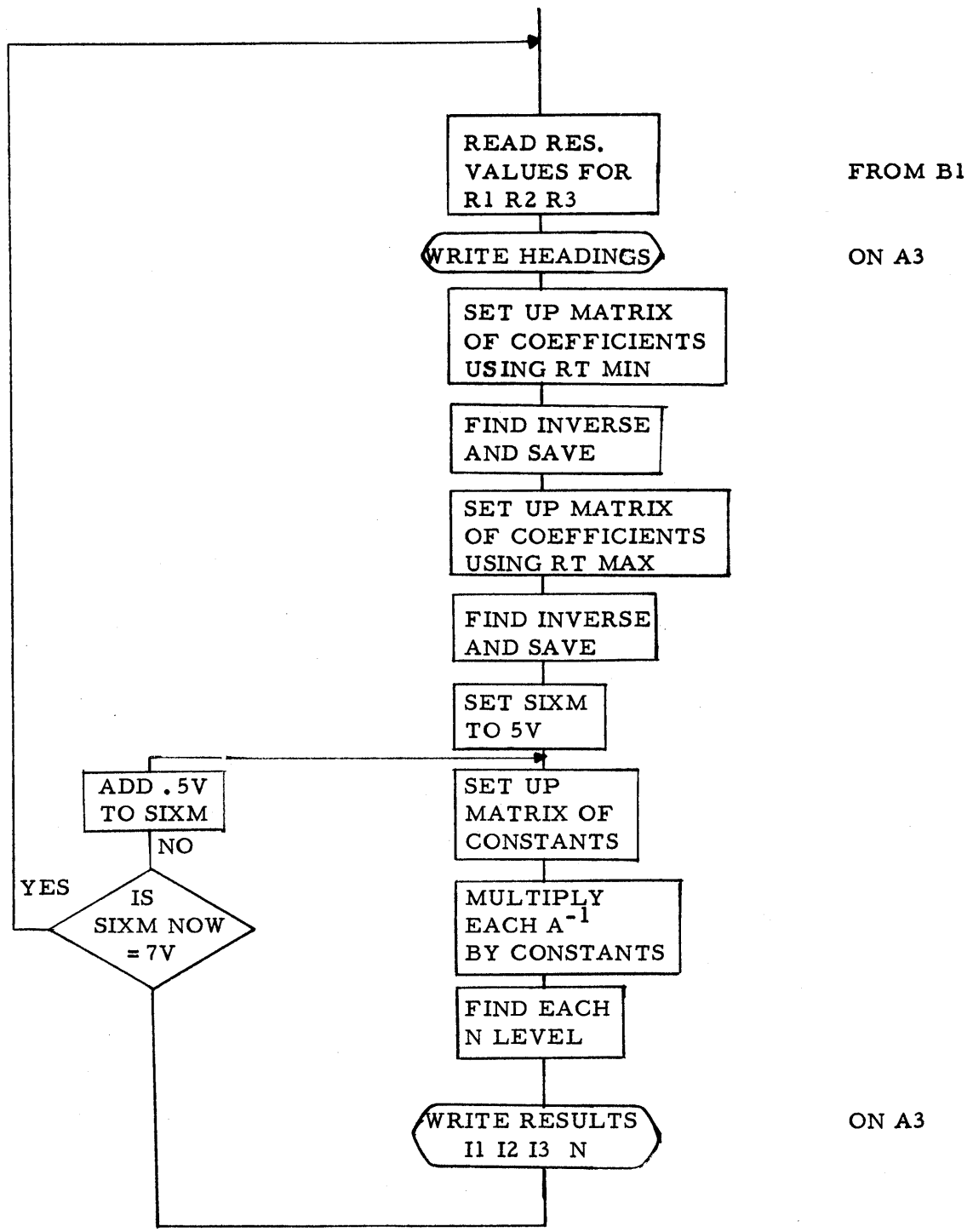
Figure 01.05



TAPE ASSIGNMENTS
 FORTRAN MONITOR OPERATION

1	567	72
* * THIS * * * * * C THIS	DATE 5/24/62 IS THE I D CARD XEQ LIST CARDS ROW LIBE IS A COMMENT CARD AND WILL ALSO SERVE AS A PAGE HEADING CARD	
	REWIND 8 CALL EXIT END	

Figure 01.07
238

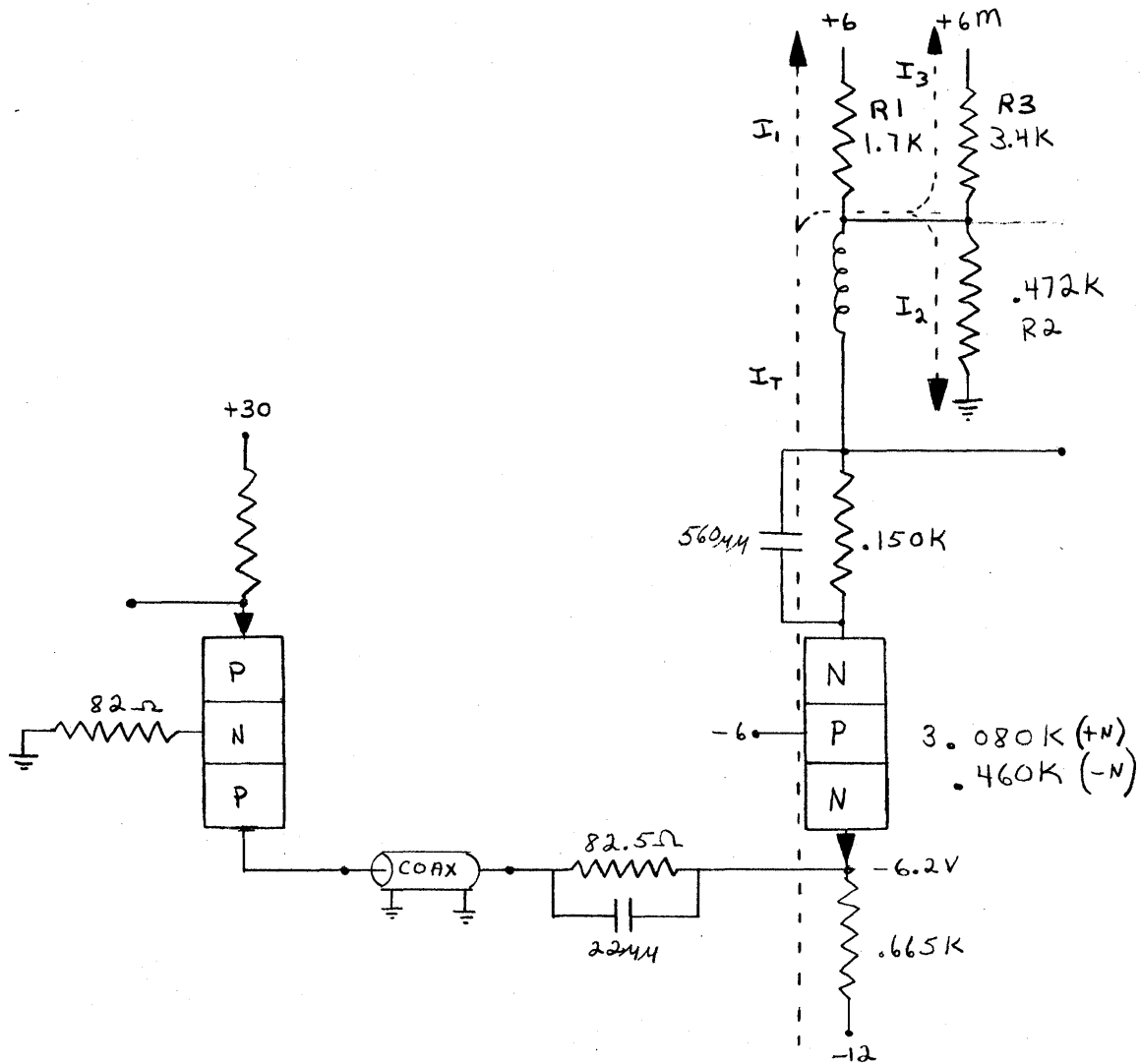


FROM B1

ON A3

ON A3

Figure 01.08
239



DT

Figure 01.09

* THIS IS AN EXAMPLE OF A LIBRARY FUNCTION

* LIST

A = 4.

B = 9.

*

C = SQRTF(A) + SQRTF(B)

*

END (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

00000 (FPT)	BCD 1(FPT)
00001 SQRT	BCD 1SQRT
00002 \$\$	CLA (FPT)
00003	STO 8
00004	STZ 4)-205
00005 2A	CLA 3)
00006	STO A
00007 3A	CLA 3)+1
00010	STO B
00011 4A	CLA B
00012	TSX SQRT, 4
00013	STO 1)+1
00014	CLA A
00015	TSX SQRT, 4
00016	FAD 1)+1
00017	STO C
00020	RCD
00021	RLA *+4
00022	LCA
00023	TAF *+3
00024	TRA 1
00025	MON 0, 0, 3
00026	IOT
00027 2047D1	HPR 1, 7
00030	TRA 2047D1
00031 3)	OCT + 203400000000
00032	OCT +204440000000'
00033 6)	OCT +233000000000
00034	OCT +000000377777
00035	OCT +000000000000
00036	OCT +000001000000
00037	OCT +000000000000

* THIS IS AN EXAMPLE OF A BUILT - IN FUNCTION

* LIST

A = -2.

B = ABSF(A)

*

*

K = -2

L = XABSF(K)

*

END (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

00000 (FPT)	BCD 1(FPT)
00001 \$\$	CLA (FPT)
00002	STO 8
00003	STZ 4)-205
00004 2A	CLS 3)
00005	STO A
00006 3A	CLA A
00007	SSP
00010	STO B
00011 4A	CLS 2)
00012	STO K
00013 5A	CLA K
00014	SSP
00015	STO L
00016	RCD
00017	RLA *+4
00020	LCA
00021	TAF *+3
00022	TRA 1
00023	MON 0, 0, 3
00024	IOT
00025 2047D1	HPR 1, 7
00026	TRA 2047D1
00027 2)	OCT +000002000000
00030 3)	OCT +202400000000
00031 6)	OCT +233000000000
00032	OCT +000000377777
00033	OCT +000000000000
00034	OCT +000001000000
00035	OCT +000000000000

THIS IS AN EXAMPLE OF A SELF - DEFINED FUNCTION

```

*      LIST

      MYFUNC (A, B, C) = A+B+C

*

      X = 2.
      Y = 3.
      Z = 4.

*

      TOTAL = MYFUNC (X, Y, Z)

*

      SUM = MYFUNC (X, Y, Z)

*

      END (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
    
```

```

00000 (FPT)      BCD 1(FPT)
00001 $$        CLA (FPT)
00002           STO 8
00003           STZ 4)-205
00004 3A        CLA 3)
00005           STO X
00006 4A        CLA 3)+1
00007           STO Y
00010 5A        CLA 3)+2
00011           STO Z
00012 6A        CLA X
00013           STO 4)1
00014           LDQ Y
00015           STQ 4)1+1
00016           CLA Z
00017           STO 4)1+2
00020           TSX MYFUNC, 4
00021           STO TOTAL
00022 7A        CLA X
00023           STO 4)1
    
```

```

00024           LDQ Y
00025           STQ 4)1+1
00026           CLA Z
00027           STO 4)1+2
00030           TSX MYFUNC, 4
00031           STO SUM
00032           RCD
00033           RLA*+4
00034           LCA
00035           TAF *+3
00036           TRA 1
00037           MON 0, 0, 3
00040           IOT
00041 2047D1    HPR 1, 7
00042           TRA 2047D1
00043 MYFUNC    CLA 4)1
00044           FAD 4)1+1
00045           FAD 4)1+2
00046           TRA 1, 4
00047 3)        OCT +202400000000
00050           OCT +202600000000
00051           OCT +203400000000
00052 6)        OCT +233000000000
00053           OCT +000000377777
00054           OCT +000000000000
00055           OCT +000001000000
00056           OCT +000000000000
    
```

Figure 01.12
243

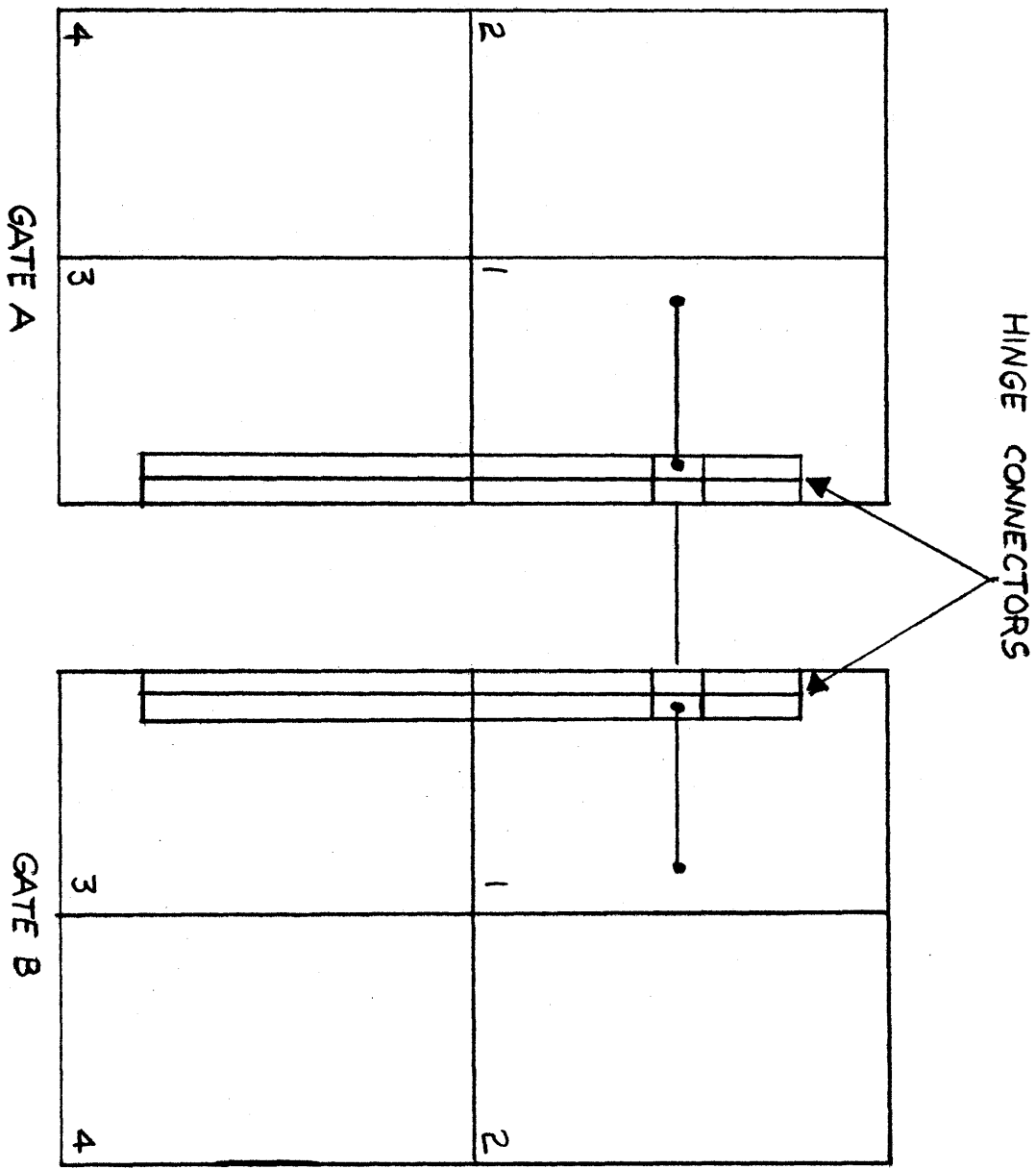


Figure 02.01
244

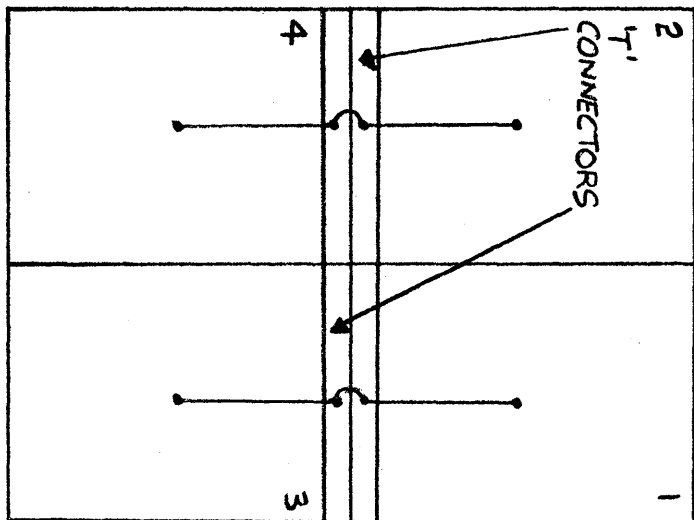
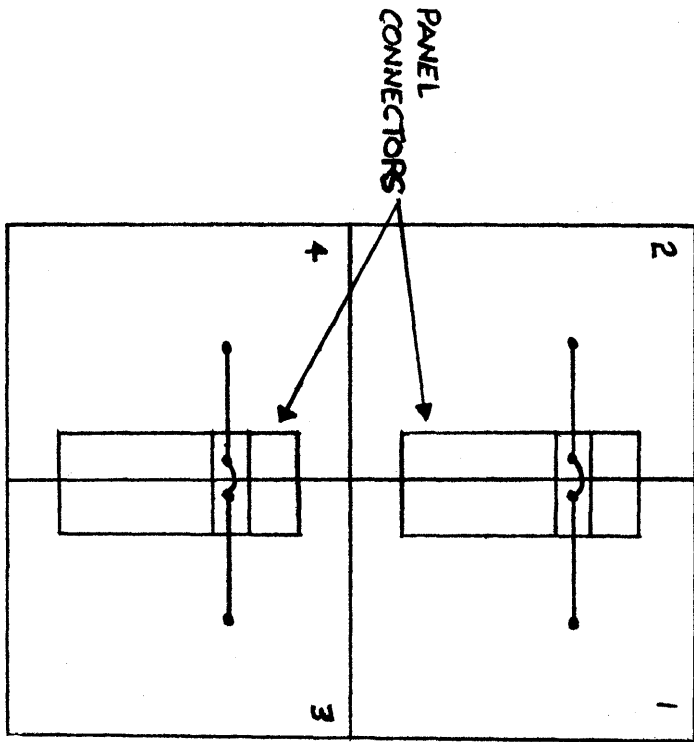


Figure 02.02
245

CONNECTIONS BETWEEN TOWERS OF A FRAME

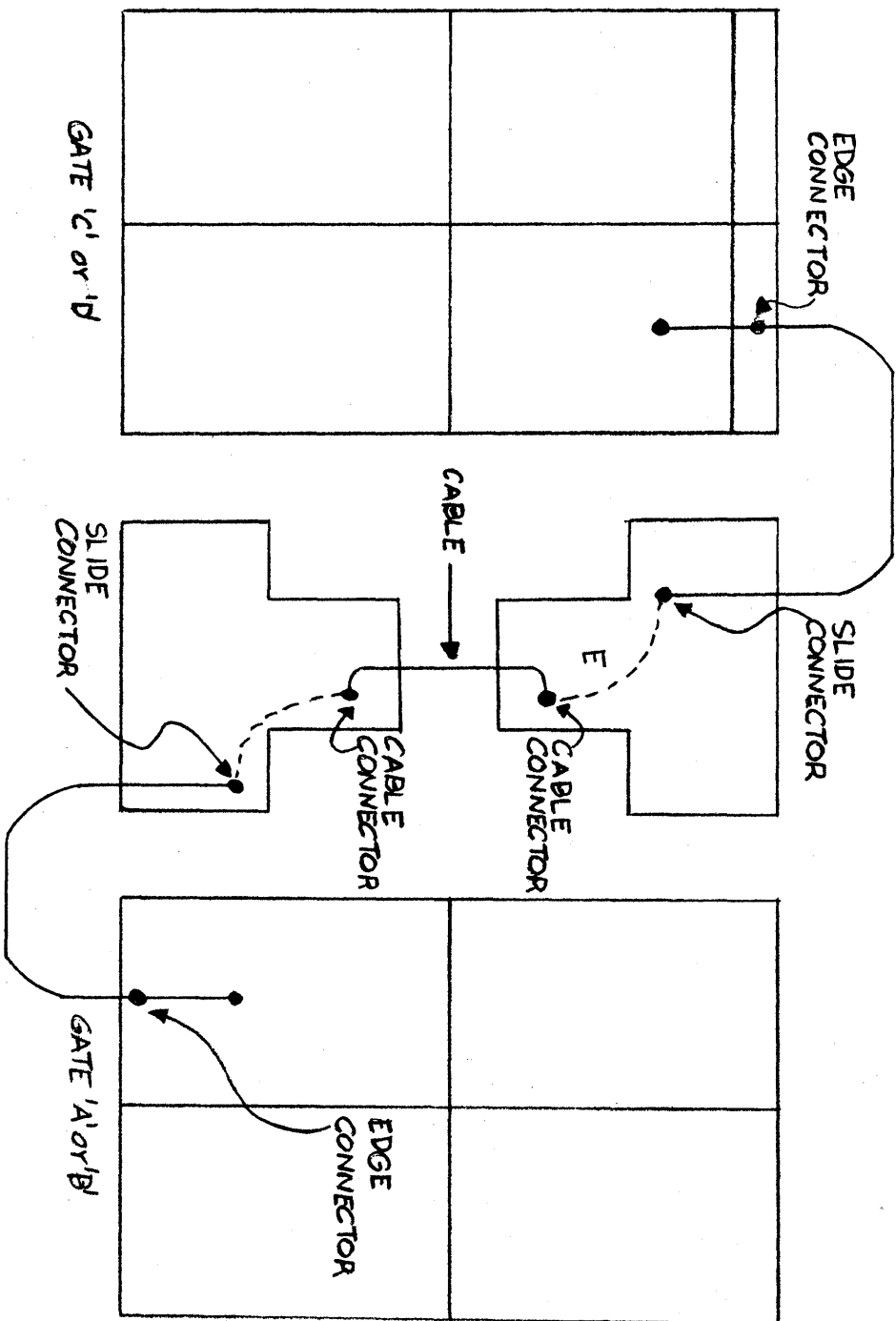


Figure 02.03
246

FRAME TO FRAME

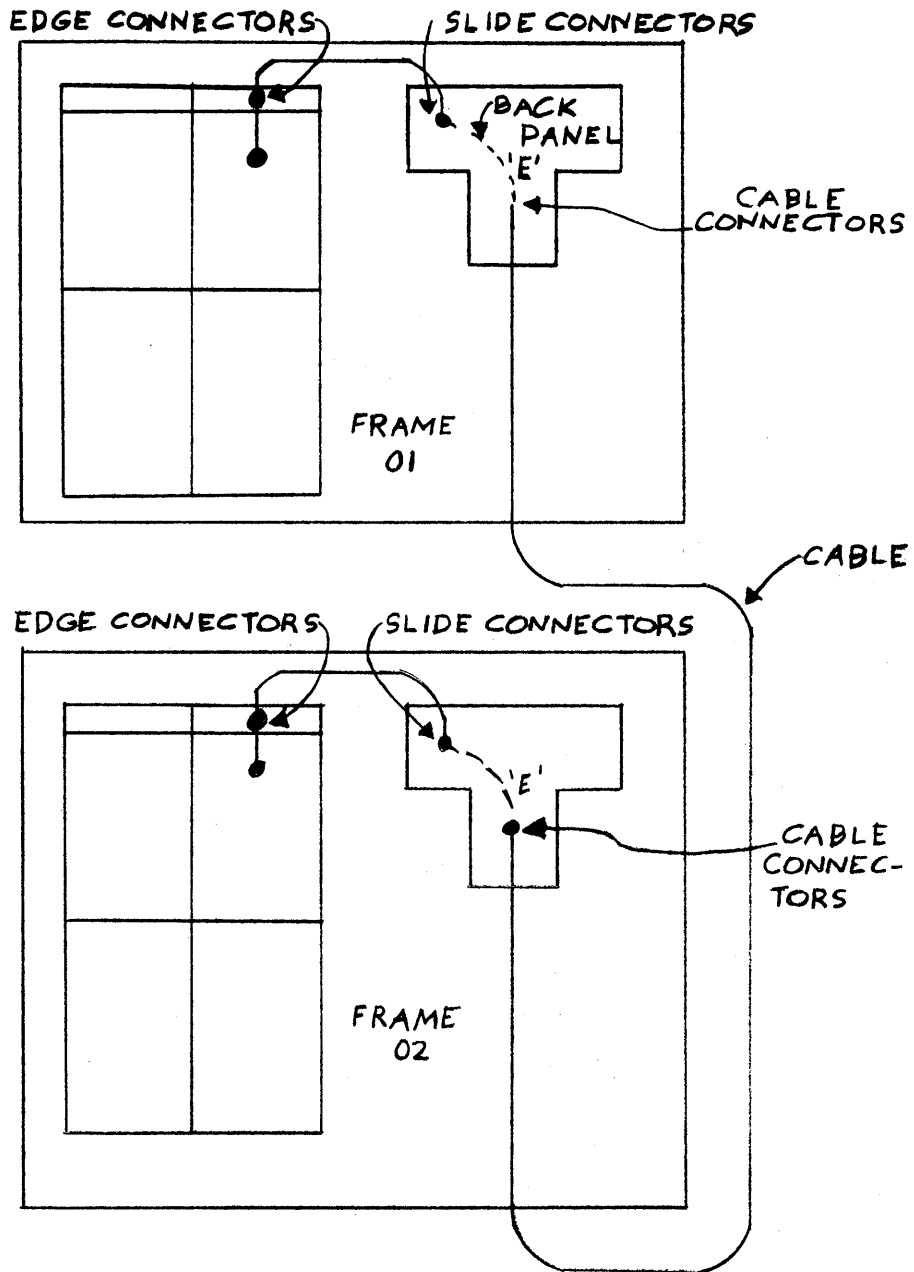


Figure 02.04
247

APPLICATION	DRIVER			TERMINATOR			TYPE OF CONDUCTOR
	Functional Name	Card Type	Comments	Functional Name	Card Type	Comments	
Where timing, noise, and pulse shape are not critical problems	-	-	Any logic block without a coupling network	R	AK--	Provides coupling network	Straight Wire
Where many nearby circuits are to be driven	DE	AEWY AEWZ	Alloy N-N Alloy P-P Current input usually from C block			Drives bases of convert blocks (maximum of 20)	Straight Wire
	DP	DMYQ	Drift N-N Drift P-P Current input usually from C block	-	-	Drives bases of convert blocks (from 4-10 bases) not widely separated	Straight Wire
		DNYL DNYJ	Drift N-N Drift P-P Current input usually from C block. Has 4 output pins and load should be balanced to prevent signal skew	-	-	Drives bases of convert blocks (from 11-40 bases) not widely separated	Straight Wire
To drive one critical circuit	-	-	Logic block without a coupling network (must not drive other circuits)	DT	DL-- DK-- DH-- DJ--	N-N input at -3V P-P input at -3V N-P input at 0V P-N input at -6V	93 OHM Coax
To drive many critical circuits at random distances (long lines)	DL	EF--	N-N may feed up	CBT	DEYR DFYR	N-P terminates OL P-N buffer between DL&local logic	93 OHM Coax
		EG--	P-P to 5 circuits				
		KS-- KT--	P-P may feed up N-N to 10 circuits				

Figure 02.05
248

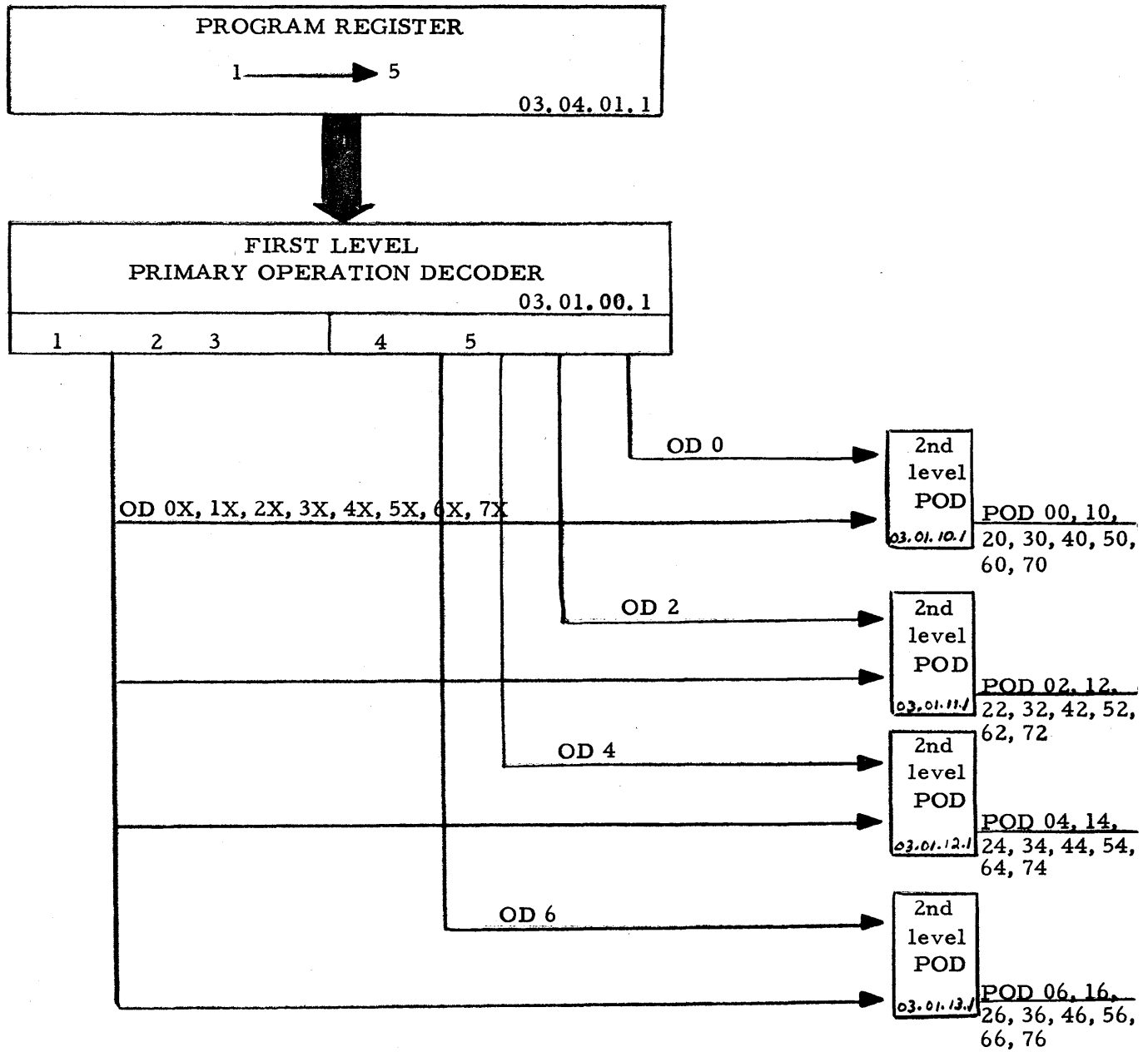


Figure 02.06
249

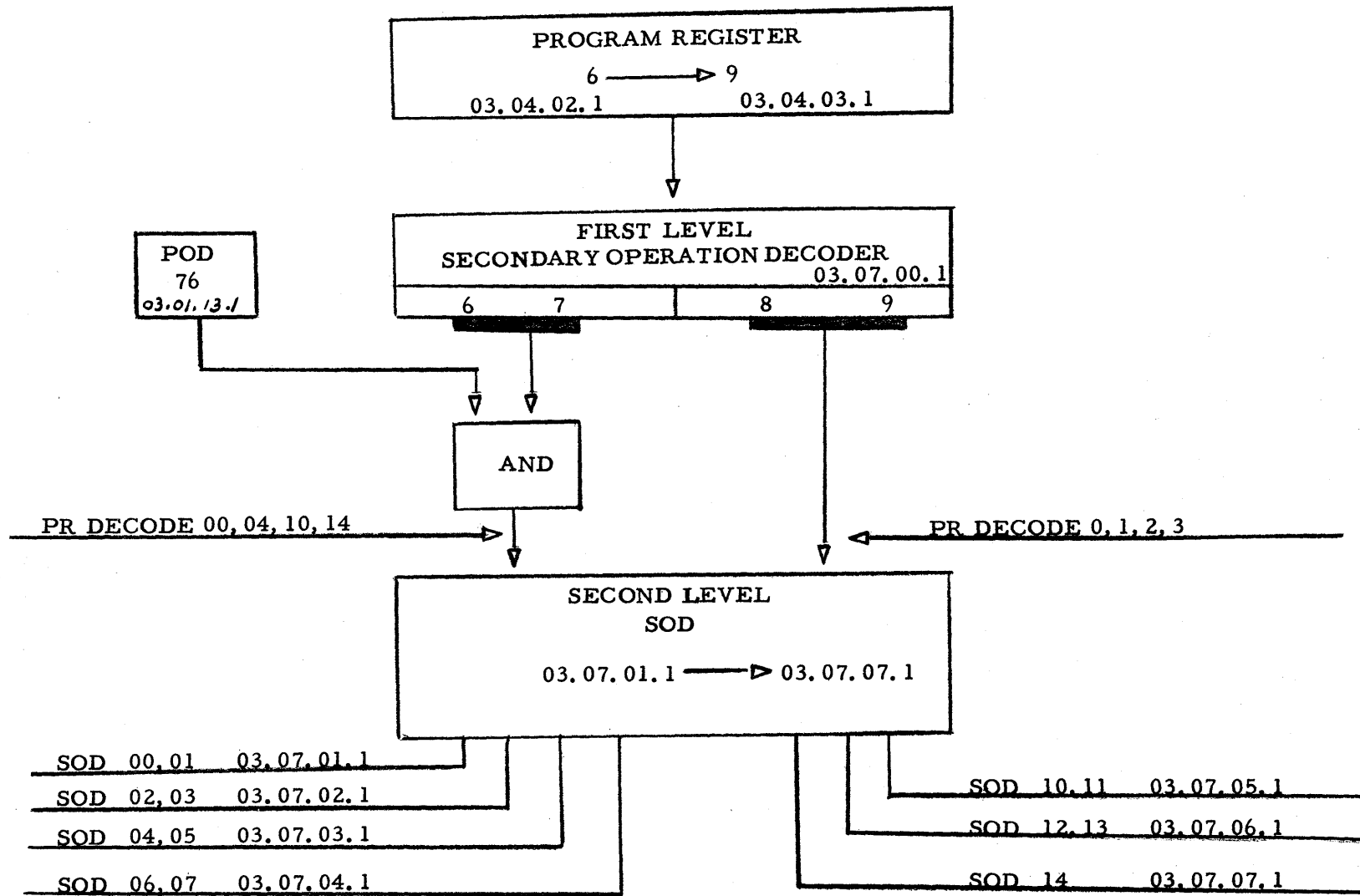
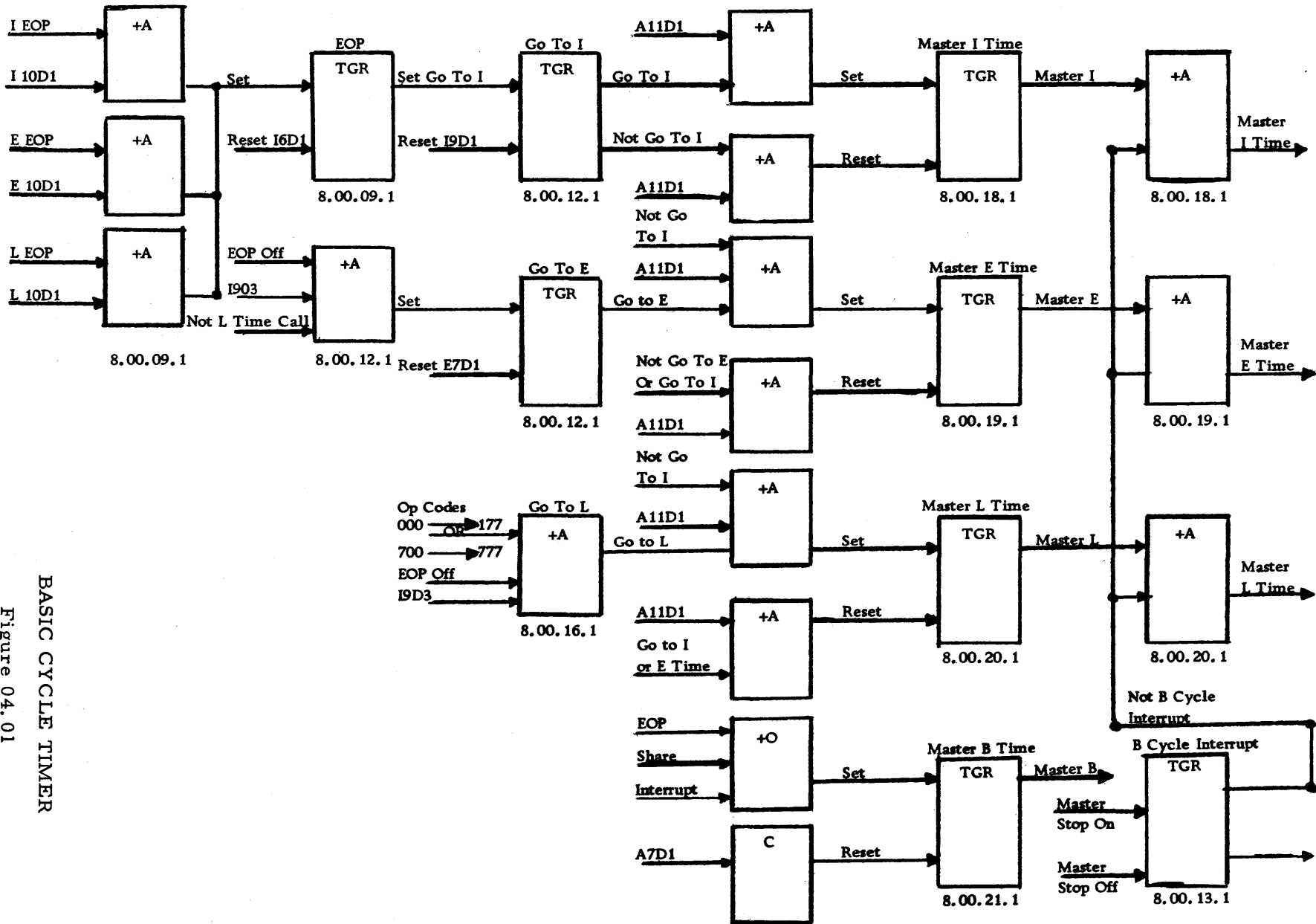
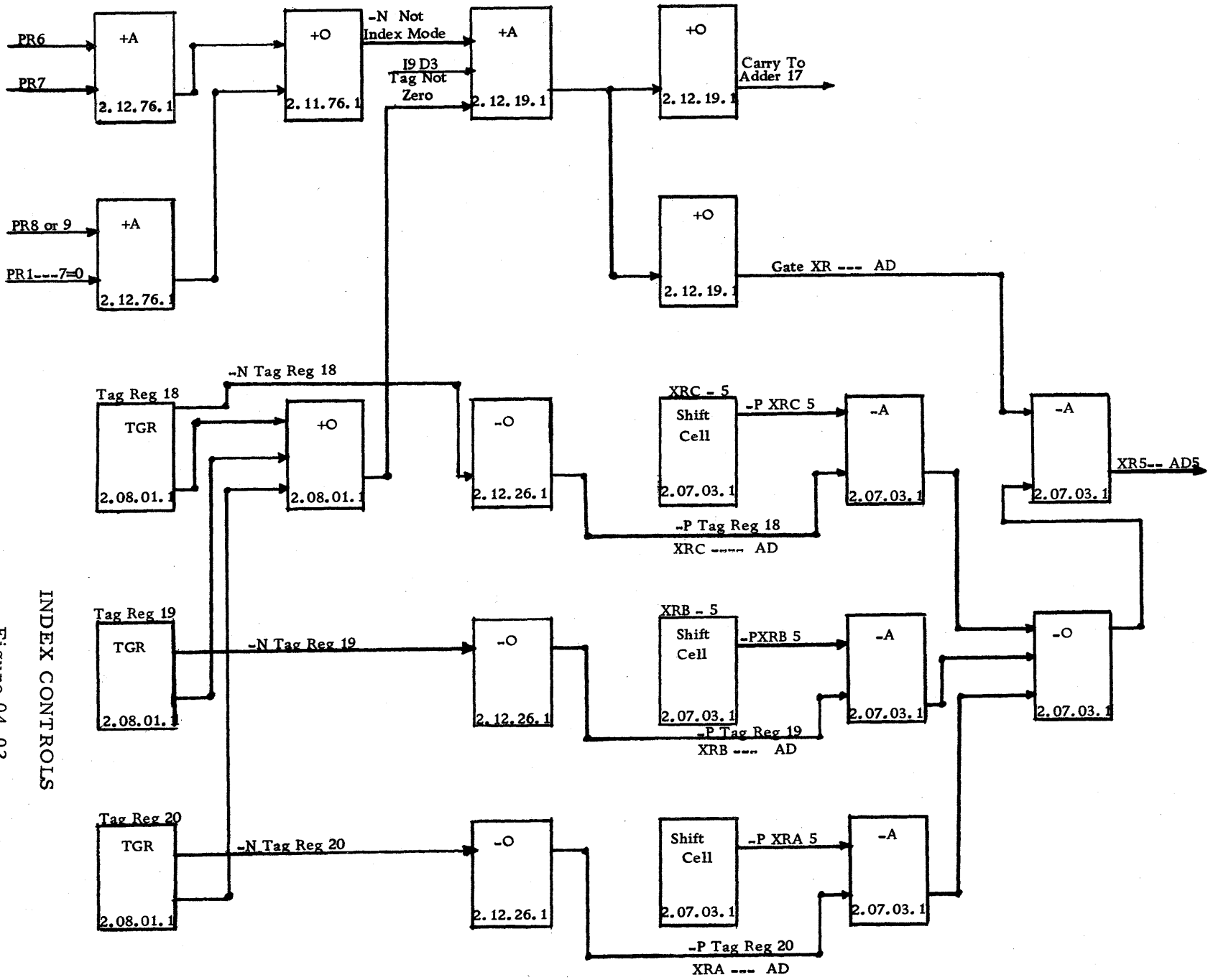


Figure 02.07

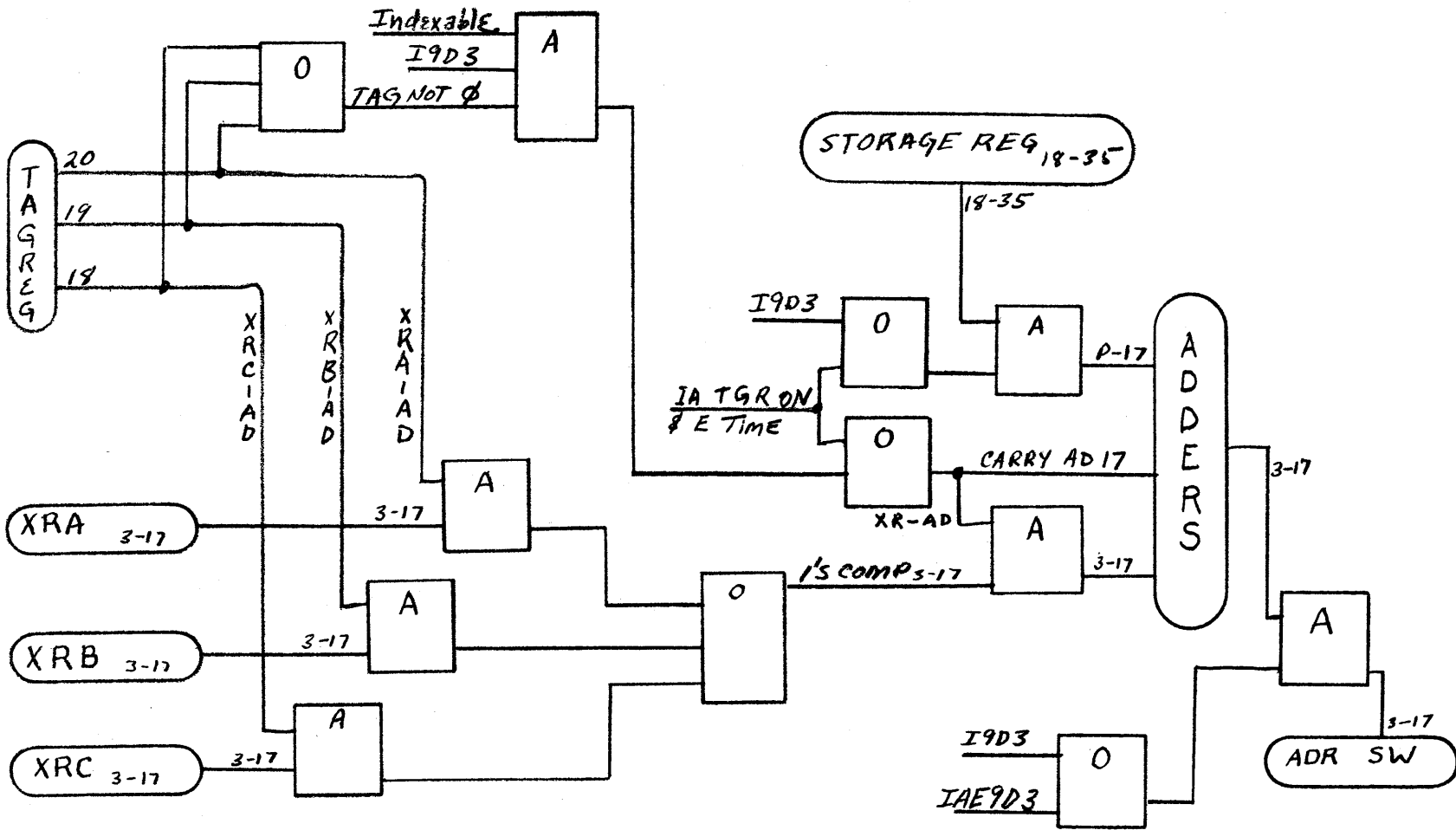


BASIC CYCLE TIMER
Figure 04.01



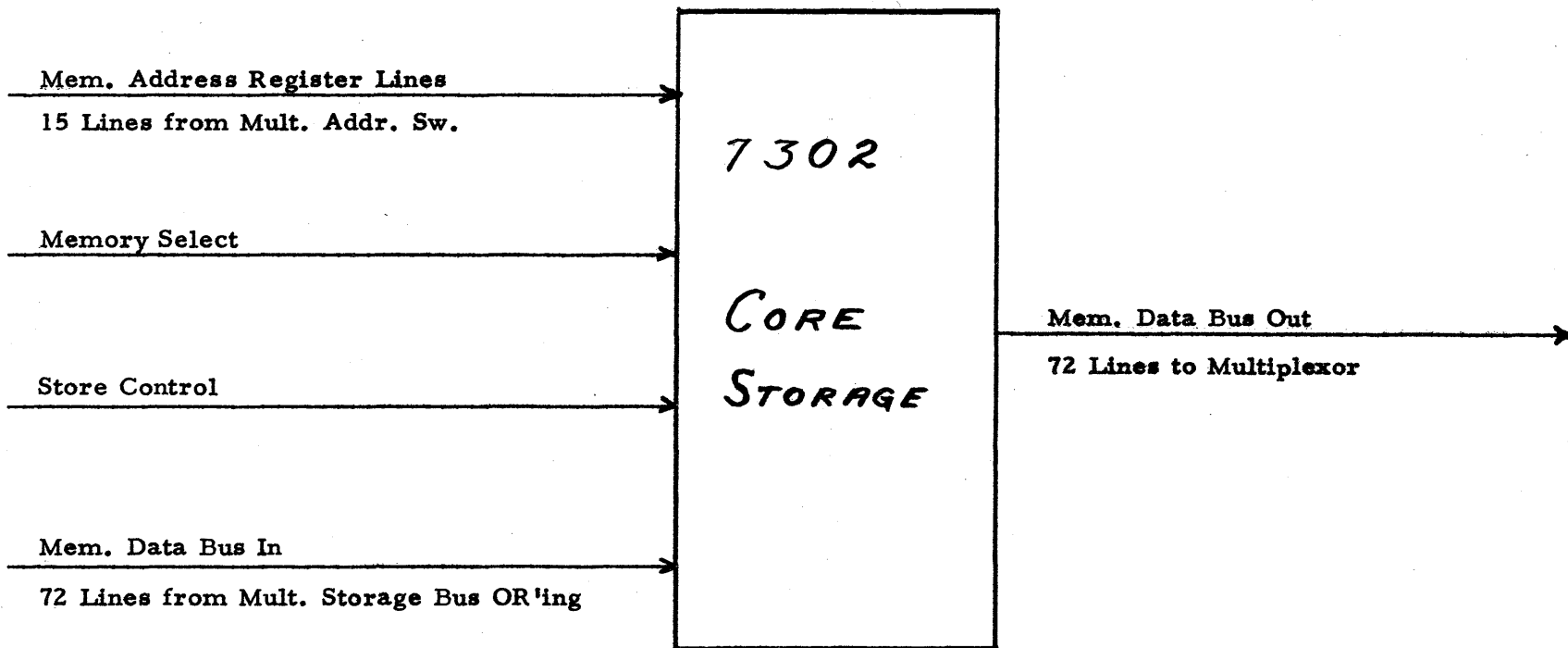
INDEX CONTROLS

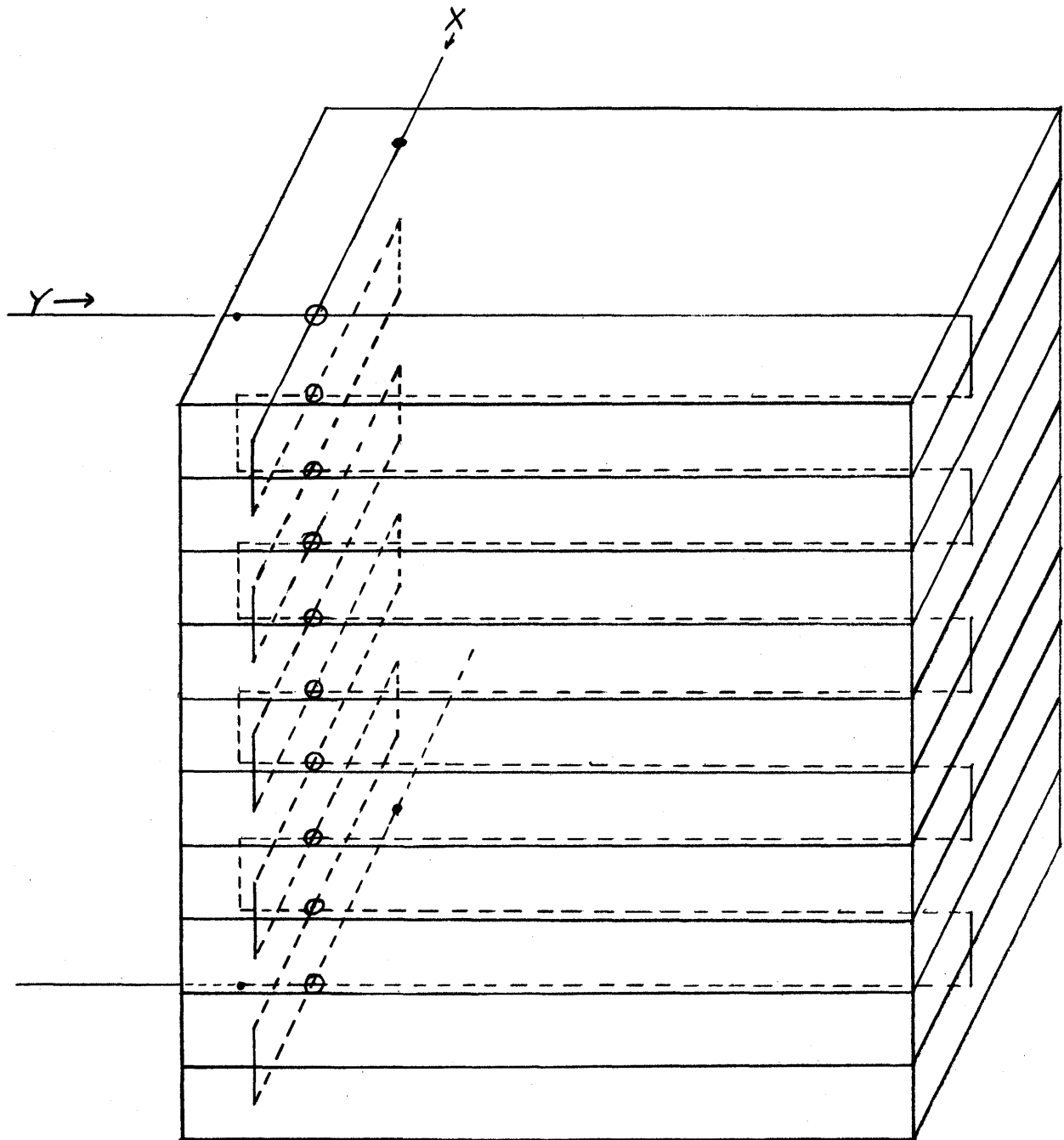
7090 ADDRESS MODIFICATION



253 Figure 04.03

HANDS
5-11-61

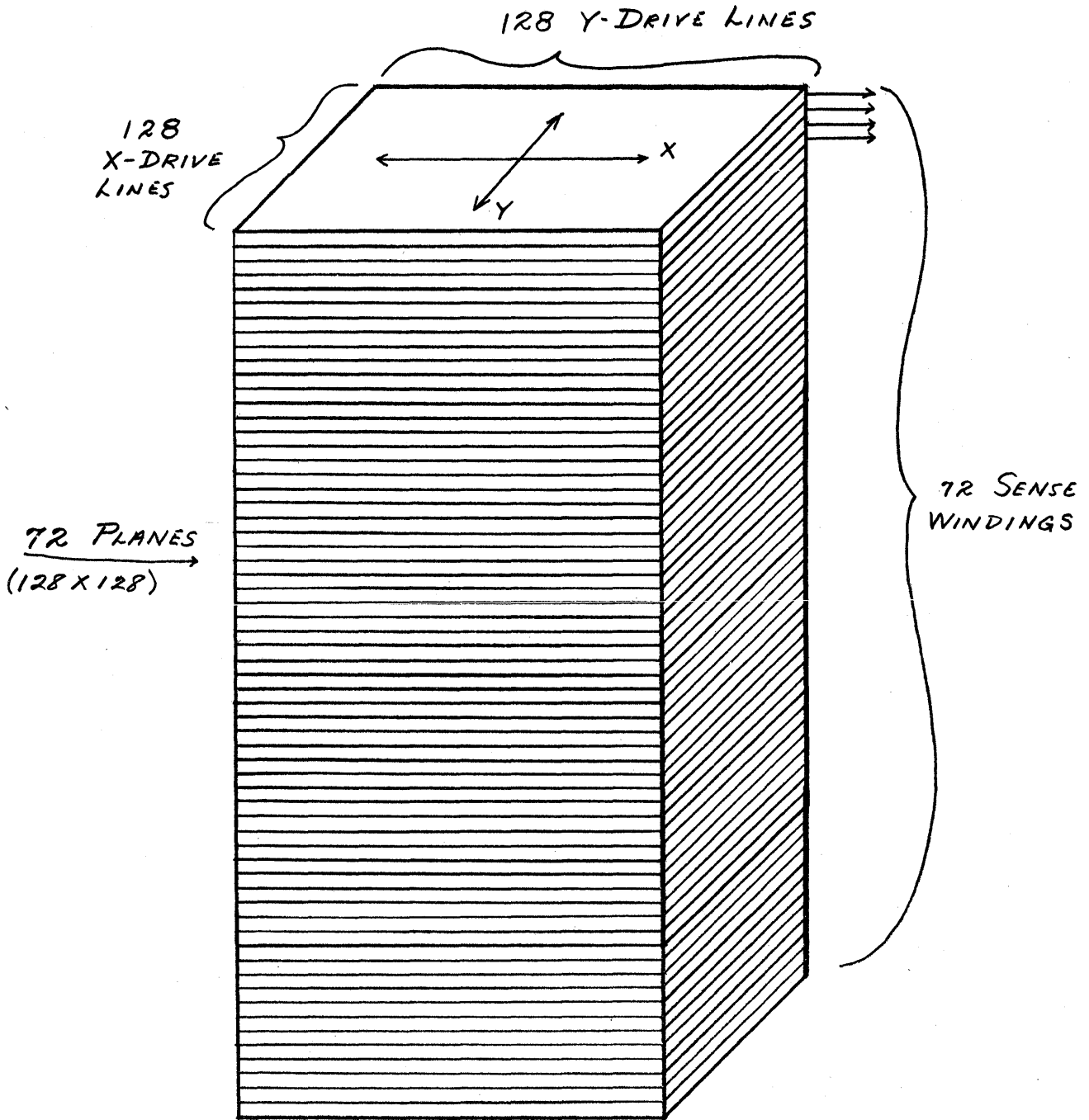


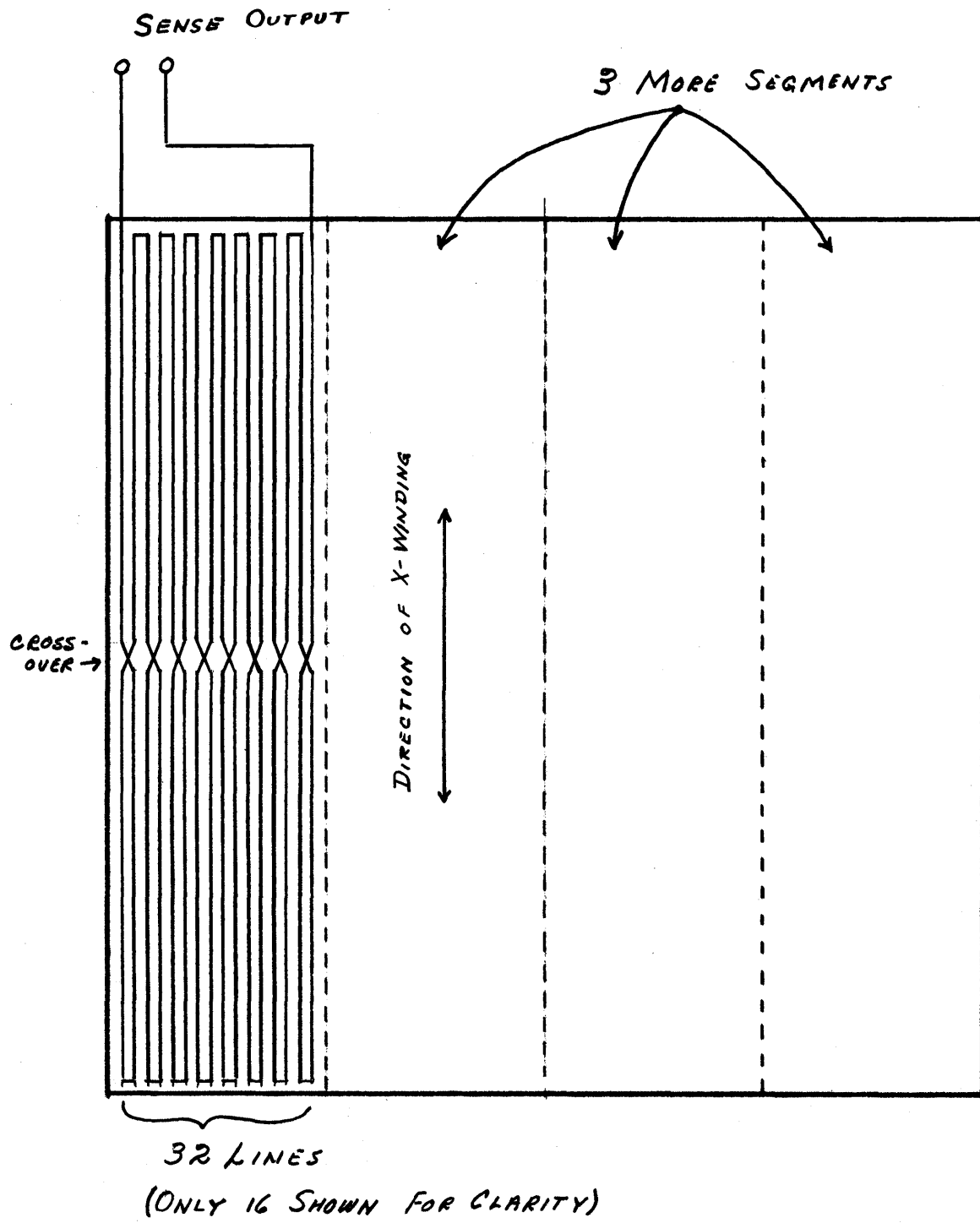


ONLY 1-X & 1-Y DRIVE LINE SHOWN FOR
GREATER CLARITY

COINCIDENT CURRENT ADDRESSING

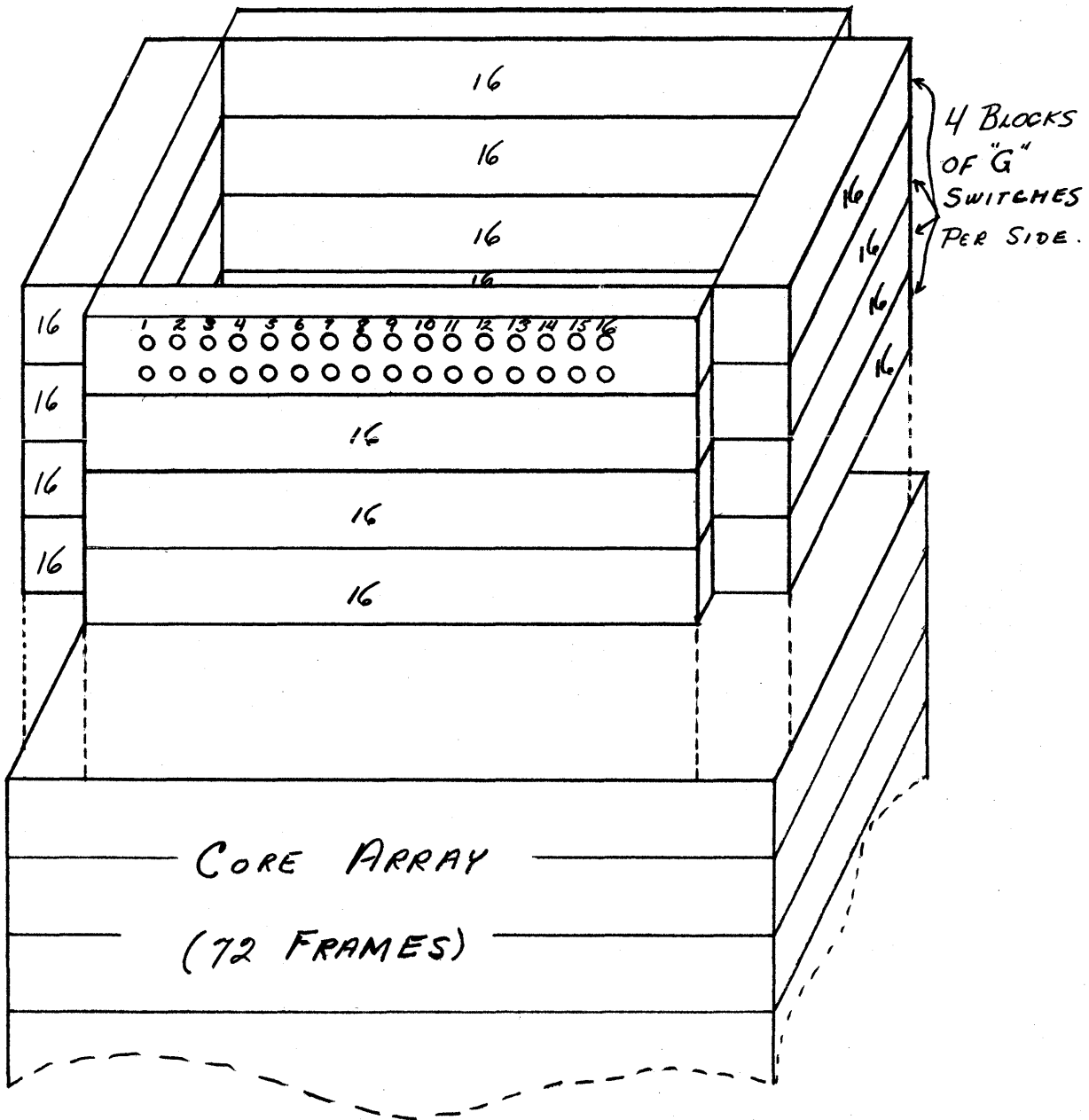
Figure





SENSE SEGMENTATION

LOCATION OF G SWITCHES
IN RELATION TO CORE ARRAY



Figure

06.05

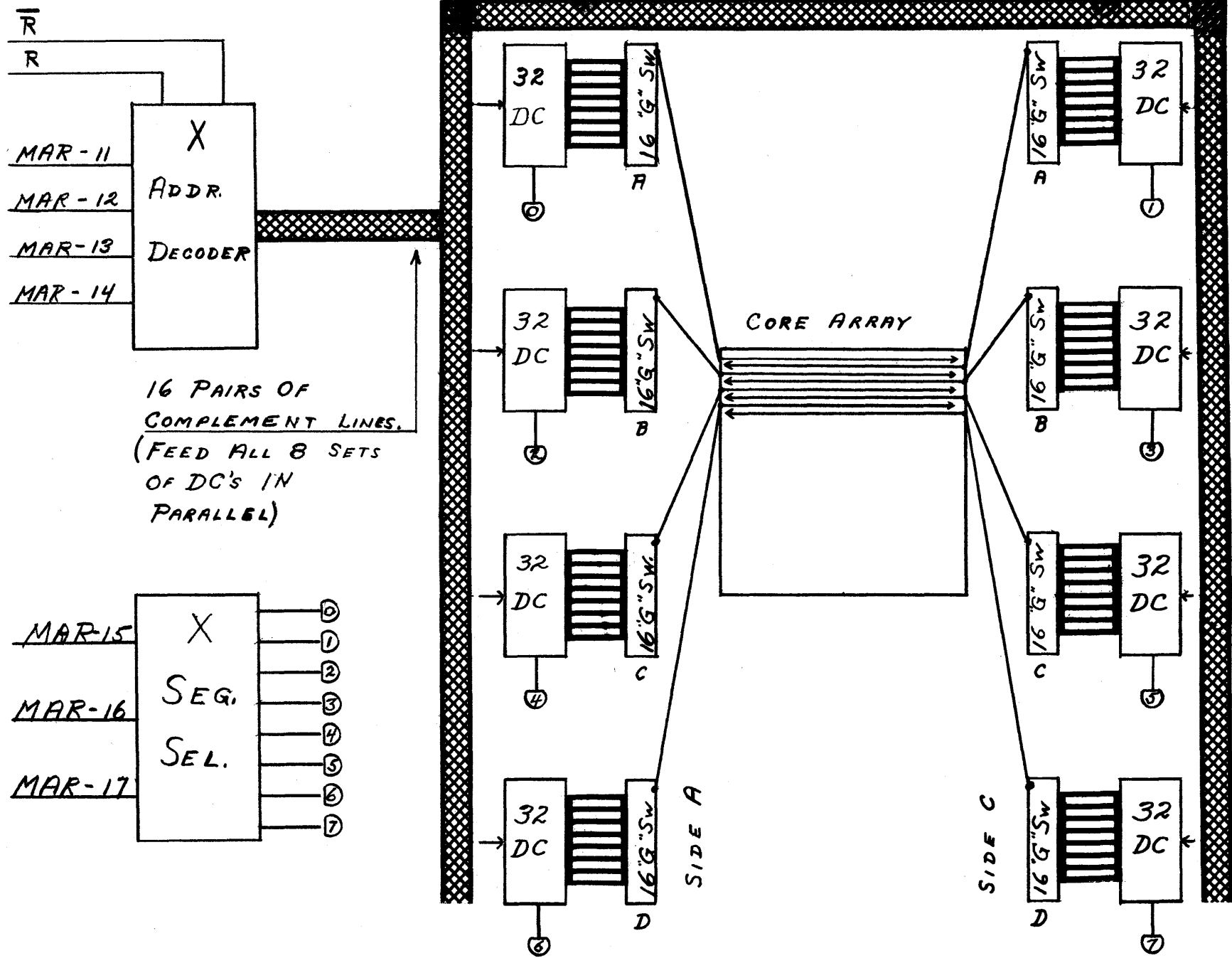


Figure 06.06
259

STORE CHANNEL - SCH
± 064X IE

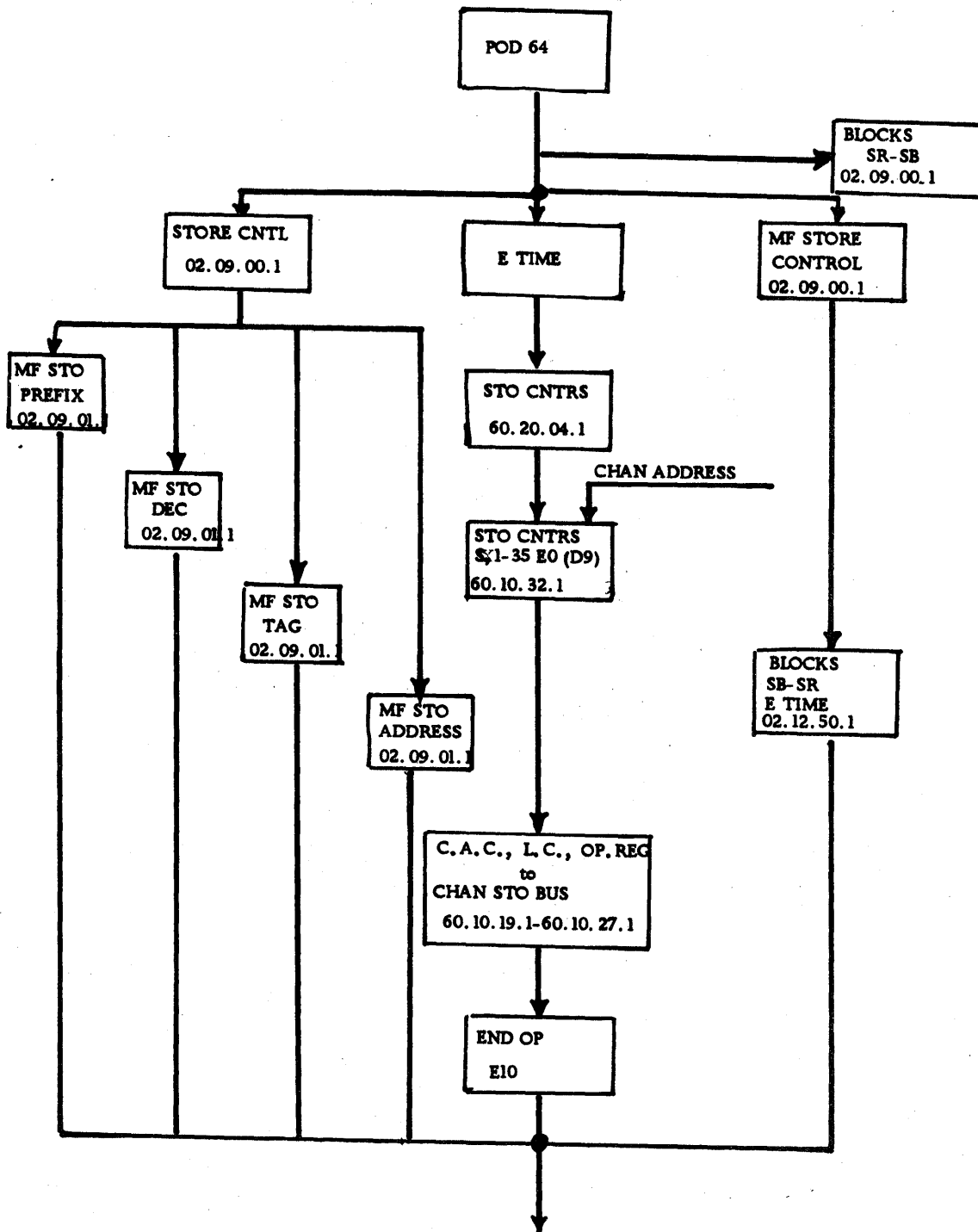


Figure 08.01
260

01.01 ANSWERS

COMPUTER CHARACTERISTICS

1. See Fig. 01.01
2. Five
3. III and IV
4. I and III
5. V, ten
6. a. 150
b. 100
c. 250
7. Control, arithmetic-logic
8. True
9. 32, 768 words, 36 bits
10. True
11. Four, I, L, E, B
12. I
13. B
14. E, E, L
15. S, 1-11
16. 21 - 35
17. Fig. 11, Page 11 of PIM

01.02 DATA REPRESENTATIONS

18. 9 row left
9 row right
24 words
19. 2, 72
20. Image, program
21. 9 row

22. Word count, 4-18, 9, 22
23. 22-36, 8 row left, 8 row right
24. Checksum, checksum, loader
25. 6, 6, 6

01.03 NUMBER CONVERSIONS

1. 2672_8 , 010110111010
2. 100000_8
3. 64
4. .2
5. .000376
6. .001953
7. .453125
8. 16384
9. a. 675
b. 305
c. 10
d. 333
e. 200
10. See pg. 131 of 7090 Ref. Man.

01.04 ANSWERS

INSTRUCTION SPECS.

1. One card
2. True
3. Location, 6
4. 7, blank, variable
5. operation, 3, 7
6. Variable, instruction
7. After col. 17.

- | | | | |
|---|-----------|-----|-----|
| 8. True | 21. START | LDQ | X |
| 9. Address | | MPY | C |
| 10. Separate "address, tag, decrement" | | STQ | CX |
| 11. Indirect addressing | | LDQ | X |
| | | MPY | X |
| | | MPY | B |
| | | STQ | BX2 |
| 12. a. contents of storage location specified | | LDQ | X |
| b. contents of entire MQ register | | MPY | X |
| c. contents of 3 through 17 of acc. | | MPY | X |
| d. contents of entire accumulator | | MPY | A |
| 13. index register | | STQ | AX3 |
| | | CLA | AX3 |
| | | ADD | BX2 |
| | | ADD | CX |
| | | ADD | D |
| | | STO | X |

01.05 ARITHMETIC INSTS. & LOGICAL OPS.

FIXED POINT ARITHMETIC

- | | | | |
|-----------------------------|-----|-----|---|
| 14. CLA
C(AC)
s, 1-35 | 22. | CLA | Q |
| 15. P, lost | | LDQ | A |
| 16. MQ | | DVP | B |
| 17. MQ, AC | | STQ | Q |

F. P. AND SHIFT

- | | |
|---------------------------|-------------------------------------|
| 18. greater, divide check | 23. S, 1 |
| 19. 201 CLA 4001 | 24. 9-35, 9 |
| 202 SUB 4002 | 25. 200 |
| 203 ADD 4003 | 26. zeros, zeros |
| 204 STO 4006 | 27. True |
| 205 ADD 4004 | 28. least, 33, less, AC |
| 206 SUB 4005 | 29. True (if you went to school) |
| 207 STO 4007 | 30. AC |
| 20. 100 LDQ 400 | 31. False |
| 101 MPY 400 | 32. multiplying by 2, dividing by 2 |
| 102 MPY 400 | 33. double, disastrous!! |
| 103 STQ 410 | |
| 104 MPY 400 | |
| 105 STQ 411 | |
| 106 MPY 400 | |
| 107 MPY 400 | |
| 110 STQ 412 | |

- 34. a. LGL
- b. ARS
- c. RQL
- d. LGR

- 35. LDQ 403
- FMP 404
- STO 410
- LDQ 400
- FMP 401
- LRS 043
- FMP 402
- FDH 410
- STQ 410
- (all address in octal)

- 36. both, one
- 37. one or both
- 38. must not, one, zero
- 39. masking, packing

40. CLA puts a mask of all bits in 17-35 of AC. This is AND'ed to location DATA to mask out the op "TRA" leaving only address of 1000. PACK which contains oct 0500 (CLA) is shifted in AC to P_s, 1-35. The ORS then "packs" it into Data which now contains "CLA 1000".

- 3. a. C(Y) S-35 replaces C(SI) 0-35
- b. C(SI) 0-35 replaces C(Y) S-35
- c. C(SI) 0-35 replaces C(AC) P, 1-35
- d. Each bit of C(AC) P, 1-35 resets the corresponding bit of C(SI) 0-35
- e. If each bit in C(AC) P, 1-35 matches the corresponding bits in C(SI) 0-35 transfer to Y

Control Operation

- 4. unconditional (TRA, etc.)
- 5. conditional, test or skip
- 6. a. C(AC) QP, 1-35 are zero
- b. AC sign is minus
- c. AC overflow indicator on
- d. AC overflow indicator off
- e. C(AC) Q, P, 1-35 not zero
- f. AC sign plus
- g. MQ sign plus
- h. C(MQ) less than C(AC)

01.06 Control, Indexing and Trap

Sense Indicators

- 1. 36
- 2. four; LDI, PIA, STI, PAI;
 - a. Set or logical OR
 - b. Reset
 - c. Invert
 - d. ON test
 - e. OFF test

- 7. a. Halts at location STOP C(AC)=3
- b. Halts at location TWO C(AC)=7
- c. Halts at location ONE C(AC)=3

8.	START	CLA	ONE
		LDQ	TWO
		TLQ	A
		LDQ	THREE
		TLQ	B
	D	STO	LOW
		HTR	START
	A	CLA	THREE
		TLQ	B
		TRA	D
	B	STQ	LOW
		TRA	D+1

9. A NOP can be entered (from console also) to remove an undesirable step in a prog. without disturbing the sequencing and addressing of the program.

Indexing Concepts

- 10. index reg., address
- 11. address (location)
- 12. 3, true, complement, decreased or increased
- 13. A, B, C; TAG; 18, 19, 20
- 14. A, B and C, multiple
- 15. 15 bit , true
- 16. Complement, subtracted, 4, complement, adding
- 17. OR'ed, 773₈, effective, 6001₈
- 18. a. START LXA INDEX, 1
 LDQ X
 GO MPY X
 LRS 035
 TIX GO, 1, 1
 STQ XNINE
 INDEX HTR 9
 b. START AXT 24, 4
 STZ SUM
 GO CLA NUM+24, 4
 TMI TIX
 ADD SUM
 STO SUM
 TIX TIX GO, 4, 1
 c. BEGIN AXT 31, 2
 STZ PSUM
 STZ MSUM
 GO CLA NUM+31, 2
 TMI NEG
 ADD PSUM
 STO PSUM
 TIX TIX GO, 2, 1

- NEG ADD MSUM
 STO MSUM
 TRA TIX
 CON HTR 30, 0, 0
- 19. START LXA CONS, 1
 GO CLA X + 50, 1
 SUB Y + 50, 1
 STO XY + 50, 1
 TIX GO, 1, 1
 CONS HTR 50
- 20. START AXT 100, 1
 SORT CLA X, 1
 CAS X + 1, 1
 TRA SWAP
 ERROR HTR START
 GO TIX SORT, 1, 1
 SLT 1
 DONE HTR START
 TRA START
 SWAP LDQ X + 1, 1
 STO X + 1, 1
 STQ X, 1
 SLN 1
 TRA GO

Indirect Addressing

- 21. 12, 13
- 22. asterisk, letter, instruction
- 23. a. 20
 b. GO
- 24. a. GO
 b. DONE
- 25. STOP

Trap

26. True
 27. a. TRAP, WOW
b. DONE, TRAP
c. NO
 28. 177, -200, 377, 000
 29. overflow, underflow
 30. location +1, instruction, address portion, 0000
 31. spill, decrement, 10
 32. see page 79 of PIM
- 01.07 Input-Output Operations
1. I/O Select, RCH, command, asynchronous
 2. destination, amount, core, I/O device
 3. see Fig. 50 in PIM
 4. OP Reg. in S, 1, 2, 19
Add Reg in 3-17
Loc. Counter in 21-35
 5. True
 6. address field
 7. see PIM pages 70 to 73
 8. no transmission, ignored
 9. a. unconditional
b. chan in operation
c. chan not in operation
d. tape check
e. end of file
 10. indirect addressing
 11. location 500
 12. a 50 word record followed by a 70 word record
 13. a. 1st record = 180 words
2nd record = 25 words
3rd record = 4 words
Total = 209 words
b. 50 words from Rec #1
4 words from Rec #3
54 words total read
 14. 15 words from Rec #1
15 words from Rec #2
10 words from Rec #3
Disconnect
 15. a. EOF
b. tape check
c. a "T" type command that disconnects DC
 16. instruction counter, 16
 17. 17, back to point in program where trap occurred and continue on
 18. Restore (RCT)

REVIEW QUESTION ANSWERS

02.00 PACKAGING & COMPONENT CIRCUITS

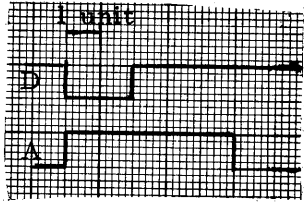
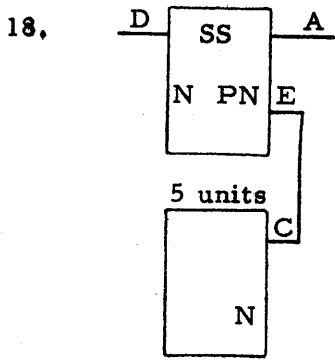
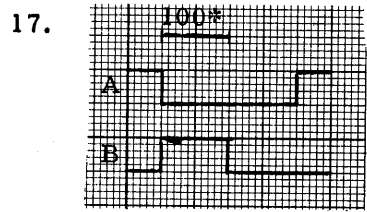
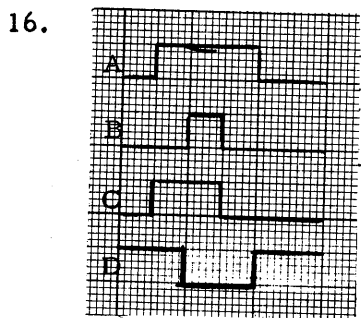
02.01 PACKAGING

1. Six, four; A, B, C, and D; E, and F
2. Provide a means of connecting external cables to the internal circuitry.
Also allows connections between tower circuits within the frame.
3. A to K, 1 to 28.
4. Circuit wiring must leave a tower.
5. A to D, circuit wiring must leave either panel 1 to 3 or 2 to 4 and vice versa.
6. A to R
7. BCD, FGH, KLM, and PQR; A, E, J, N
8. F; A and B; 9 to 16
9. The ribbon cables must be placed in the trough so that the panel column numbering at the tailgate slide connectors remains in a top to bottom sequence for all gates.
10. 20 and 40
11. A to R and S to Z 1 to 8
12. J to R

02.02 COMPONENT CIRCUITS

1. a. + .78
b. -6.48
c. -6.78
2. +.8V, ground or 0, PNP
3. +.8V, minus six, NPN
4. ground or 0, minus six, in phase
5. The forward bias across the input transistor base to emitter junction is higher.
6. 60*
7. Cap cut

- 8. Type A used 909 ohm resistor to +6V and type B uses 4.53K resistor to 30V. The collector load resistors have different values.
- 9. Type B provides a better input current source so that transistor parameters are less critical.
- 10. 20*
- 11. The 82 ohm resistor suppresses oscillations caused by the inductive coupling networks. The coils compensate for the output capacitance to produce optimum square-wave response.
- 12. Increase number of input legs
- 13. +N
- 14. Down
- 15. Negative-Off



REVIEW QUESTION ANSWERS

03.00 POWER SUPPLY

03.01 LOCATIONS

1. On PCU panel (additional blower CB's are located on each frame for each gate)
2. Marginal check relays
3. Two, located in PCU
4. In the data channel with which the particular tape drive is associated
5. No, they are located in the PCU
6. The 208 volt, 400 cycle motor generator output is regulated by controlling the generator field current
7. One on console and one on PCU
8. 208 V 3Ø 60 cycle from wall via data channel
9. By air flow detect cards located in each gate
10. 2, one on console and one on PCU
11. Power - indicates core power (variac cycle 2) on. Console Power On - indicates modular supplies, (variac cycle 1) on.
12. a. Input power to that module is removed
b. Power sequencing for that module occurs

03.02 STANDARD MODULAR SUPPLY

1. Phase 1 Wire N TB4-7
Phase 1 Wire P TB3-6
Phase 2 Wire Q TB3-7
Phase 2 Wire R TB2-6
Phase 3 Wire S TB2-7
Phase 3 Wire T TB4-6
2. Protection is lost; blown fuse, overtemp indication and air flow detect indication but power is not dropped with mag amp card removed.
3. Loss of 1/2 of one phase of rectified voltage; therefore low voltage on +30V and -36V supplies
4. a. Power drops on multiplexor frame - rest of system stays on.
b. Gates A, B, C, D of CPU 1 - (Through opening of interlock circuits by use of CB's side switches)

5. Voltage shown on the 6v, 12v meter is taken from the output of the M/C variac which is fed by the 208v, 400v, line. Actual DC is not measured by the meter.
6. T5 is bucking transformer for the -6v supply. Without T5, the output would be approximately 9 volts.
7.
 - a. No voltage would be developed across the CB trip coil but operation of the machine would not be affected otherwise.
 - b. No voltage developed across the CB trip coil and a blown fuse would not be detected.
8. Selection of gate and module for +6 or -12 biasing - 9.03.10.1 to 9.03.30.1

03.03 CORE STORAGE POWER SUPPLIES

1. The 30V-60V meter is reading actual DC voltages in the core storage supply.
2. Output voltage would increase
3. No. This is only an indication and has no control over power. The CE should check these lights occasionally.
4.
 - a. X and X matrix switch bias
 - b. On the tank top, syst 00.02.02.0 test receptacle B-24
5. Nothing under normal conditions. In the case of several neons ionized, excessive current could flow through the control winding of the amplifier and burn it out. The diode carries the excess current, within limits.
6. Current would not flow in the transistor or the SR control winding. There would be no control saturation and output voltage would be increased.
7. Forward bias on the transistor is increased. This causes more current to flow in the transistor and in the control winding of the saturable reactor. Since the flux set up by the control winding is opposed to the supply winding flux, the effective inductance is increased and the output voltage is reduced.
8. Mag. flux

03.04 POWER SEQUENCING

1. Only after Emergency Off switch has been restored, power on reset depressed, Generator CB1 restored and then PWR ON depressed.
2. These are side switches ON HR30. C. E. Reference Manual Figure 3.1-19, where they are called overload relays.
3. On Relay, Fuse and Diode Panel, Ref. Manual Fig. 4.1-5. This panel is in PCU near marginal check variacs.

4. Two cycles. That module will get its power on along with the core special supply.
5. MG blower
6. a. The MG thermal will operate due to heat buildup in the MG
b. -----
c. Because HR30 drops
7. So that the full system can be brought up simultaneously, or an individual unit. First cycle must complete before second starts.
8. 2
9. To prevent bringing power up on any unit that has a CB-1 down, and to drop power when on the whole frame when a CB drops due to a fault in one of the two TOWER supplies.
10. The special voltages, +60, +30, -6 volts
11. TD-1. This is a dash-pot controlled delay to allow DR12 and HR26 to hold through TD 1-2 points 5 seconds after power off.
12. 400-cycle interlocks to ensure all three phases are up.

03.05 MARGINAL CHECKING

1. To prevent selecting any gate for biasing while voltage is not at normal.
2. They will still bias but will not have the proper interlock protection described in answer 1.
3. Just the skill and knowledge of the customer engineer. If driven high or held up for an extended period driver trouble will result.
4. C gate - See systems 9.05.11.1
5. True. The selection is not affected, but the protection against a second selection while at bias would be lost.
6. It is selected as channel 4 for biasing. (Still operates as channel 2 for Select Inst.)

03.06 POWER CONVERTER

1. a. P1 adjusted for correct output voltage
b. P2 adjust the correct operating point of the control circuit and is not to be varied.
c. P3 adjusts the stability and is set to prevent "hunting"
(See Ref. Man. Page 3-2)
2. Control of field current in generator

3. The flashing circuit is cut into operation whenever the output of the generator falls below 200v approximately. The generator field is flashed whenever power is brought up for the first time, after a Normal Off and after an Emergency Off, Power On after DC Off does not flash the field.
4. 400 cycle surge current (short) overload 400 cycle prolonged mild overload. Loss of input 60 cycle AC or less than 60 % of normal voltage.
5. No effect. Power to Generator flashing circuit is interrupted but it is not in use when power is already up.
6. With T3 having an open winding there will be no source of field flashing. If the Mg set is inactive for a long (??) period of time, there is a possibility of being unable to get any output from generator due to loss of residual magnetism.

03.07 MISCELLANEOUS CIRCUITS

1. The 400 cycle section is not functioning.
2. The light on 9.02.04.1 shows the first power-up cycle is completed, the light on 9.06.21.1 shows the second (core storage) power-on cycle is completed.
3. A side lights Data Channels "Power ON" lamp
B side generates Power On Reset to Channel
4.
 - a. False
 - b. False
 - c. False
5.
 - a. Pwr Ck lite on Channel & I/O Pwr Ck lite on Console
 - b. Fuse light on tape drive
 - c. Pwr Ck lite on Multiplexor and Cent. Comp. Pwr Ck lite on Console
 - d. None
6. Loss of 48V will cause the "Open fuse detection" Mag. Amps. to drop CB-1 on all SMS power supplies. No lights will be lit on any frames.

03.08 TROUBLE-SHOOTING

1.
 - c. Power will come up part way but then will go back down
 - d. CB1 will trip for Power Supply A
2.
 - a. CB34 has tripped
 - b. CB35 has tripped
 - c. CB29 has tripped
 - d. CB28 has tripped
3.
 1. c.
 2. a.
 3. d.
 4. e.
 5. b.
 6. f.

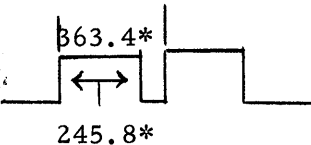
4. MG Blower will not run. Power would come up normally but a thermal condition should drop it later.
5. e.
6. a, e.

CYCLE & INSTRUCTION TIMING

ANSWERS --- REVIEW QUESTIONS

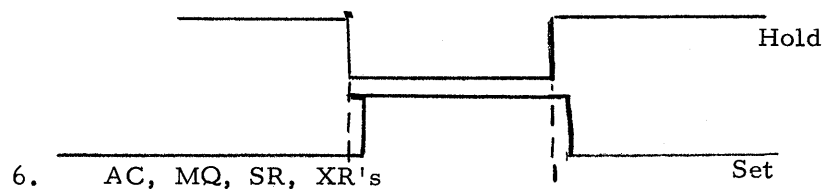
04.01 Timing Circuits

1. crystal, 2.75, multiplexor
2. 2.1818
3. clock, CP Set pulse
4. odd clock drive, even clock drive
5. ring, twelve
6. clear (PWR ON)
7. 364*
8. last half, even clock drive pulses.
9. 182*
10. 12
11. gated output of preceeding trigger.
12. output of two stages later.
13. 08.00.44
14.
 - a. PWR on Reset
 - b. Clear
 - c. A11 (D1) pulses
15. Hold all other triggers off and turn on "O" trigger.
16. 2
17. Clock triggers 7 and 8 would be on. All others OFF. Number 9 could not be turned on so #10 would not come on. These two not being turned on would prevent #7 and 8 from being turned off.
18. A7(D3) because a delay of one timing pulse is assumed when a pulse travels from Multiplexor to CPU
19. do not compensate
20. A1D1, A0D1, A1
21. 1.4*

22. 5.5 MC $70 \pm 10^*$
Allow shift cells to function, step shift counter
23. True
24. CP timing (occurrence not duration) This is to compensate for block delays so shift cells in machine can operate at optimum.
25.  Output would be a combination of full osc. pulse and delayed 64* pulse.
- FREQ 2.75 MC
Pulse Width 245.8*

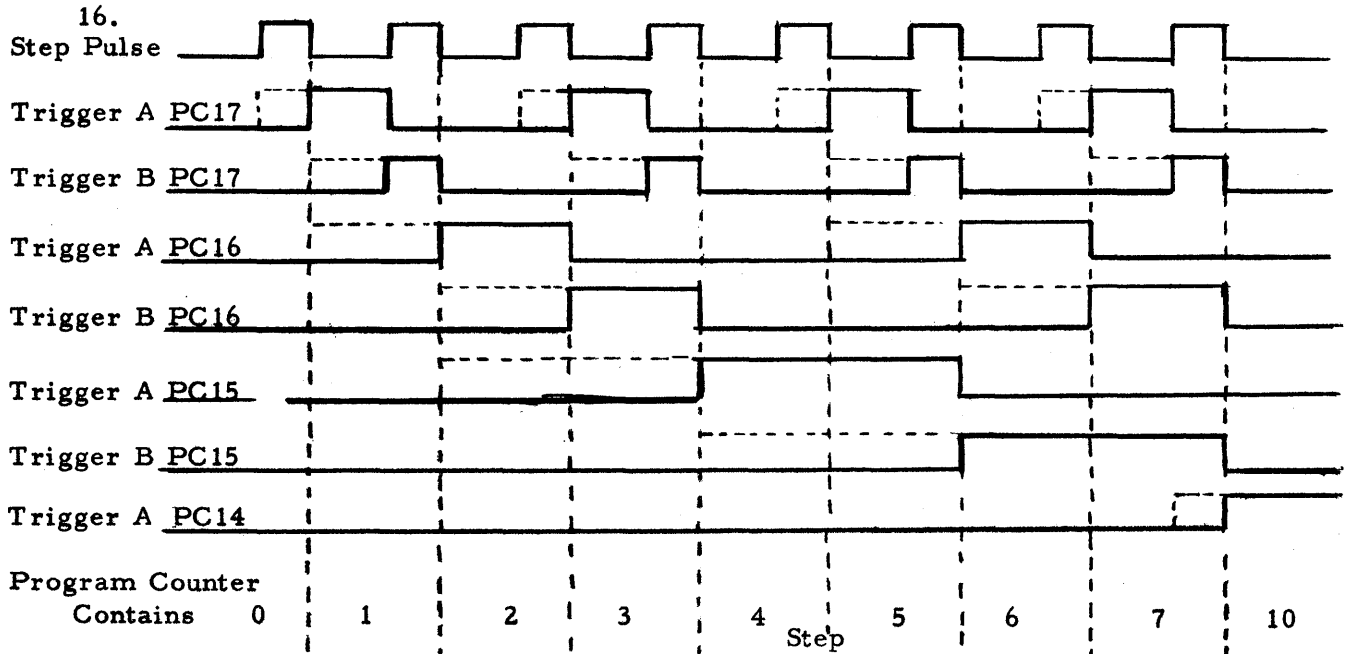
04.02 Basic CPU Circuitry Operation

1. See Fig. 3.1-2A in CE Inst. Ref. Manual
2. See Fig. 3.1-2B in CE Inst. Ref. Manual
3. Bi-stable storage, and simultaneous read-in and read-out.
4. TO/TA, in, upper TA, lower TO.
5. Hold must drop before set comes up and then the hold must come back up before the set drops.



7. Sum and Carry (Also And Look-Ahead and Or Look Ahead)
8. Seven, six
9. True
10. Two
11. And look ahead, OR look ahead, adder.
12. 1st stage LAC, CO, LAC OR, carry to adder 35.
13. Adder carry inputs, group.
14. Floating point

15. Five, three, speed-up or look-ahead.



17. See figure in CE Inst. Ref. Manual

18. a. +N
 b. 57₈
 c. 60₈

19. Nothing happens with the rise of the step pulse but trigger B (lower) comes on with the fall of the pulse.

20. 17B (lower) goes off and 17A (upper) comes on with the rise of the pulse, 17A (upper) goes off with the fall of the pulse.

21. 15 A (upper) and 16 B (lower)
22. 14 A (upper) and 17 A (upper) are on, SC = 11
23. 16 A (upper), 17 B (lower), 15 A would be present

04.03 Cycle Generation

1. I, E, L, B
2. 1 (I)
3. A11
4. I6
5. Program Register
6. IE, IE, IL, IL, IE
7. L Time Call - 000-177 and 700 up
E Time Call - 200-677
8. Block the output of the I, E, L time triggers when the Master Stop Trigger is on or when an End Op. or Go to E condition arises when in B time.
9. No, next cycle may be a B cycle
10. A11
11. AO
12. Stop, interrupt
13. Go to E, End Op on.
14. End op trig on Sets go to I, I9D1
15. E7D1
16.
 - a. End op on
 - b. Share - with L cycles of CPU instructions except 1st L cycle.
 - c. Interrupt - between E cycles of a convert inst. except 1st E cycle.

04.04 I Cycle

1. Get the instruction from core, decode it, route the address.
2. A9 (D3)
3. I7 (D1)
4. MX Storage Buses
5. SXA (I, E)
6. To allow a 40* positive gate to Reset the AR before reading in
7. Whatever location address that was sitting in AR at that time.
(No reset to AR)
8. 1 and 2 will be routed to Program Register 8 and 9. TXI will be sitting is PR
9. 7371 octal
10.
 - a. Bug is "E" output
 - b. I trg on via End Op (Normal)
 - c. Multiple time error on at I4
 - d. MST on I4
 - e. Machine stop
 - f. No "I" output after 4 time
11.
 - a. Should go I E L L, etc., but goes I, E.....
 - b. I, E - Ok
 - c. No L time call +A (4G) - 12.1
 - d. No EOP on E
 - e. Machine being up with no cycle output in automatic
12.
 - a. Cyclic makeup should be I L L L-End Op.
 - b. I - Ok
 - c. "E" forced through +A (5A) - 8.00.12.1
 - d. "L" next + A at 4G - 8.00.12.1
 - e. SC = 0 End Op.
 - f. Shift works properly

13. PC Time

0	I8	ADR Reg.	00000
		Prog. Reg.	CLA or 500
		Tag Reg.	0
1	I10	SR.	+0500 000 00100
		Prog. Reg.	CLA or 500
1	E4	SR	+0500 00 0 00100
		ADR Sws.	00000
		ADR Reg.	00100
1	E11	SR	+000 000 000003
		ADR	00001
2	IAE11	SR	+000000000003
		ADR Reg.	00003
3	I10	SR	+0774 00 2 00005
		ADR Sws.	00003

14. Page 08.00.09.1. A0 at 4C.

15. Yes. The first E cycle could come from the block at 5A or the E Time Call. Since this is a one cycle instruction, there is no need for an E cycle following the 1A/E and the End Op trigger would have been turned on giving Go to I and blocking the setting of the Go to E trigger.

16. a. No. At no time is an attempt to set PC 8 made.

b. No. At I9D3 nothing would be taken to the address switches thus during the IAE cycle the C (00000) would be brought out and the address 300 would be placed in the address register at E 11 of the IAE cycle. Thus the next instruction would come from 300 and the machine would halt with 301 in the PC.

c. No. The block at 2A would still function normally.

d. Yes. This would cause all zeroes to be placed in the Adr. Reg. at IAE 11 and the next instruction would come from location 000 resulting in the program never halting.

e. Yes. For the same reason as (d) above. This line is the output of an A0 circuit and a short on it will hold down the output of all the Or'ed circuits.

17. Yes. The failing output is the MF SB OUTPUT line which feeds from CPU to MPLXR. This would not affect the performance of a type POD 76 instruction.

18. Yes. The failure would affect SR pos. 7 only and this would not be used for any decoding or control on a POD 76 instruction. The Prog. Reg. is fed directly from the SB which has not been affected.

19. C(acc) is 2, C(PC) is 2. The failure would prevent setting a one into SR Sign which does not occur in this program.
20. C (acc) is 000..000, C (PC) is 2. Nothing could be placed into the SR during I or E time. This would not affect the cycling of the program or its ability to halt correctly. All references to core for data would be to location 000 and no data could be set into the storage reg. resulting in an empty accumulator.
21. Same answer as 7 above except that the op codes would have appeared in the SR at I7. C (acc) = zero because the op code for HTR = +0000.
22. 08.00.12.1. The block at 5B.

04.05 Indexing

1. effective address.
2. Indexable - modify address
Non - indexable - usually modifies Index Register content, cannot modify address
3. two's
4. True
5. System tracing
6. AC = 13
TIX decoded as HTR - Stop at 3.
7. No - Contents of location 437

04.06 Indirect Addressing

1. a. TRA 0000
b. (1) 0005
(2) 0002
(3) 0003
(4) 0003
2. a. Normal I, (E), End Op.
b. TRA would EOP thru "-0" (2G) - 8.00.00.1
3. I9, E11

4. SR bits 12, 13
5. I9, E2
6. "+N to E Time controls" from 3D pin B.

04.07 Manual Control Operations

1. 200*, Man. Ctrl, Trig.
2. Any of the individual operation Triggers coming on.
3. B cycle interrupt, True
4. " Not B Cycle interrupt gates 1, E or L cycles - So - We have to depend on the "Go to I" trig being on. It doesn't get Reset by Clear or Reset.
5. True
6. PC 2 trig (3.05.00)
7. A4 (after op panel trig goes off), A1.
8. 24 ms, 104 ms.
9. I, I7, A11
10. Normal machine speed, depression of the single cycle or mult step keys.

04.08 Diagnostics

1. A1
2. Sign key down
3. See 8.3 of CE handbook
4. 9M51
5. 9T51
6. 9T55; Keys 19, 20, 23, 24 Down' 35 down for graph.
7. Yes - 9S51L is loaded into upper memory itself

8. Checksum error stop for tape diagnostics - it didn't load - pick up or drop off bits.
- 9 same as question 8.
10. loc. 00000

PRELIMINARY INSTRUCTIONS

REVIEW QUESTION ANSWERS

05.01 - Data Flow - Answers

1.
 - a. Accumulator
 - b. MQ
 - c. Address Switches
 - d. Adders
 - e. Sense Indicators
 - f. Storage Register
 - g. Operators Entry Keys

2. Accumulator, Storage Register, MQ, Index Registers.

3. Accumulator position 1 to adder or complement of accumulator 1 to adder.

4. Storage Register, Accumulator, Complement Accumulator.

5. SR17 and SR35 on one input, complement AC17, AC17, AC35, XR17, and MQ5 on the other input.

6. SR3, Forced one to AD3-6-8, SR21, and Shift Counter 12 on one input, AC3, Complement AC3, and SR3 on the other input.

7. Accumulator Q

05.02 Fixed Point Arith, and Work Xmissions - Answers

1. +400000002124₈

2. $+104_8$
3. $+176_8$
4. This will cause the SR Sign to be set plus at E9.
5. A3 if a Q Carry occurs regardless of the signs. Page 02.10.36.1
6. $+, Q=1, P-1, 1-35=000000000000$
7. To save the contents of AC Q during and ACL Instruction and leave it unchanged.
8. A. 7211_8
B. 10257_8
9. It should Halt at 100 on a HTR 40 instruction. With the Bug on the machine it would Halt at Zero on a HTR 000. The STO would set into the Program Register as an STZ.
10. A. $-000\ 000\ 000\ 000$
B. $+060\ 200\ 000\ 040$
C. With the trouble on the machine the accumulator would not be gated and set to the Storage Register at E1D1 leaving the instruction in the SR (Set in at I7D1). This would thus be gated to the storage Bus at E4D3 and stored.
11. $MQ=176_8$
 $c(200) = -0600\ 0000\ 0200$
12. MF Store Control, Store Control, Store Prefix, Store Decrement, Store Tag, Store Address.
13. POD60 all bring up the Store Control line which forces up all four store Partial Word lines. POD62 all store partial words and therefore do not use Store Control.
14. The Sense Indicators would be stored. The POD 60 would give Store Control to store the entire SR at E4D3. The Bit in PR6 is not examined on a POD60 Inst. The Bit in PR9 at 4E on page 02.09.00.1 will gate the AC to SR at E1D1 but at block 4C PR7 will cause the SI register to be gated to the SR and set in at E2D1. Storage would take place in Location '00 minus XRA.
15. This line drops the MF SB gate to prevent Data from core coming to CPU on store operations.

05.03 Shift Instructions - Answers

1. $+000000037774$

2. It is normally plus all the time. It goes minus when using the Machine Cycle Key except at A3, A7, or A11 to allow only 3 shifts per cycle.
3. -N 17D6 controls the start and -N SC ZERO controls the end of the Shift Gate.
4. Prevent L end op on shift counter equal to 7 or less when using Machine cycle key. Shift counter must go to zero under this condition.
5. 20000₈
6. 26₈ (Reset at 17)
7. One
8. Sixteen
9. In case a Shift of less than 11₁₀ places occurs the L11 will still be able to shift signs. Also, this allows a shift of zero places to transfer MQ and AC signs to make them agree.
10. A "-N A 10D1 Delayed" prevents sensing SC=7 or less during the second half of L10. This means that the SC must equal 7 or less at the beginning of L10. before the stepping pulse (A CP Set Pulse) can step the counter.

05.04 Transfer Instructions - Answers

1. This line is used to bring up "Xfer Conditional" except in the case of unsuccessful one cycle transfers, transfers in trap mode, and the STR instruction. See page 02.10.09.1.
2. It blocks the normal "I" cycle function of AD3-17 to AS3-17 at I9D3. This makes the AD to AS dependent upon the "Xfer Conditional Line."
3. True.
4.

Conditions Not Met	Conditions Met
SR-AD A9D3	SR-AD A9D3
AD-AS A9D3	AD-AS A9D3
AS-AR I11D1	AS-AR I11D1
L Cycle	L Cycle
PC-ASq A0D3 & End Op	AR-PC L9D1
AS-AR A11D1	PC-AS A9D3 & End Op
	AS-AR A11D1
5. Allows the End op tgr to be set at E10 of an Indirectly Addressed single cycle transfer instruction.

6. Blocks using SR21-35 of the Indirect Addressed Location when an indirectly addressed one cycle transfer is unsuccessful.
7. "Xfer Conditional" without "one cycle transfer condition not met."
8. PC to AS at A9D3 with the "End op" tgr on is performed as any other instruction with "End op" on (excluding "XEC" and "STR") through the AND circuit at 4E-4F page 03.05.09.1. This is blocked on a successful one cycle transfer by "minus on one cycle transfer not trap" which is brought up by "PC Transfer Control" (page 02.10.08.1).
9. +A at 3G - STR - TXL - TXI - TXH - TTR - TSX - TIX - TEF - TRC - TCN - TCO- TNX - TRA
+A at 3F - TNZ - TZE - TQO - TNO - TOV - TMI - TPL - TQP
+A at 3D - TRA - TLQ - TCO - TRC - TIF - TIO - TIX

05.05 Control Instructions - Answers

1. POD42, POD 66, POD 00 and Not PR 8 or 9.
2. Block at 4E for HPR, Block at 3B for HTR.
3. True.
4. a. Program Stop Lite will come on.
b. Yes. By depressing "start" to turn off the Program Stop" tgr. this releases the +A at 3C page 04.20.16.1 to allow "Single Step" to turn off the MST.
c. The machine will perform the "L" cycle of the HTR which stopped it. During this "L" cycle, normally the HTR forces "Condition Met" and sets the Adr. Reg. (containing 00001) into the Program Counter at L9. This is then routed to Adr. Sws and Adr. Reg then to MAR. when the "End op tgr" comes on causing the next instruction to come from Location 1. Since the machine is in manual, in this case, the block at 4F page 02.10.07.1 will prevent "Condition Met" and the program counter will remain the same at "End op" time. Since the PC is not stepped at I9 of an HTR it still equals 004 and will bring out and execute the HTR again.
5. A. PC = 4 at the Halt.
B. PC = 2 at the Halt.
6. a. A. PC = 4
B. PC = 3
b. A. PC = 2
B. PC = 2

- c. A. PC = 4
- B. PC = 3

7. Block 4F.

05.06 Skip and Sense Instruction - Answers

1. Location 7. Although the CAL will place a bit in Acc P, the PBT (-760....00001) will have the address modified in the address by XRA prior to being set into the Shift Counter where the class and unit Address is decoded. Therefore the Address Switches will appear as 70144. The Shift Counter will be set by 10-17 or 144 and decode a class 14, unit address 04 and the instruction is decoded as an MSE Sense Lite 4. This will fail to skip. The LBT will skip as the address 0001 would not have been changed during the "E" cycle of the CAL instruction thus placing a bit in Acc. 35.

2.	CLA	I9 (D1)	2D	02.11.51.1
	CAS	I9 (D1)	2D	02.11.51.1
		L4 (D1)	3I & 3G	02.11.50.1
		L9 (D1)	3I & 3G	02.11.50.1
	LAS	I9 (D1)	2D	02.11.51.1
		L9 (D1)	3I	02.11.50.1
	HTR	Blocked by pin F of 2E page 02.11.51.1 at I9 (D1).		

3. 08H09 D6 on page 00.20.91.0. 7100 Volume VII

4. Unit Address = 00, Shift Counter equal 36X, Not PR PU Sense, and SOD 00.

5. Shift Counter equals 36X.

6. UA02 is UA Gate 1 and UA Gate 2 on page 03.03.02.1.

UA Gate 1 is No bits in SC 14 and 15.

UA Gate 6 is one in SC 16, Zero in SC 17 on page 03.03.15.1

7. POD - 17 - Output taken from TO of PR (03.04.01.1) to the POD (03.01.00.1)

SOD - I8 - Output to SOD (03.07.00.1) taken from TA of PR (03.04.01.1) which cannot come on during Reset of PR, I7 (D1)

Channel Address - I7 - The time the SR is set and the POD 76 comes up. (06.00.06.1)

Class & Unit Address - I11 - The time when the Shift Counter is set. (02.11.78.1)

05.07 Indexing Instructions - Answers

1. PAX
2. True - Example, to use an Index Reg. to add 7 to an address

000	AXC	7, 1,
001	TRA	10, 1,

The transfer will be to location 17. The XR will be set to 77771 and when taken to the adders during the TRA it will appear as 00006 plus a 1 carry to adder I7.

3. 42000
4. On 02.12.92.1 a set to AC - Sign is developed at I6 (D1) of PXA or PXD instructions with no input gate to acc. sign.
5. The true number originally in the XR. The complement of XR is gated to the adders from E0 to E5. At E1 the 2's comp. of the XR is set back into the XR thus the output (always complemented to the adders) is the true original contents. This true number is set into the SR to be stored at E4 (D3) and is also set back into the XR at E5 to restore it to the original value.
6. False. An indexable instruction is one capable of having its address portion modified by an index register. Therefore Index transmission and Index Transfer instructions are not indexable even though they require the use of a Tagbit. The decoding of indexable instructions is done on page 02.12.76.1 block 1A.
7. +P TXI (02.12.76.1), +N Index Transfer Met, +N Cond. Met (02.10.07.1), -P Minus on Transfer Met (02.10.09.1), +N AR to PC (03.06.05.1).
8. Answer b. The program counter plus one is in the address switches at I11 because the "End Op Tgr." is on. AXT has a POD 76 and therefore gates the Adr. Sw. 10=17 to the shift counter at I11 even though it is meaningless and will be reset in I time of the following instruction,
9. False - The TXI is not a conditional transfer.
10. XRB = 34. Since the one's complement (and the resulting error of one) is used twice, once on a complement and once on a true number, it is self correcting and no error results.
11. I7 of the following instruction
12. When in Memory Nullification Mode (ESNT Instruction) and the 16K/24K switch is set to 24K allowing a 704 program to use 8K of memory.

13. I3 (D1) - (Normal operation of I cycle, page 03.06.05.1) and L9 (D1) Minus on TR met (03.06.05.1), Cond. Met (02.10.09.1). Index Trans. Met (02.10.07.1) and TSX (02.12.76.1)

14. XRA = 7 XRB = 0 XRC = 77775

<u>PC Step</u>	<u>XRA</u>	<u>XRC</u>	<u>TRA?</u>
000	40	40	Yes
003	40	40	No
004	33	40	No
005	33	40	No
006	33	77771	Yes
003	33	77771	No
004	26	77771	No
005	26	77771	No
006	26	77771	Yes
003	26	77771	No
004	21	77771	No
005	21	77771	No
006	21	77771	Yes
003	21	77771	No
004	14	77771	No
005	14	77771	Yes
001	14	77771	No
002	14	77775	Yes
003	14	77775	No
004	7	77775	No
005	7	77775	Yes
001	7	77775	No
002	7	77775	Yes
003	7	77775	Yes
007	7	77775	

15. a. TXH, TXL
 b. TIX
 c. TNX
 d. TIX, TNX, TXH, TXL
 e. TXI, TSX

05.08 Trap Mode - Answers

1. Block setting of the IA control trigger on transfer instructions executed in Trap Mode. This will not affect program operation because if the transfer is not met, the address portion of the instruction is not used, if successful it will transfer to location 1 and still not use the address portion.

2.
 - a. IE (See Block 3G page 08.00.02.1 and Block 2G page 08.00.00.1)
 - b. IE
 - c. IEL (See Block 5D page 08.00.16.1 and Block 5A page 08.00.12.1)
 - d. IEL
3. The program counter is stored on all transfers, the trap to location 1 is the conditional factor in trap mode operation.
4. Blocks stepping the program counter at I9 so that the true address of the transfer may be stored and allows it to step at E9 in case the transfer condition is not met.
5. They cause the decrement portion of a transfer instruction to be saved in the Storage Reg. This is necessary only on TXI, TIX, g/TNX, TXH, TXL, instructions where the conditions met will be determined during the "L" cycle following the "E" cycle forced by trap mode.
6. Only the store address line is conditioned. Page 02.09.01.1.

05.09 I/O Instructions - Answers

1. PR is + and POD 64 or 54 and PR 8 & 9 = 0; or, POD 76 and SR 23-26 = 0001_2 (or $1xxx_8$) in address field.
2. PR is - and POD 64 or 54 and PR 8 & 9 = 0; or, POD 76 and SR 23-26 = $x010_2$ (or $2xxx_8$) in address field.
3. 03F53G pin 02. See page 06.00.06.1. Block 2D, the output note #2. Referring to note 2 shows that "+P Sel. Chan. D" is brought to the above pin - Channel C is brought to pin 2 of 03F53F and E and F to 03F49D and 49E respectively.
4. Allows the BOT indicator to be turned off when tested. (I2 of the following instruction).
5. L1 (D2)
6. L1 until I5 of the following instruction.
7. No. The signal is inverted at block 3G page 02.10.80.1 to prevent skipping when condition is met.
8. Up from L1 to I5 - Brings up "Cond. Met" to transfer address reg. to program counter at L9 via the "Minus on TR Met" line (page 02.10.09.1).

05.10 SI Instructions - Answers

1. Electronic Reset is the only input control line that will affect the sense indicator regardless of the contents of the SR.
2. At any time SR1 equals one. It is not gated by instruction code or cycle time.
3. The lower trigger will latch on.
4. The one in the lower trigger is set into the upper trigger at the same time the lower trigger is reset. The capacitor in Block 1D insures that the upper trigger will have sufficient time to latch. Since pin A of Block 2I was and will remain minus, the "and" at 3F will not be conditioned.
5. The upper trigger is reset. Input pin D of Block 3F will go minus before pin C of that block will go plus.
6. As the control line goes minus, the upper trigger will not latch. Block 3F will be conditioned and cause the lower trigger to latch when pin D of Block 2E goes plus again. The capacitor at Block 2G insures the input to the trigger will remain long enough to latch it.
7. No. Only on "clear."
8. OSI
9. The "Control field" or "Mask" (18-35 of the instruction) is gated to Ad. 18-35 on a RIR and Ad. P-17 on a RIL. When the adders P-35 are gated to SR S-35, the mask appears at SI 0-17 or SI 18-35 inputs.
10. The PIA causes the acc sign to be cleared. None of the others affect it.
Page 02.12.92.1.
11. Sense Indicators 0-35 = 523456765030
12. Acc = 20₈. Sense Indicators are all zeros.

05.11 Trouble Shooting - Answers

1. A. The Accumulator is loaded with a one in position 35. Then the ability to load AC35 and shift left is checked for 4 shifts. At completion of a successful pass the AC. would contain +00000000020₈. Successful completion would leave AC=+20₈
XRA = 1.
b. The apparent trouble is that the TZE was executed prior to the XR running out. This could be caused by failure of any circuit that would prevent a shift from AC 33 to AC 32 as the XR equals 2 at the time AC 32 should be a one preventing the TZE from ending the program.

2. a. The program loads the AC. with +377 → 77 and adds 1000 → 001 which results in a +101₂ in AC P, 1 and 2. This is stored logically resulting in a -20000 → 0 in Storage. The Accumulator is cleared by shifting out and the -2000 → 0 is placed in the MQ, S → 35. A logical shift places MQ, S, 1 and 2 into the low order acc positions which will then contain 5₈. The Acc is tested to see that it contains at least a bit in position 35.

If the Acc = zero the error halt would occur at PC = 12. If bit 35 only was zero, the error halt would be at PC=10. A correct halt is PC=11.
b. Any malfunction in the following areas:

Adders and Lookahead - No carry from 3 to 2.
No input from SR pos. 2
No input from AC pos. 2

Acc - Pos. 2 cannot be loaded.

MQ - Pos. 2 cannot be loaded.
Pos. S or 1 cannot be shifted into.

3. a. PC should equal 5 as TLQ does not transfer when the MQ equals the Acc.
b. Any trouble that would cause the MQ to look smaller:
Carry adder 35 lost.
Lost bits on MQ to SR L2 D1
Fail to complement AC to SR at L time.

4. a. The program places -12 in the acc. then adds +4 for a sum of -6 octal. Subtracting a -3 leaves a sum of -3. This is compared logically with a -5 in storage. Since an LAS instruction does not consider signs as Plus or Minus, the SR will have the largest number and the computer should skip 2 instructions and halt at loc. 6.

- b. Since the computer only skipped one instruction it appears that either the L5D1 or L9D1 inputs to page 02. 11. 50. 1. Block 5H has been lost.
- 5.
- a. Ability of TIX to reduce an XR by its decrement and its ability to transfer is tested.
 - b. Since the error halt occurred at C it indicates the CAS found the accumulator larger than the word at Location A. This indicates that the transfer failed - AD 3 carry could be turned on incorrectly or Index Trans. Met. could fail to come up.

06.00 Review Question Answers - 7302 Oil Core

06.01 General Concept of Core - Answers

1. 32,768 36-bit words, 1,179,648
2.
 - a. Read out - sends to system the contents of an address specified by the system.
 - b. Store - takes a word or partial word from the system and places it in the address specified by the system.
3. The time required to make available to the system the contents of any address.
4. 1.05 usec.
5. 2.2 us.
6. The ability to obtain any core address in one memory cycle.
7.
 - a. See 2a above.
 - b. See 2b above.
 - c. Memory Data Bus In - 36 input data lines from multiplexor to the MDR
 - d. Memory Data Bus Out - 72 output data lines from MDR to multiplexor.
 - e. This instruction allows the addressed core location to be OR'ed in the MDR with the MDBI lines.
 - f. Memory Address Register - A 15 position register which contains the contents of the addressed core location.
 - g. Memory Data Register - A 72-position register which, at the end of a memory cycle, contains the contents of the core location read out (read-out operation) or the contents of the core location to be read into (store operation)
 - h. A pulse sent to core by the system which tells core to start a memory cycle.
8. A0

06.02 Ferrite Core Theory - Answers

1. 23
2. Reliability, low cost and high-speed operation
3. Contents of the addressed core location is reset to zero when read out. (Regenerated on re-write cycle of core.)
4. Draw a hysteresis loop and explain it. See figure in manual of instruction.

5. Read out core X - flips from "1" to zero - voltage induced in sense winding and amplified by sense amplifier - output of SA turns on position X of MDR.

06.03 Coincident Current Addressing - Answers

1. 128 X, 128 Y
 - a. 72 planes
2.
 - b. 0 to 71, top to bottom
3.
 - a. The amount of current ($1/2I$) passed by either an X or Y drive line "half selects" the cores on these individual lines.
 - b. When a core is forced to change its status from 0 to 1 or 1 to 0 it is said to have flipped.
 - c. Noise is the resulting voltage (usually undesirable) that is induced in a wire or other conductive material, due to a magnetic field change in the immediate vicinity. For example, when a current pulse is fed through an X winding, a noise pulse is generated in the sense winding parallel to it. Also a half-selected core induces noise in its sense winding.
 - d. The resultant of 2 half select currents ($1/2I + 1/2I$ equals full select current). This occurs where X and Y drives cross.
 - e. A process of selecting one word in the array. In the 7302 this is done by selecting a specific X and Y drive line. The intersecting point of the X and Y lines is the addressed location.
4. 4, It insures that a bit on the sense winding reaches the SA input within a fixed period of time regardless of the address selected. In order to clarify this definition assume that a single sense winding is used and a time graph is plotted as follows:
 - a. All cores in the plane to be sensed contain bits
 - b. Each address is selected and the arrival time of the signal on the sense winding is measured at the input of the sense amplifier.A study of the spread in arrival time will show that it is desirable to reduce this spread. This is accomplished by segmenting the sense windings.

06.04 Basic Core Cycle and Addressing - Answers

1. None. Memory select is the only timing pulse sent. Memory generates its own timing cycle from the memory select pulse.
2. It is mixed with 4 MAR positions in the address decoders to develop 16 lines during read and their complements during write.
3. It isn't. It is a read-write sequence.
4. 1170
5.
 - a. A type of switch which uses an addressing scheme to control its output.
 - b. A specific type of matrix switch. It uses 16 pairs of input complement lines which are wound in a specific pattern through 16 G switches. This wiring pattern permits selective addressing of any one of the 16 G switches.
 - c. A molded assembly of 16 G switches.
6. 33. Thirty-two input lines and bias winding
7.
 - a. 128
 - b. 8
 - c. 128
 - d. 8
 - e. 256
 - f. 256
 - g. 288
8.
 - a. 15-17
 - b. 4-7
 - c. 8-10
 - d. 11-14
 - e. 4-5
9. 4, Y (SX YZ the alphabetic sequence shown may be helpful to remember which lines are parallel to which).
10. A exclusive OR B, (A not equal to B)
11. This arrangement permits an addressed G switch to flip on at read time (16 of 32 lines conduct) and off at write time (the complement of the 16 addressed lines conduct).

06.05 Sense Amps - Regs - Timings - Answers

1. 72, 15, 72
2. It is used to gate the sense amplifiers at a time in which the sense winding gives the greatest noise-free output.
3. A strobe generator is a group of circuits, arranged to provide properly timed strobe pulses for all planes.
4. 11, 12, A = 11, 12 B = 11, $\overline{12}$ C = $\overline{11}$, 12 D = $\overline{11}$, $\overline{12}$
5. 9, 15
6. MDBO, Z driver control line
7. See systems 00.01.01.0
8.
 - a. Sense winding segmentation
 - b. Sense winding figure 8 wiring pattern
 - c. Delay the Y drive at read time
 - d. Adjacent cores are physically arranged at 90° angles to each other.
 - e. Sense and inhibit windings are perpendicular to each other
 - f. X and Y drive lines are skewed
 - g. Controlling core temperature.
9. They are never up simultaneously.

06.06 Special Circuit - Answers

1. MAR, MDR, sense amplifier, core drivers
2. "+ Mem Select" and "- MAR" line, "+ Mem Select" and a "+ MAR line"
3.
 - a. -2volts
 - b. +0.4 volts
4. When it is set at Mem Select time which is A₀ in the system
5. Approx .4 usec following Mem Select
6. -MDBI line, +DIG
7. It isn't. A sense amplifier input is all that is needed to set a MDR position.
8. Conditioned for the full memory cycle by the RO trigger (01.02.04.1, 4I)

9. Approx. .94 usecs following memory select and up throughout the cycle.
10. Drawing should show a supervisory input AND, a trigger and an output AND. See figure in manual of instruction.
11. A differential amplifier is used because it can be driven by either the positive-going or the negative-going sense winding output signal.
12.
 - a. The emitter follower couples transformer T1 to the strobe gated amplifier and provides drive current for this amplifier.
 - b. It blocks a sense amplifier output when required by a store operation and it provides a drive pulse and drive current for the level setter.
 - c. It acts as a clipper to its capacitive input pulse and develops a +P bit output and a -P no-bit output.
13.
 - a. Array winding, 90 ohm terminating resistor to +60v
 - b. G switch primary, 56 ohm terminating resistor to +30v
 - c. G switch primary, 56 ohm terminating resistor to +30v
14. To prevent the power TX from being driven to saturation. This allows for a fast turn-off of the power TX.
15. It requires a +level from the output of the Y address decoder and a +level from the segment gating.

06.07 CE Panel and Temp. Control - Answers

1. False. This line is not used in the 7090 system
2. The Z driver timing trigger turns on the timing error trigger, which blocks all inputs conditioned by mem select and provides a reset for both the Z driver timing trigger and itself.
3.
 - a. Prevents the Z drivers from conducting
 - b. Turns on the error-check trigger which signals CPU to continually step MAR through successive addresses
 - c. It is used in conjunction with the check On-Off switch to establish what the bit configuration should be in each address tested. If an address tested does not agree with the check Ones-Zeros setting, the error check trigger is turned off and memory cycling is stopped with the error address in MAR and the contents of the error address in the MDR.

- d. Blocks the strobe of an address. Thus during a mem cycle the MDR is reset normally and its zero eontents are written back to core.
 - e. Conditions the switches on the CE panel so that they are active. With this switch off the other switches are inactive.
- 4.
 - a. One of the Z drivers in conducting continuously
 - b. Low oil pressure or oil in the heater line is 115 degrees F (excessively hot)
 - c. All DC voltages to memory are up.
 - d. Error was recognized while testing
 - e. This indicator is not used on the 7090 system. When used on the 7030 system it checks for a correct parity count of MAR and for correct decoding of the X and Y decoders.
 - f. On when the test switch is on (test status).
 - g. A blown fuse, loss of -48v or an open thermal.
- 5. 102, 3, 2
 - 6. 110, 98
 - 7. There isn't any. The fan is powered directly from the supply voltage.

REVIEW QUESTION ANSWERS

08.00 DATA CHANNEL

08.01 INTRODUCTION AND PROGRAMMING

1. 721 Punch - 716 Printer - 729 Tape Units
2. 711 Card Reader - 729 Tape Units
3. Refer to IRM, Page 5, "Buffering on the IBM 7090"
4. Eight
5. Refer to IRM, Page 5, "Physical Description"
6. Write Select Instruction
7. Refer to IRM, Page 8, "Selection of Channel and I-0 Unit"
8. Select; proper channel - correct unit - and put it in motion reading
9. True
10. True
11. Select instruction will hang up in CPU
12. To load the channel with an I-0 Command
13. Refer to IRM, Page 8, "I-O Command Operation"
14. S, 1, 2, 19 - Operation Code
3 through 17 - Word Count
21 through 35 - Initial Address
15. No, strictly in channel
16.
 - a. One 3 word record
 - b. One 3 word record
 - c. One 3 word record
 - d. Two 3 word records
 - e. Two 3 word records
17.
 - a. One 3 word record
 - b. Two 3 word records
 - c. One 3 word record
18.
 - a. Space 3 3/4 inches of tape
 - b. 3 word record no spacing
 - c. Two 3 word records - IORP does not space tape in this example
 - d. One 3 word record preceded by 3 3/4 inches of blank tape

19. a. ECW - The E cycle of either an RCH or LCH during which we load the Op Reg, W.C., and CAC.
- b. BCW - A storage reference cycle initiated by channel during which we load the Op Reg., WC, and CAC.
- c. BDW - A storage reference cycle initiated by channel for the purpose of Data transfer between the DR and storage.
20. I-O Check indicator will be turned on and LCH will function as a No-Op.
21. I-O Check indicator will be turned on but the RCHB will load the channel.
22. IOCT will write one word and when it does not find an LCH waiting, will disconnect.

No I-O Check

23. None
24. a. IOCP 1st record 2 words of 2nd record and proceed
 IOSTN last word of 2nd record and look for LCH
 IOCD 3rd record and 2 words of the 4th record
- b. IORP 1st record only
 IOCD 2nd record
- c. IORP 1 word of 1st record
 IOCD 2 words of 2nd record

	RTBA	
	RCHA	ALPHA
	LCHA	ALPHA + 1
	CLA	
	ADD	
	HTR	
ALPHA	IOCT	COCA, , 50
	IOCD	COCA + 50, , 50

08.02 FUNCTIONAL UNITS

1. Chan Input switches - Calc Entry - Tape Register
2. a. CAC
- b. LC
- c. WC
- d. CAC
- e. WC
- f. LC
3. S Count Control
 - 1 Record Control
 - 2 Transfer Control
 - 19 Transmission Control

4. Location of the next command
5. Contents are used to locate data in storage
6. Group counter
7. Refer to IRM, Page 20
8.
 - a. ECW cycles
 - b. Load LC (manual)
 - c. TCH command
9. Refer to IRM, Page 19 (LR CR)
10. 2 refer to IRM, Page 19
11.
 - a. MDR
 - b. MXSB
 - c. CHSC
 - d. CH, IN, SW
 - e. DR, OR, WC, CAC

WRS RCH BDW

1. Not Data Selected, Not Channel Interlocked, Channel Address and Secondary Op Write
2. It indicates that channel has accepted the "Select" and tells CPU that it can now End Operation. Refer to IRM, Fig. 9.
3. Ready - Not File Protected (Sel Rdy and Wr)
4. We use this Non Data Disconnect to reset the channel unit select triggers so as to allow "Stacking".
5. Millisecond
6. Yes, if the Channel addressed does not exist or is in manual.
7. Yes, if the RCH is executed and the addressed channel is not Data Selected
8.
 - a. Get the contents of its specified address and load them into the Op Reg, WC and CAC
 - b. Load and advance the LC
 - c. Refer to IRM, Fig. 10
9.
 - a. DR Not Loaded
 - b. WRS
 - c. WC not equal to zero
 - d. Refer to IRM, Fig. 11

10.
 - a. Turn on the priority trigger
 - b. Refer to IRM, Fig 11 and 13
11.
 - a. Load Data Reg with data word
 - b. Indicate that DR is loaded (turn on DR LD tgr)
 - c. Step CAC and WC
 - d. Refer to IRM, Fig. 11 & 12
12. a, b, c, d, (except Ind. 18) and e
13. End Op, Share, and Interrupt
14. End Op Control
15.
 - a. Retain priority line to channel C kills channel C
 - b. 60.70.02.1 - Ret. Pri. 5B, 4A, 4B, 3C
16. Not Remote Required
17. Because there may be a sense instruction following that will make use of the Channel Unit Select Tgrs.

08.03.02 WRITING

1. A delay counter output determines when tape has been in motion long enough.
 - WD 50 Not at LP
 - WD 320 at LP
2. To allow channel to get another data word.
3. No, Write Clock Cntl tgr does the job. Note: The indicator for this tgr is labeled "WR Gate" on the console
4. One, any echo returned from tape is acceptable
5. The Demand Gate Tgr is turned on during the 5th WC cycle
6.
 - a. Reset the TR
 - b. Gate new word from DR to TR
 - c. Generate Tape Demand so we can get new word into DR
7. False
8. False
9. To develop correct character spacing for dual density on 729 Mod II and IV Tape Drives.
10. Write First Word Test

11. Tape Group Counter

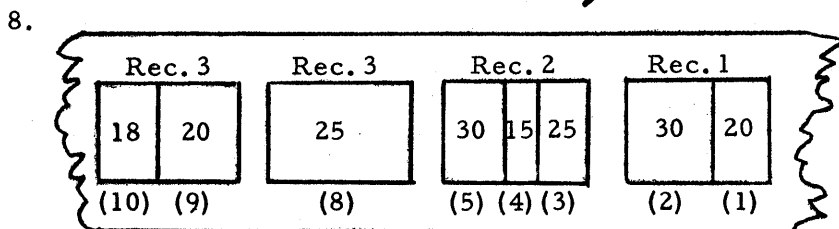
08.03.03 DISCONNECT

1. False
2. It will allow a Data Disconnect to take place after completion of Read checking the record.
3.
 - a. Space to a position to write the check character
 - b. Write the check character
 - c. Assure that tape continues to move far enough so as to be able to read check the remainder of the record
 - d. Determine when we have completely read checked the record and activate Data Disconnect
4. When the DC produces a WDD 60
5. WDD 20
6.
 - a. Block reset of RDD
 - b. Reset RD COND
 - c. Sample LRCR
 - d. Reset Tape Unit Select tgr.
 - e. Data Disconnect if Data Disconnect Gate Tgr is on

08.03.04 READ CHECKING & MISCELLANEOUS

1. True
2. 1st bit drops before clock completes a cycle. RC 4 tgr input insures clock steps to zero.
3. False - Skew B is not VRC checked
4. True
5.

(1)	e	(7)	l
(2)	b	(8)	f
(3)	c	(9)	m
(4)	d	(10)	n
(5)	h	(11)	m
(6)	d, k, o		
6. seventeen
7. I/O check disconnect 60. 38.02.1 (3F) sets disc gate tgr.



9. Record Control Pulse - A7 to A5 (60.36.02.3)
10. Record Control Sync Tgr. (60.36.02.3)
11. -A (2F) 60.80.03.1
12.
 - a. True
 - b. False
 - c. False
 - d. False
13.
 - a. False
 - b. False
 - c. True
 - d. False
14.
 - a. False
 - b. True
 - c. False
 - d. False
 - e. False

08.03.04 TAPE READ SELECT OPERATION

1. Not Data Selected - Not Channel Interlocked - Channel Address - Sec Op Read
2. Ready and in Read Status (Sel, Rdy & Rd)
3. Until the DC reaches a value of RD 30 or RD 160 is started at LP
Equivalent times can be found in your CE Handbook
4. Because information must pass through the DR input circuits to reach the Op Reg, WC, and CAC. Refer to ISD's for details.
5. 501
6. 2 - High Clip and Low clip *Refer to IRM, Page 114 for details of the final amps.
7. 1st Bit from Skew Reg A
8. Read Clock Gate Refer to IRM, Page 59
9. Gate Skew B to the R/W reg. and LRCR
10. End Rd Delay - - As soon as 1st character is read
11.
 - a. Sample Skew A VRC
 - b. Gate Skew A or B to the R/W Reg and LRCR
 - c. Turn on RDD
 - d. Step Group counter
 - e. Reset Skew registers with a delayed RC 7

12. RDD will be running during all but the 1st RC cycle
13. RC 3 of the 2nd character
14. Turn off RDD - - Reset R/W Reg. so that next character can be gated in at RC 7.
15. Group counter will have been stepped to zero and the Demand Gate tgr. turned on
16. In the R/W Reg.
17. The following RC 3 or RDD 36
18. The Tape Demand caused by: RC 3 RDD and Demand Gate
RDD 36
19. Refer to IRM, Fig. 33, the fall of Tape Demand
20. DR LD on, RDS, WC not zero
21.
 - a. Gate DR to SB
 - b. Turn off DR LD tgr.
22. There is no more RC 4's to turn off RDD and it continues to run the delay counter.
23.
 - a. Gate last character to the TR
 - b. Sample R/W Reg VRC
 - c. Gate TR to DR
 - d. Indicate that DR is loaded (turn on DR LD)
 - e. Set check character tgr.
24. RDD 136
25. RDD 36
26.
 - 4
 - a. R/W Reg. VRC
 - b. CC 2
 - c. TWI
 - d. LRRCR
27. When Cntl Wd Gate becomes active (RDS and WC equal zero)
28.
 - a. Yes
 - b. Yes *Refer to IRM, Page 65, "Skipping Commands"
 - c. No
29. There is a special circuit that steps WC and CAC (Ind 19 and Sync Gate)

- 30. a. No
b. No
- 31. Immediate Data Disconnect
- 32. Refer to IRM, Fig. 37

08.05 TCH-IA OPERATION

- 1. A TCH is an unconditional transfer which allows a programmer to transfer somewhere else for another portion of his IO program.
- 2. In multiplexor so we can extract its address portion and route it directly to MAR.
- 3. It allows the channel executing the TCH to keep priority and also activates BCW Required.
- 4. The address portion of the TCH command, by bringing up a gate which allows us to take positions 21-35 of the Mx Stg Buss to the Mx Adr Sw.
- 5. By looking for a bit in position 18 of commands.
- 6. 2
- 7. Yes
- 8. IORP
- 9. It will completely tie up the system since this channel will retain priority constantly.
- 10. 2

08.03.06 NON-DATA SELECT OPERATIONS

- 1. Not channel interlock, Channel address, Sec op rewind
- 2. The tape unit will send the signal "Select and Rewind" back to channel after the rewind relay has been picked
- 3. Yes, the tape unit must write forward before going into read status & rewinding.
- 4. Refer to IRM, Fig. 45
- 5. Disconnect on the rewind and turn on the BOT indicator.
- 6. The tape will move backward to LP and the BOT indicator will be turned on.
- 7. The backspace file gate. tgr.
- 8. Refer to IRM, Fig. 47, 48, & 49

9. The class address - Refer to IRM, Fig. 50
10. Yes - Execute an SDN to a tape unit that is not ready

08.07 DATA CHANNEL TRAP

1. It allows automatic monitoring of channel operations by permitting channel to signal CPU by trapping when various conditions arise.
2. Redundancy Check - Reading an EOF - Command Word situation (T type command did not find LCH waiting).
3. Through the use of the Enable instruction
4. To disable all channels
5. No
6. Yes
7. Refer to IRM, Fig. 56
8. False
9. Enabled for CWT_n & transfer type command (IOCT, IORT, or IOST) finds no LCH waiting in L time.
10. EOF Sync., Enable CWT or EOF, Restore
11. False
12. Channel trap tgr. set at I6 only.
13. False (Inhibits only)

08.08 CONSOLE OPERATIONS

1. Four Keys
 - a. Auto/Manual in manual
 - b. On/Offline in off line
 - c. Load Data Reg
 - d. Write Tape
- Two Switches
 - a. Tape cycling on
 - b. Unit select switch in proper position
2. Zero - Use positions 21-35 to load the Loc Ctr.
3. Load Command holds the DR reset so that if you loaded it first you would just wipe it out.

4.
 - a. Auto/Manual switch in manual
 - b. On/Off line switch in on line position
 - c. Set up entry keys with IOCD 1000, , 30
 - d. Depress Load Command
 - e. Depress write punch key
5. You won't. There is no indicator for Op Reg 18 on the channel console.
6. The reset key resets all the triggers in channel. It also functions as a Stop Read tape key.
7. False, not unless you are in cycling mode.
8. Yes, it will cause the operations to be repeated over and over.
9. End of tape will be sensed and a Rewind operation will be initiated
10. Yes

08.09 DATA CHANNEL MISCELLANEOUS

1. A tape demand occurring while the DR is still loaded and WC not = zero.
2. Yes
3. No
4. False
5. Not unless you execute and IOT every time you use a channel. There is only one I-O Check indicator on the system.
6. False
7. False

08.11 9T55

1.
 - (1) A
 - (2) B
 - (3) D
 - (4) L
 - (5) D
 - (6) G
 - (7) B, A

08.12 INTRODUCTION TO CAU

1. Model 1 or 3
2. It is powered by a standard SMS power supply in the 7607 it is put in
3. Change the card machine voltage levels to N or P levels used in transistorized computer.
4. Five. Read Card Reader, Write Punch, Write Printer Write Printer Binary, Read Printer.

08.13 BASIC CAU

1. The first CB set pulse from the selected card machine
2. 4 usec
3. Card Drive pulses
4. 32
5. Once for each row or 12 times
6. All Card Ring triggers off and Sample pulse.
12
7. CB reset pulse between cycle points
8.
 1. End of Row
 2. End of Record
 3. Data Disconnect
9. Card Ring trigger one

08.14 CARD MACHINE POWER DISTRIBUTION

1. In the HS -- card which develops drive current to pick a magnet or relay.
2. False - dropping power to channel only prevents the distribution of 46V and 55V to the card machines.
3. False - controls both 46V and 55 V.
4. True

08.15 READ CARD READER

1. The select trigger ON in CAU.
2. Fourteen. 13 from CB reset and 1 from EOR reset
3. Thirteen. 12 from CB set and 1 from EOR set.
4. To keep the Channel Unit triggers free for sense instructions by preventing any Data-Select or Non-Data Select from Ending Operation in channel.
5. To generate an EOR pulse to be sent to channel and to keep the Card Machine moving if another card is to be processed.
6. Read pulse control, Read Gates (Left and Right) and Sample pulse.
7. No. Indicator 19 ON prevents generating any read pulse.
8. Its output is gated with an after 12 CB line.
9. End of Record Pulse.
10. Data Disconnect and the drop of Card reader select picks R20 in reader which opens up the Brush Isolation points to prevent even reading the card.

08.16 WRITE PUNCH

1. Card Punch Selected trigger ON in CAU.
2. Write Pulse Control, Write Gate (Left and Right) and Sample pulse.
3. A CB in the punch must provide a gating line to the Calc exit blocks for every row which holds up the lines over to punch.
4. Fourteenth CB reset from the EOR reset CB.
5. P-31
6. P-32
7. Allow a CB reset before we disconnect the punch.
8. Never select the punch. Would I/O check when RCH arrived with selected.

08.17 PRINTER OPERATIONS

1. Write pulse control is up for only the 1 row time. This is controlled by Printer Binary trigger, Printer Write Select and ON 9 CB.
2. Yes
3. Eleven - Once for each echo time.
4. Until after 8 CB time.
5. Turned off each row by rise of Write Right Gate.
6. To insure that the BDW cycles for 1 row, 0 row, 11 row, and 12 row echoes are controlled by read circuits.
7. After 19 CB or the 20th row.
8. After 12 CB line blocks the turn ON.
9. The BDW cycle will be requested as soon as the Channel Write Cards trigger is turned ON by the Card Control trigger for the 11th row processing.
10. -P After 8 CB line is always minus. This line feeds pin E of 06A4F04 (5B) on 80.20.03.1. To prove this answer you will have to use the 716 mechanical timing chart located on page 49 of the 711, 716, 721 Manual of Instruction.

08.18 SENSE OPERATIONS

1. Modify card machine operation by use of CPU Instructions.
2. The sense instruction will set a particular Unit Trigger in channel. If the printer is selected this unit trigger will cause a sense exit to be hot corresponding to the Unit Trigger. If this exit is wired in the printer to the overflow hub you will initiate the overflow.
3. Two, CP1, CP2
4. One, from the printer.
5. Cause overflow, suppress overflow, suppress spacing, double space or suppress printing.
6. EOR and Channel Unit Select trigger on.
7. +A at 5A + 5B - allows non-data disc. To reset channel Unit Sel Trgs. so they will be free for PSE Instruction.
+A at 5C - allows non-data disc. when we are through punching so Channel Unit select trgs. will be free for RDS CR Instruction.

8. Normal I, L, end op.

08.19 MANUAL OPERATIONS

1. Systems 80.60.01.1 - A at 3A with appropriate gate causes Card Control trigger and Card Sync Trigger to be reset. Now we cannot gate card drive pulses to step card ring.
2. When the first three words are read into 0000, 0001, and 0002, the IOCP proceeds and we get a BCW cycle. Load Control trigger on and BCW tells the system the first instruction has been sent to location 0001.
3. Interlock Reset
4. With Switch on, depressing Write Printer causes a Write Printer Binary operation to be performed.
5. No. There can be only one Card Reader per Data Channel. It is not necessary to set a Channel Unit Select Trigger.

SIMULATED MACHINE ERROR ANALYSIS PROBLEM ANSWERS

1. There Is No Tape Unit On #5

First of all, CPU is hung up in L time; the PC indicates that we have not Ended operation on the REW instruction. Investigation of the channel console shows that channel is interlocked since "Channel Unit Select" #5 is still on. This explains why the REW did not End Op. Now why is the Chan Unit Sel Tgr still on?--It should have been reset by "Non-Data Disc" which is activated when "Busy" comes up in the tape control circuits. Further investigation reveals that the "Write Tgr" was not turned on, therefore not activating Busy. Since "Sel & Rdy" were missing, it would prevent the turn on of the Write Tgr.

2. Write 1st Word Test Missing

Tape is in motion and it appears as if we should be writing since the Write tgr and the Wr Tgr Release Tgr are on. The fact that the Wr Cond. tgr is on is an indication that we have reached WD 50 or 320. At this point, the 1st data word should have been transferred to the TR since the DR was loaded. Note that the TR was not loaded. There are other indications that also point to Wr 1st Word test such as: The WC is not running and Tape Demand has not been generated since we took only one BDW cycle.

3. Tape Demand Has Not Occurred

In this problem the WC is running and the R/W Reg indicators are blinking, which is an indication that we are writing something. Further investigation reveals that the words in the TR and DR are the same. Looking at the WC indicates that the 1st word is being written over and over. Since tape demand did not occur at Wr 1st Word test time, we did not initiate the second BDW cycle.

4. Demand Gate Tgr Never Turned On

Note that the WC is running and the R/W Reg indicators are blinking, which are indications that we are writing. We are writing the 1st data word over and over, but the 2nd data word is in the DR; as evidenced by the word counter. We did not recognize when the 1st word had been written.

5. Disconnect Tgr Never Turned On

In this problem, since the WC has been reduced to zero and the Data Disc Gate tgr has been turned on, we can assume that the 4 word record was written. However, since the Wr Tgr Release tgr is still on, we have not written the check character. Since Wr Cond. is still on, we are led back to Disc Call; which is the output of the Disc tgr.

6. WDD 20 Is Missing Except At The Go Tgr Reset

Here again we can assume that the record and its check character were written, since the WC=0, the Data Disc Gate is on, and the Wr Tgr Release tgr is off. Note that the delay ctr is running because of the WDD tgr. We must have gotten a WDD 60 to write the check character but did not get the WDD 20 which resets the WDD tgr and the delay ctr.

NOTE: The particular circuit can be seen in your ISD's where WDD 20 is developed.

7. RDD 144 Missing

Here again we can assume that the information was written and investigate why we did not Data Disc. It looks like WDD operated correctly, since the Wr Tgr Rel tgr and the Go tgr were reset. The Data Disc Gate tgr is on and the delay ctr. is running for RDD. We must have advanced as far as RDD 36 and 128 because the Check Char tgr is on and RD Cond is off. RDD 144 being missing would cause a failure to Data Disc along with failure to reset the Check Char, WR, and Disk triggers.

8. Cannot Turn On The Data Disc Tgr.

In this problem the best clues are that the WC was reduced correctly and the Data Disc Gate tgr is on; which are indications that the information was written and an attempt was made to disconnect but we could not. This much is fine but we must go further to explain the other indicators; such as why is the Write Delay trg on? Remember what will take place when in writing a record we reach RDD 144 and do not want to Data Disc (example: IORP). What has happened here is that we write the record, and tried to Data Disc but were unable. Now as far as the tape control is concerned, it thinks channel wants to write another record; but since there is nothing to write, we must "Space Tape" over and over.

9. Control Word Gate is Missing

Investigation of the indicators will show that the IOCP is still in the Op Reg but it is WC=0. The "bug" shown above would prevent the IOCP from activating BCW Req. when its WC was reduced to zero. To explain the other indications channel is "Spacing Tape" repetitively.

10. Record Control Pulse Missing

Investigation shows that we are operating on the IORP; it is WC=0 but it has not proceeded to the IOCD to disconnect. Since the IOCP did proceed correctly, we can assume that the trouble is not in the BCW Req circuitry. Record Control Pulse is the most logical trouble that is peculiar to an IORP.

11. Nothing Was Written On Tape

You should have found two obvious clues during investigation of this problem. One--Why didn't we Data Disc? Two--What caused the Echo Error? As far as the Disconnect problem, you should agree that RDD never took place. Echo errors are caused by not "Flipping" any write tgrs in the tape unit. To correlate these two indications, realize that if nothing was written there will be nothing to read and nothing to start the RC which is necessary before we can ever turn on the RDD tgr.

12. EOR Produced Unit Reset On IORP

Close observation should show you that channel is trying to execute the IOCD, but that there is no tape unit select trigger on. In writing the 3 word record using the IORP, RDD 144 occurred; but would not have produced a Unit Reset since the Disc Gate tgr is still on. Refer to ISD 60.36.04.1.

13. RC Sync Tgr Never Turned On

In this problem the only clues you have are the fact that there were only two records written and the contents of the LC. Note that the program should have written 7 records; the fact that the LC is at 7 indicates that an attempt was made to use all the commands.

If you could watch this operation, you would note that the value of the LC after writing the 1st and while writing the 2nd record would not be setting at 1 where it should be. Since the R C Sync tgr did not turn on, it allowed the generation of multiple Record Control Pulses at the end of the 1st record which caused channel to "skip" the IORP's.

This is a good example of a trouble where assumptions or calculated guesses must be made.

14. End RD Is Missing

Note from the indicators shown that everything has progressed correctly up to the point where we should have set RD COND, Read Gate, and reset RD.

15. Read Gate Did Not Turn On

Probably the best approach to this problem would be to determine why nothing was read into storage but yet the word counter was reduced. Note in your IRM, Fig. 38, that Read Gate must be present to allow TR to DR gating but it does not interfere with the generation of tape demands which will initiate BDW cycles and store an empty DR.

Also note that we have not Data Disconnected even though the WC=0 Read Gate is needed here also.

16. R C 3 RDD Is Missing To Produce Tape Demand

Note that the console indicators tell you next to nothing in this example. Now, why didn't we store each word of the record rather than just the last. The only logical reason is that there were no tape demands to indicate that a word was read until the end of the record and we generated one at RDD 36.

7090 LAB GUIDE

7090 LAB GUIDE.

The projects in this guide were designed to aid you in learning methods and techniques for maintaining a 7090 system. It is not intended to act as a source of knowledge--it is a guide for hands-on experience to supplement the classroom lecture.

The contents of this guide plus a brief description of each project are in the following order (your instructor will assign these projects to you):

LABORATORY PROJECTS

1. SMS Frame Orientation Project - This project will be used to help you gain a better understanding of the physical features of the 7090 system.
2. Wire Wrap - This project provides a series of wire routines which will give you an opportunity to try your skill at most of the possible wire wrap situations.
3. 7151 Console Orientation - The four parts of this project plus the questions which follow will lead you thru the basic operations of the 7090 operator's console.
4. Oscilloscope Applications - The six projects in this section will give you an opportunity to see 7090 type pulses under normal circumstances and to apply the scope to problems of increasing difficulty and complexity. They should help cement your understanding of component circuits and of the addressing scheme of the system.
5. Power Supply - This package will serve as a list of power supply components. You will be given lab time with power off the system and you will use this opportunity to become familiar with the location of the components designated in this project.
6. 7617 Console Orientation - This package will provide you with an opportunity to use the data channel console to control tape operation and card machine operation.
7. Diagnostics - The lab package for diagnostics consists of two portions. The first portion is made up of reference material which will help you to understand how to operate the various diagnostics available on the 7090 system. The second portion is a lab project which will provide you with an opportunity to apply what you learned from the first portion.
8. Oil Core Scoping Project - This project will allow you to become familiar with the locations of the various components of the oil cooled core and to see representative waveforms during normal operation of the device.
9. Air Cooled Core Familiarization - This project will act as a guide to help you become aware of the various important locations associated with this unit.

7090 SYSTEM PROJECTS

These projects contain a series of descriptions of machine situations. In each case, a cover sheet will tell you the number of the project, what procedures were used on the machine, and what results were obtained. The results are shown in the pages following the cover sheet. They appear in the order of occurrence. Three different types of result are provided in these projects:

1. Diagnostic Print-Outs - Where more than one print out relates to a given project. They are lettered and are presented in alphabetical sequence. In all cases, the diagnostics were read into the machine with the bug already on.
2. Pick-Sheets - Two different forms are used for this purpose. One form is used to record the status of indicators on the CPU console during a halt or hang-up condition. The other form serves a similar purpose but is designed for the Data Channel Console. Where more than one pick-sheet relates to a given project, they are numbered and are presented in numeric sequence.
3. Dumps - Dumps of appropriate areas of storage are presented where they are required.

Please note the following convention which is used on Pick-Sheets for the Data Channel Console and the CPU Console.

Indicator appears to be on SOLID -



Indicator appears to be FLICKERING -



The following are the project numbers with the portion of the 7090 system that they apply to.

- 10 thru 12 - Preliminary Instructions
- 13 thru 18 - Data Channel (This includes Date Channel, TAU, and CAU)
- 19 thru 21 - Advanced Instructions

LABORATORY PROJECT 1

SMS FRAME ORIENTATION

CHECK
OFF
BELOW

DO EACH OF THE FOLLOWING

- () 1. LOCATE THE FIRE EXTINGUISHER IN THE MACHINE ROOM.
- () 2. LOCATE THE EMERGENCY POWER OFF CONTROLS.
- () 3. FIND EACH OF THE FOLLOWING FRAMES. NOTE THE NUMERICAL DESIGNATION (FRAME TYPE) AND THE LOCATION DESIGNATION (FRAME NUMBER) FOR EACH.

FRAME NAME	FRAME TYPE	FRAME NUMBER
() CPU I	7108	01
() CPU II	7109	02
() MULTIPLEXOR	7606	03
() CORE STORAGE	7302	01
() DATA CHANNEL A	7607	06
() CHANNEL A CONSOLE	7617	09
() DATA CHANNEL B	7607	06
() CHANNEL B CONSOLE	7617	09
() CPU CONSOLE	7151	08
() CARD PUNCH	721	
() CARD READER	711	
() PRINTER	716	

- () 4. USING ONE OF THE DATA CHANNELS, LOCATE AND EXAMINE EACH OF THE FOLLOWING ITEMS.

() GATE A	() TOWER LATCHES
() GATE B	() GATE LATCHES
() GATE C	() TOWER CASTERS
() GATE D	() GATE CASTERS
() TAILGATE E	() BLOWERS
() TAILGATE F	() LEVELING PADS

- () 5. USING THE SAME FRAME, FIND EACH OF THE ITEMS LISTED BELOW.

() 2 POWER SUPPLIES	() 4 GANGS OF BLOWER CBS
() 4 CONVENIENCE OUTLETS	() 2 GANGS OF POWER CBS
() 2 POWER ON-OFF SWITCHES	() 2 FUSE LOCATIONS

- () 6. USING WHAT EVER FRAMES ARE REQUIRED, FIND EACH OF THE FOLLOWING. IF YOU REQUIRE ASSISTANCE IN DISTINGUISHING ANY OF THESE ITEMS CONSULT YOUR INSTRUCTOR.

() LAMINAR BUS	() HINGE CONNECTORS
() VOLTAGE CHAIN	() EDGE CONNECTORS
() VOLTAGE OVERLAY	() INTER-PANEL CONNECTORS
() GROUND PLANE	() T ROW
() INSULATION PLANE	() SYSTEM POWER OFF SWITCH

-- INSTRUCTIONS --

IN EACH OF THE FOLLOWING YOU ARE GIVEN TWO POINTS IN THE SYSTEM WHICH ARE ELECTRICALLY COMMON. IN EACH CASE, DETERMINE BY OBSERVATION THE INTERMEDIATE POINTS THROUGH WHICH THIS CIRCUIT PASSES. ENTER THESE POINTS IN THE SPACES PROVIDED. YOU MAY USE REFERENCE MANUALS IF NECESSARY.

- () 7. YOU MAY TRACE THIS CIRCUIT IN EITHER DATA CHANNEL.
STARTING POINT 06B 1G23 C

ENDING LOCATION 06F 53F16

- () 8. YOU MAY TRACE THIS CIRCUIT IN EITHER DATA CHANNEL.
STARTING LOCATION 06B 3B09 H

ENDING LOCATION 06B 2H07 F

- () 9. YOU MUST USE CPU FOR THIS CIRCUIT.
STARTING LOCATION 02B 1G18 A

ENDING LOCATION 01D 4J08 D

LABORATORY PROJECT 2

WIRE WRAP

1. WIRE BETWEEN THE POINTS SHOWN BELOW. MAKE GOOD SOLID WRAPS WITH NO OPEN WRAPS, NO OVERWRAPS, AND NO SHINERS. THERE SHOULD BE AT LEAST A QUARTER TURN OF INSULATION ON THE PIN. EACH WIRE IS TO LAY CLOSE TO THE PANEL AND WIRES SHOULD BE TIGHT BETWEEN PINS.
2. CHANGE THE TYPE OF TOOL YOU ARE USING SO THAT WHEN THE PROJECT IS COMPLETE YOU WILL HAVE USED EACH TYPE. (HAND TOOL, RATCHET GUN, POWER GUN).
3. WRAP BETWEEN THESE POINTS.

FROM	VIA	VIA	TO
F17C	F17B	E15Q	D15C
D15C			B15G
B15G	B16F	B18B	B22B
B17B			B26B
B26B	B27F		D27E
D27E	E27Q	F25B	F24C
F24B			F17B
B14B			B13C
B14B	B13B		D13D
D13D			F13D
F13D			F13E
D13D	C13R	B02N	B02B
D13B	D12C	E02B	F02D
F02D			F01C

4. WHEN YOU HAVE COMPLETED THE WIRING INFORM YOUR LAB INSTRUCTOR. HE WILL INSPECT YOUR WORK.
5. AFTER YOUR WRAPS HAVE BEEN INSPECTED REMOVE THEM WITH AN UNWRAP TOOL.
6. CLEAN UP THE AREA AND RETURN ALL TOOLS TO THE TOOL BOX.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

PART I - LOAD CARDS, DISPLAY, CLEAR

PART II - SINGLE STEP, MULTIPLE STEP, RESET,
START, ENTER INSTRUCTION, ENTER MQ

PART III - DISPLAY EFFECTIVE ADDRESS

PART IV - CONTINUOUS ENTER INSTRUCTION

PART V - QUESTIONS

PART VI - APPENDIXES

NOTE - TWO PREPUNCHED IBM CARDS ARE REQUIRED TO PERFORM THIS PROJECT.
IF YOU DO NOT ALREADY HAVE THESE CARDS, OBTAIN THEM FROM YOUR LAB
INSTRUCTOR.

NOTE - THE FOLLOWING SWITCHES AND KEYS ARE NOT DEALT WITH IN THIS PRO-
JECT BECAUSE THEY ARE NOT CONSIDERED APPROPRIATE TOPICS AT THIS
TIME IN THE COURSE. - DISPLAY INDICATORS, I/O INTERLOCK SWITCH,
B CYCLE SWITCHES, LOAD TAPE BUTTON.

LABORATORY PROJECT 3

7151 CONSOLE ORIENTATION

PART 1 - LOAD CARDS, DISPLAY, CLEAR

1. DEPRESS THE CLEAR KEY ON THE MAIN CONSOLE (7151) IN ORDER TO SET CORE STORAGE TO ALL ZEROS.
2. PLACE CARD NUMBER 1 IN THE HOPPER OF THE CARD READER (FACE DOWN, 9 EDGE FIRST)
3. DEPRESS (BUT DO NOT HOLD) THE START KEY ON THE CARD READER. THE CARD READER WILL TAKE ONE MACHINE CYCLE AND STOP WITH THE CARD JUST IN FRONT OF THE FIRST OR UPPER BRUSHES. THESE BRUSHES ARE NOT USED FOR READING DATA ON THE 7090 SYSTEM.
4. DEPRESS (BUT DO NOT HOLD) THE START KEY A SECOND TIME. THE CARD READER WILL TAKE A SECOND MACHINE CYCLE. THE CARD WILL PASS BENEATH THE FIRST OR UPPER SET OF BRUSHES AND STOP JUST BEFORE THE SECOND OR LOWER SET OF BRUSHES.
5. DEPRESS THE START KEY A THIRD TIME. NO MECHANICAL ACTION WILL TAKE PLACE BUT THE READY LIGHT ON THE CARD READER WILL COME ON. THIS LIGHT INDICATES THAT THE CARD READER IS NOW READY FOR OPERATION BY THE COMPUTER. THIS IS THE NORMAL SEQUENCE OF EVENTS THAT OCCURS WHENEVER YOU START A DECK OF CARDS INTO THE CARD READER
6. NOTICE THE FEED KEY ON THE CARD READER. THE FUNCTION OF THIS KEY IS TO PASS THE CARDS THROUGH THE FEED INTO THE STACKER WITHOUT THEIR BEING READ.
7. DEPRESS THE FEED KEY. NOTHING SHOULD HAPPEN BECAUSE THE CARD READER IS READY. (READY TO READ CARDS UNDER COMPUTER CONTROL AND THEREFORE IGNORING MANUAL COMMANDS)
8. DEPRESS THE STOP KEY. THE READY LIGHT WILL GO OFF. THE CARD READER IS NOW NO LONGER AVAILABLE TO THE COMPUTER BUT IT IS AVAILABLE FOR MANUAL OPERATIONS.
9. DEPRESS THE START AND STOP KEYS SEVERAL TIMES TO DEMONSTRATE THE MAKING READY AND NOT READY OF THE THE CARD READER.
10. WITH THE READY LIGHT OFF, DEPRESS THE FEED KEY AND FEED THE CARD THRU TO THE STACKER.
11. PLACE THE CARD BACK IN THE HOPPER.

12. DEPRESS THE START KEY AND HOLD IT DOWN UNTIL THE CARD READER STOPS. NOTICE THAT THE READY LIGHT IS ON. THIS IS THE QUICK WAY TO ACCOMPLISH STEPS 3, 4, AND 5 ABOVE.
13. PLACE THE MANUAL-AUTOMATIC SWITCH ON THE CPU CONSOLE IN THE AUTOMATIC POSITION.
14. DEPRESS THE LOAD CARDS KEY ON THE CPU CONSOLE. THE CARD READER WILL TAKE ONE MACHINE CYCLE. THE CARD WILL BE PASSED BENEATH THE READING BRUSHES. THREE WORDS, THE CONTENTS OF 9 ROW LEFT, 9 ROW RIGHT, AND 8 ROW LEFT WILL BE TRANSMITTED TO CORE STORAGE. THEY WILL BE STORED IN THAT ORDER AT CORE STORAGE LOCATIONS 00000, 00001, AND 00002.
15. THIS IS AN UNCHANGEABLE RESULT OF HAVING DEPRESSED THE LOAD CARDS KEY. NO PROGRAM WAS REQUIRED TO TRANSMIT THESE THREE WORDS, BUT NO MODIFICATION CAN BE MADE AS TO THE NUMBER OF WORDS TRANSMITTED OR WHERE THEY ARE STORED.
16. PLACE THE MANUAL-AUTOMATIC SWITCH IN MANUAL. PLACE 00000 IN THE ADDRESS PORTION OF THE OP. PANEL KEYS. DEPRESS THE DISPLAY STORAGE BUTTON. THE CONTENTS OF STORAGE LOCATION 00000 WILL BE PLACED IN THE STORAGE REGISTER FOR YOU TO EXAMINE.
17. DISPLAY LOCATIONS 00001 THRU 00005 IN THE SAME MANNER. COMPARE THE CONTENTS OF EACH LOCATION TO THE CARD CONTENT SHOWN FOR CARD NUMBER 1 IN APPENDIX 1 OF THIS PROJECT.
18. DEPRESS THE CLEAR KEY AND DISPLAY THESE LOCATIONS AGAIN. THE BITS ARE STILL PRESENT BECAUSE CLEAR IS NOT EFFECTIVE IN MANUAL. THIS SERVES AS PROTECTION AGAINST ACCIDENTAL CLEARING DURING A MANUAL OPERATION.
19. PLACE THE MACHINE IN AUTOMATIC. PRESS CLEAR. RETURN TO MANUAL.
20. DISPLAY LOCATIONS 00000 THRU 00005 AGAIN TO VERIFY THAT THEY NOW CONTAIN ZEROS.
21. TRY EACH OF THE FOLLOWING AND TAKE CAREFUL NOTE OF THE RESULTS.
 - A) WITH THE CARD READER READY AND THE MACHINE IN MANUAL STATUS DEPRESS THE LOAD CARDS BUTTON.
 - B) WITH THE MACHINE IN AUTOMATIC AND THE CARD READER NOT READY, DEPRESS THE LOAD CARDS BUTTON.
 - C) WITH INFORMATION FROM THE CARD ALREADY IN STORAGE, DISPLAY THE STORAGE LOCATIONS CONCERNED, WITH THE MACHINE IN AUTOMATIC STATUS.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

PART 2 - SINGLE STEP, MULTIPLE STEP, RESET,
START, ENTER INSTRUCTION, ENTER MQ

1. PLACE CARD 2 IN THE CARD READER AND, USING THE LOAD CARDS KEY, READ IT INTO STORAGE. THE FIRST THREE WORDS OF THE CARD, WHICH ARE TRANSMITTED FREE, ARE SUCH THAT THE COMPUTER WILL READ THE BALANCE OF THE CARD INTO STORAGE ALSO. CPU WILL WAIT AT LOCATION 00001 UNTIL THIS IS ACCOMPLISHED. THEN IT WILL PROCEED WITH THE NEXT INSTRUCTION IN SEQUENCE WHICH WILL COME FROM LOCATION 00002.
2. THE COMPUTER IS IN AN ENDLESS LOOP AS YOU CAN SEE FROM THE CARD LISTING IN APPENDIX 2. PLACE THE COMPUTER IN MANUAL AND IT WILL STOP AFTER COMPLETING THE INSTRUCTION IN PROCESS.
3. THE INSTRUCTION REGISTER HOLDS THE OPERATION CODE OF THE INSTRUCTION JUST COMPLETED.
4. THE STORAGE REGISTER CONTAINS THE NEXT INSTRUCTION TO BE PERFORMED.
5. THE INSTRUCTION COUNTER (PROGRAM COUNTER) CONTAINS THE STORAGE LOCATION OF THE NEXT INSTRUCTION TO BE PERFORMED.
6. STUDY THE PROGRAM UNTIL YOU FEEL YOU UNDERSTAND WHAT IT SHOULD DO. DETERMINE WHICH PART OF THE PROGRAM THE MACHINE WAS IN WHEN YOU CAUSED IT TO STOP BY PLACING IT IN MANUAL.
7. WITH THE MACHINE IN MANUAL, DEPRESS THE SINGLE STEP BUTTON. OBSERVE THE OPERATION OF THE INSTRUCTION REGISTER, THE INSTRUCTION COUNTER, AND THE VARIOUS DATA REGISTERS.
8. PROCEED IN THIS MANNER, OBSERVING THE OPERATION OF THE PROGRAM. FOLLOW THE LISTING IN THE APPENDIX AND RELATE IT TO THE DISPLAYS YOU OBSERVE. IF THERE IS ANY DISPLAY YOU DO NOT UNDERSTAND, CONSULT YOUR LAB INSTRUCTOR AT ONCE.
9. NOW DEPRESS THE MULTIPLE STEP KEY. AS LONG AS YOU HOLD THE KEY THE MACHINE WILL CONTINUE TO PROCESS THE PROGRAM, BUT AT A RATE GREATLY BELOW ITS NORMAL AUTOMATIC RATE.
10. THIS FEATURE MAY BE USED TO SPEED UP MANUAL OPERATIONS WHEN YOU DO NOT WISH TO OBSERVE EACH OPERATION.
11. OPEN THE COVER TO THE C.E. PANEL AT THE RIGHT OF THE CONSOLE. LOCATE THE HIGH SPEED-LOW SPEED MULTIPLE STEP TOGGLE SWITCH. CHANGE THE POSITION OF THIS SWITCH AND NOTE THE EFFECT ON MACHINE OPERATION.

12. ATTEMPT MULTIPLE STEP AND SINGLE STEP OPERATION WITH THE MACHINE IN AUTOMATIC STATUS. NOTE THE RESULT.
13. WITH THE MACHINE IN AUTOMATIC STATUS, DEPRESS THE START KEY ON THE CONSOLE. THE MACHINE IS ONCE AGAIN OPERATING THE PROGRAM AT HIGH SPEED. NOTE THAT THE WHITE READY LIGHT IS OFF AND THE YELLOW AUTOMATIC LIGHT IS ON.
14. WITH THE MACHINE STILL IN AUTOMATIC, DEPRESS THE RESET KEY. ALL REGISTERS HAVE BEEN RESET AND THE PROGRAM HAS BEEN INTERRUPTED. HOWEVER, THE PROGRAM STILL EXISTS IN CORE STORAGE.
15. IN ORDER TO RESTART THE PROGRAM IT WILL BE NECESSARY TO MANUALLY TRANSFER TO THE START OF THE LOOP.
16. PLACE THE OP CODE FOR THE INSTRUCTION TRA IN POSITIONS S-11 OF THE OP PANEL KEYS.
17. PLACE THE LOCATION OF THE CLA INSTRUCTION IN POSITIONS 21-35 OF THE OP PANEL KEYS.
18. DEPRESS THE ENTER INSTRUCTION BUTTON. THIS BUTTON CAUSES THE MACHINE TO PERFORM THE INSTRUCTION THAT IS IN THE OP PANEL KEYS. OBSERVE WHAT HAPPENS TO THE INSTRUCTION COUNTER WHEN YOU DEPRESS THE ENTER INSTRUCTION BUTTON.
19. NOTHING HAPPENED BECAUSE THE MACHINE IS IN AUTOMATIC AND IS THEREFORE IGNORING THE ENTER INSTRUCTION BUTTON. PLACE THE MACHINE IN MANUAL AND DO STEP 18 AGAIN.
20. THE PROGRAM COUNTER SHOULD NOW READ 00002. THIS IS THE STORAGE LOCATION OF THE NEXT INSTRUCTION THAT WOULD BE PERFORMED IF THE MACHINE WAS PUT INTO OPERATION.
21. PLACE THE MACHINE IN OPERATION. IT SHOULD RUN AS BEFORE.
22. STOP THE PROGRAM.
23. DISPLAY LOCATION 00004. YOU WILL CHANGE THE NUMBER OF PASSES THRU THE LOOP BY MODIFYING THIS INSTRUCTION.
24. PLACE THE OP CODE OF THE INSTRUCTION (AXT) IN THE OP PANEL KEYS. DO THE SAME WITH THE TAG. THIS INFORMATION SHOULD BE THE SAME AS IT WAS ORIGINALLY IN CORE.
25. MULTIPLY THE PRESENT ADDRESS PORTION OF THE AXT INSTRUCTION BY TWO AND PLACE THIS NEW NUMBER IN THE ADDRESS PORTION (21-35) OF THE OP PANEL KEYS.
26. DEPRESS THE ENTER MQ BUTTON. THIS WILL MOVE THE CONTENTS OF THE KEYS TO THE MQ. SEE IF THIS WAS DONE PROPERLY.
27. THIS NEW INSTRUCTION MAY NOW BE STORED IN PLACE OF THE OLD ONE BY PLACING A STQ AND THE PROPER ADDRESS IN THE KEYS AND DEPRESSING THE ENTER INSTRUCTION KEY.

28. DISPLAY TO SEE THAT THE PROPER INFORMATION WAS STORED.
29. RUN THE PROGRAM IN AUTOMATIC, SINGLE STEP, AND HIGH AND LOW SPEED MULTIPLE STEP. NOTICE THE DIFFERENCES WHICH RESULT FROM YOUR MODIFICATION TO THE PROGRAM.
30. MAKE ANY OTHER MODIFICATIONS TO THE PROGRAM THAT YOU MAY WISH.
31. CLEAR STORAGE.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

PART III - DISPLAY EFFECTIVE ADDRESS

1. CLEAR STORAGE.
2. MANUALLY STORE THE FOLLOWING PROGRAM. WHEN YOU HAVE IT STORED, CHECK TO BE SURE IT IS IN STORAGE CORRECTLY. ALL NUMBERS SHOWN ARE IN OCTAL.

00000	ENK	
00001	XCA	
00002	AXT	100,1
00003	STO	500,1
00004	TIX	*-1,1,1
00005	HPR	
00006	TRA	00000
3. PLACE SOME BINARY VALUE IN THE OP PANEL KEYS. SINGLE STEP THE PROGRAM UNTIL THE INSTRUCTION AT LOCATION 00003 HAS BEEN PERFORMED SEVERAL TIMES.
4. DISPLAY THE APPROPRIATE SEQUENTIAL STORAGE LOCATIONS AND FIND THE INFORMATION WHICH HAS BEEN STORED.
5. PLACE THE HIGH-LOW SPEED MULTIPLE STEP TOGGLE SWITCH IN THE LOW SPEED POSITION.
6. MOMENTARILY DEPRESS THE MULTIPLE STEP BUTTON SO THAT THE MACHINE HAS GONE THRU SEVERAL PASSES OF THE LOOP.
7. WHEN YOU RELEASE THE KEY, THE PROGRAM MAY STOP WITH 00003 IN THE INSTRUCTION COUNTER AND STO 00500,1 IN THE STORAGE REGISTER. IF IT DOES NOT STOP IN THIS MANNER YOU MAY SINGLE STEP THE PROGRAM UNTIL IT DOES.
8. THE PROBLEM AT THIS TIME IS TO DETERMINE WHICH STORAGE LOCATION WILL BE ACTED UPON BY THE STO INSTRUCTION WHICH IS ABOUT TO BE PERFORMED. THIS COULD BE DETERMINED BY USING OCTAL OR BINARY ARITHMETIC TO SUBTRACT THE PRESENT CONTENTS OF THE XR FROM THE ADDRESS IN THE INSTRUCTION. THIS COULD BE TIME CONSUMING.
9. THE MACHINE WILL HAVE TO PERFORM THIS CALCULATION IN ORDER TO EXECUTE THE STO INSTRUCTION. WE CAN CAUSE IT TO DO THE SAME THING FOR US, BEFORE IT EXECUTES THE STO, BY DEPRESSING THE DISPLAY EFFECTIVE ADDRESS BUTTON. THE COMPUTER WILL OPERATE ON THE INSTRUCTION THAT IS IN THE STORAGE REGISTER WHEN THE KEY IS DEPRESSSED. DEPRESS THE DISPLAY EFFECTIVE ADDRESS KEY NOW. THE EFFECTIVE ADDRESS (ADDRESS PORTION OF THE INSTRUCTION MINUS THE CURRENT VALUE OF THE SPECIFIED XR) IS NOW DISPLAYED IN THE STORAGE REGISTER. IT HAS BEEN CALCULATED FOR YOU BY THE COMPUTER. RECORD THIS ADDRESS.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

PART IV - CONTINUOUS ENTER INSTRUCTION

1. CONTINUOUS ENTER INSTRUCTION IS A MEANS OF REPEATEDLY EXECUTING THE SAME INSTRUCTION UNDER MANUAL CONTROL (NOT STORED IN CORE) AT NORMAL COMPUTER SPEEDS.
2. RESET THE MACHINE TO CLEAR ALL REGISTERS.
3. STORE A BIT IN POSITION 35 OF SOME STORAGE LOCATION.
4. PLACE THE INSTRUCTION ADD AND THE ADDRESS OF THE LOCATION HAVING THE BIT IN THE OP PANEL KEYS.
5. PRESS THE ENTER INSTRUCTION BUTTON SEVERAL TIMES. WATCH THE TOTAL ACCUMULATE IN THE ACCUMULATOR AS THE SAME INSTRUCTION IS PERFORMED AGAIN WITH EACH DEPRESSION.
6. ON THE C.E. PANEL AT THE RIGHT OF THE CONSOLE, FIND THE CONTINUOUS ENTER INSTRUCTION TOGGLE SWITCH. PUT IT IN THE ON (UP) POSITION.
7. NOTICE THAT THE READY LIGHT ON THE CONSOLE WENT OFF. THIS IS AN INDICATION TO THE OPERATOR THAT SOME SUCH SWITCH IS ON AND THE MACHINE IS, THEREFORE, NOT READY FOR NORMAL OPERATION.
8. PLACE THE MACHINE IN AUTOMATIC. DEPRESS THE START KEY. THE INSTRUCTION IN THE OP PANEL KEYS WILL NOW BE PERFORMED REPETITIVELY UNTIL YOU STOP IT. NOTE THE OPERATION OF THE ACCUMULATOR.
9. AFTER THIS HAS RUN FOR A WHILE, CHANGE THE OP CODE FROM ADD TO SUB.
10. YOU MAY STOP THIS OPERATION BY--
 - A. PLACING THE MACHINE IN MANUAL.
 - B. DEPRESSING THE RESET BUTTON.
 - C. TURNING OFF THE CONTINUOUS ENTER INSTRUCTION SWITCH.
 - D. CHANGING THE OP CODE TO A HALT.
11. CLEAR THE MACHINE.
12. ANSWER THE QUESTIONS IN THE FOLLOWING SECTION.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

PART V - QUESTIONS

1. CARDS ARE PLACED IN THE CARD READER (FACE UP) (FACE DOWN) AND (9 EDGE FIRST) (12 EDGE FIRST).
2. IF THE START KEY ON THE READER IS NOT HELD DOWN, HOW MANY DEPRESSIONS OF THE KEY ARE NORMALLY REQUIRED BEFORE THE READY LIGHT COMES ON.

3. THE FEED KEY IS EFFECTIVE ONLY WHEN THE CARD READER IS (READY) (NOT READY).
4. THE LOAD CARDS KEY IS EFFECTIVE ONLY WHEN THE MACHINE IS IN THE (MANUAL) (AUTOMATIC) STATUS.
5. THE DISPLAY KEY IS EFFECTIVE ONLY WHEN THE MACHINE IS IN (MANUAL) (AUTOMATIC) STATUS.
6. THE CLEAR KEY IS EFFECTIVE ONLY WHEN THE MACHINE IS IN (MANUAL) (AUTOMATIC) STATUS.
7. TELL WHAT IS IN THE INSTRUCTION COUNTER, THE INSTRUCTION REGISTER, AND THE STORAGE REGISTER WHEN THE MACHINE IS MANUALLY FORCED TO STOP BY BEING PUT INTO MANUAL STATUS.

INSTRUCTION COUNTER CONTAINS _____

INSTRUCTION REGISTER CONTAINS _____

STORAGE REGISTER CONTAINS _____

8. DESCRIBE THE EFFECT OF THE ENTER MQ BUTTON.

9. GIVE AN EXAMPLE OF A SITUATION WHERE YOU MIGHT USE THE CONTINUOUS ENTER INSTRUCTION FEATURE.

10. TELL WHAT HAPPENS WHEN THE DISPLAY EFFECTIVE ADDRESS KEY IS DE-PRESSED.

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

APPENDIX 1

CARD 1

9L	00000	OCT	77777777777
9R	00001	OCT	77777777776
8L	00002	OCT	77777777775
8R	00003	OCT	77777777774
7L	00004	OCT	77777777773
7R	00005	OCT	77777777772

LABORATORY PROJECT 3
7151 CONSOLE ORIENTATION

APPENDIX 2

CARD 2

00000	IOCD	3,,25
00001	TCOA	1
00002	CLA	13
00003	AXT	42,2
00004	AXT	10,1
00005	ALS	1
00006	ARS	1
00007	TIX	5,1,1
00010	ALS	1
00011	TIX	4,2,1
00012	TRA	2
00013	OCT	1

LABORATORY PROJECT 4A
OSCILLOSCOPE APPLICATIONS

- I. PURPOSE
DEMONSTRATE THE OPERATION OF A - 0 CIRCUIT

- II. PROCEDURE
EXAMINE THE THREE INPUTS TO AN OR CIRCUIT, NOTE THEIR TIMING,
AND COMPARE THEM TO THE OUTPUT.

- III. MACHINE REQUIREMENTS
C.P.U.

- IV. PREPARATION
REVIEW THE OPERATION AND CHARACTERISTICS OF THE DEZE CARD.
REVIEW BASIC OPERATION OF TEKTRONIX OSCILLOSCOPE.

- V. REFERENCES
 - A. PAGE 79 OF C.E. MANUAL OF INSTRUCTION SMS COMPONENT
CIRCUITS
 - B. PAGES 5 - 16 OF C.E. MANUAL OF INSTRUCTION TEKTRONIX
OSCILLOSCOPES

VI. SCOPE SETUP

A. TIME BASE-A CONTROLS
TRIGGERING MODE - AC LOW FREQUENCY REJECT
TRIGGER SLOPE - PLUS EXTERNAL
TIME PER CENTIMETER - .2 MICROSECONDS

B. PREAMPLIFIER CONTROLS - CHANNELS A AND B
MODE - ALTERNATE
VOLTS PER CENTIMETER - .2

C. SIGNAL INPUTS

1. TRIGGER INPUT A LINE NAME +N AO D1
 SYSTEM PAGE 08.00.40.1
 LOCATION 03A4E17A

2. CHANNEL A LINE NAME +N AO D1
 SYSTEM PAGE 08.00.40.1
 LOCATION 03A4F17A

3. CHANNEL SCOPE THE FOLOWING POINTS IN
 SEQUENCE. ON THE ATTACHED CHART
 RECORD THE TIME RELATIONSHIPS OF
 THE PULSES OBSERVED. LABEL THE
 VOLTAGE REFERENCES (SHOW A TRACE
 ALSO)

A. LINE NAME - A1 D1
 SYSTEM PAGE 08.00.48.1
 LOCATION 03A4F21D

B. LINE NAME -N A5 D1
 SYSTEM PAGE 08.00.48.1
 LOCATION 03A4F21E

C. LINE NAME -N A9 D1
 SYSTEM PAGE 08.00.48.1
 LOCATION 03A4F21F

D. LINE NAME +P A1,A5,A9 D1
 SYSTEM PAGE 08.00.48.1
 LOCATION 03A4F21B

Channel A

Channel B
Signal a

Signal b

Signal c

Signal d

LABORATORY PROJECT 4B
OSCILLOSCOPE APPLICATIONS

I. PURPOSE

DEMONSTRATE THE OPERATION OF A +A CIRCUIT.

II. PROCEDURE

EXAMINE THE TWO INPUTS TO AN AND CIRCUIT, NOTE THEIR TIMING,
AND COMPARE THEM TO THE OUTPUT.

III. MACHINE REQUIREMENTS

C.P.U.

IV. PREPARATION

REVIEW THE OPERATION AND CHARACTERISTICS OF THE DAZW CARD.

V. REFERENCES

PAGE 63 OF C.E. MANUAL OF INSTRUCTION SMS COMPONENT CIRCUITS

VI. SCOPE SETUP

- A. TIME BASE-A CONTROLS
TRIGGERING MODE - AC LOW FREQUENCY REJECT
TRIGER SLOPE - PLUS EXTERNAL
TIME PER CENTIMETER - .5 MICROSECONDS
- B. PREAMPLIFIER CONTROLS - CHANNELS A AND B
MODE - ALTERNATE
VOLTS PER CENTIMETER - .2
- C. SIGNAL INPUTS - PART A

- 1. TRIGGER INPUT A LINE NAME +N MASTER E TIME
 SYSTEM PAGE 08.00.19.1
 LOCATION 02B1B17E
- 2. CHANNEL A LINE NAME +N GO TO I TIME
 SYSTEM PAGE 08.00.18.1
 LOCATION 02B1E15E
- 3) CHANNEL B LINE NAME +N A11 D1
 SYSTEM PAGE 08.00.18.1
 LOCATION 02B1E15F

PLACE THE INSTRUCTION CLA (+0500) IN THE OP PANEL KEYS, TURN ON THE CONTINUOUS ENTER INSTRUCTION SWITCH, PRESS THE START KEY. ON THE ATTACHED CHART, RECORD THE TIME RELATIONSHIPS OF THE PULSES OBSERVED.

- D. SIGNAL INPUTS - PART B
- 1. TRIGGER INPUT A SAME AS ABOVE
- 2. CHANNEL A SAME AS ABOVE
- 3. CHANNEL B SAME AS ABOVE

TURN THE PREAMPLIFIER MODE SWITCH TO THE ADDED ALGEBRAICALLY POSITION AND RECORD THIS PULSE. ADJUSTMENT OF THE VERTICAL POSITION CONTROLS MAY BE NECESSARY. THE HIGHEST PORTION OF THE WAVEFORM OBSERVED REPRESENTS THE TIME OF CO-INCIDENCE OF THE TWO INPUT PULSES.

E. SIGNAL INPUTS - PART C

1. TRIGGER INPUT A SAME AS ABOVE

2. CHANNEL A LINE NAME +P TURN ON MASTER I
 TIME
 SYSTEM PAGE 08.00.18.1
 LOCATION 02B1E15G

3. CHANNEL B NOT USED

PLACE THE PREAMPLIFIER MODE SWITCH IN THE A ONLY POSITION. RECORD THIS PULSE WHICH IS THE IN-PHASE OUTPUT OF THE AND CIRCUIT.

Part A

Channel A

Channel B

Part B

Algebraically
Added

Part C

Channel A

LABORATORY PROJECT 4C
OSCILLOSCOPE APPLICATIONS

I. PURPOSE

DEMONSTRATE THE OPERATION OF A TRIGGER CIRCUIT USING A
-TO AND A -TA

II. PROCEDURE

EXAMINE THE OUTPUTS OF THE -TO AND -TA IN RESPECT TO EACH
OTHER. NOTE THE TIMING DIFFERENCES

III. MACHINE REQUIREMENTS

C.P.U.

IV. PREPARATION

REVIEW THE OPERATION AND CHARACTERISTICS OF THE DAZW AND THE
DFZE CARDS

V. REFERENCES

PAGES 63 AND 86 OF C.E. MANUAL OF INSTRUCTION SMS COMPONENT
CIRCUITS

VI. SCOPE SETUP

- A. TIME BASE-A CONTROLS
TRIGGERING MODE - AC LOW FREQUENCY REJECT
TRIGGER SLOPE - MINUS EXTERNAL
TIME PER CENTIMETER - .1 MICROSECONDS
- B. PREAMPLIFIER CONTROLS - CHANNELS A AND B
MODE - ALTERNATE
VOLTS PER CENTIMETER - .2
- C. SIGNAL INPUTS - PART A
 - 1. TRIGGER INPUT A LINE NAME -N A1 D1
 SYSTEM PAGE 08.00.40.1
 LOCATIONS 03A4E19F
 - 2. CHANNEL A LINE NAME -P A1 D2
 SYSTEM PAGE 08.00.40.1
 LOCATIONS 03A4E19G
 - 3. CHANNEL B LINE NAME -N A1 D1
 SYSTEM PAGE 08.00.40.1
 LOCATIONS 03A4D19G

ON THE ATTACHED CHART, RECORD THE TIME RELATIONSHIPS OF THE PULSES OBSERVED. THESE ARE THE IN-PHASE OUTPUTS OF A MINUS TRIGGER. NOTE THE DIFFERENCE IN DURATION OF THE TWO PULSES.

- D. SIGNAL INPUTS - PART B
 - 1. TRIGGER INPUT A LINE NAME -N A1 D1
 SYSTEM PAGE 08.00.40.1
 LOCATIONS 03A4E19E
 - 2. CHANNEL A LINE NAME +P A1 D2
 SYSTEM PAGE 08.00.40.1
 LOCATIONS 03A4E19H
 - 3. CHANNEL B LINE NAME +N A1 D2
 SYSTEM PAGE 08.00.40.1
 LOCATION 03A4D19B

ON THE ATTACHED CHART, RECORD THE TIME RELATIONSHIPS OF THE PULSES OBSERVED. THESE ARE THE OUT OF PHASE OUTPUTS OF A MINUS TRIGGER. NOTE THE DIFFERENCE IN DURATION OF THE TWO PULSES. EXPLAIN WHY THESE PULSES ARE NOT OF THE SAME DURATION.

Part A

Channel A

Channel B

Part B

Channel A

Channel B

LABORATORY PROJECT 4D
OSCILLOSCOPE APPLICATIONS

I. PURPOSE

DEMONSTRATE THE OPERATION OF A BINARY TRIGGER (TB)

II. PROCEDURE

OBSERVE THE INPUT LINE, IN PHASE AND OUT OF PHASE OUTPUTS,
AND DELAYED OUTPUTS OF A -TB IN POSITION 17 OF THE WORD CTR.

III. MACHINE REQUIREMENTS

DATA CHANNEL

IV. PREPARATION

REVIEW THE OPERATION AND THE CHARACTERISTICS OF THE FF-- AND
FJ-- CARDS. REVIEW DELAYED SWEEP OPERATION OF THE
TEKTRONIX OSCILLOSCOPE.

V. REFERENCES

- A. PAGES 104 - 105 OF C.E. MANUAL OF INSTRUCTION SMS
COMPONENT CIRCUITS
- B. PAGES 18 - 21 AND 30 -35 OF C.E. MANUAL OF INSTRUCTION
TEKTRONIX OSCILLOSCOPES

VI. SCOPE SETUP

A. TIME BASE B CONTROLS

1. TRIGGERING MODE - AC
2. TRIGGER SLOPE - PLUS EXTERNAL
3. TIME PER CENTIMETER - 2.0 MICROSECONDS

B. TIME BASE A CONTROLS

1. STABILITY AND TRIGGERING LEVEL CONTROLS FULLY CLOCKWISE
2. TIME PER CENTIMETER - 1 MICROSECOND OR LESS

C. SIGNAL INPUTS

1. TRIGGER INPUT B LINE NAME +N STEP WC
 SYSTEM PAGE 60.30.01.1
 LOCATIONS 06C2K06D
2. CHANNEL A LINE NAME +N STEP WC
 SYSTEM PAGE 60.30.01.1
 LOCATIONS 06C2K06D
3. CHANNEL B SCOPE THE FOLLOWING POINTS IN
 SEQUENCE. ANSWER THE QUESTIONS
 FOLLOWING EACH POINT.

- (A) LINE NAME -N STEP WC 17
 SYSTEM PAGE 60.30.01.1
 LOCATION 06C2J07H

- (1) THE LENGTH OF THIS PULSE, MEASURED AT THE REFERENCE LINE, IS -----.
- (2) THE PULSE SWINGS FROM ----- VOLTS TO -----VOLTS.
- (3) THE LENGTH OF THE PULSE ON CHANNEL A IS -----.
- (4) EXPLAIN THE REASON FOR THE UNUSUAL VOLTAGE LEVELS OF THE B PULSE AND THE REASON FOR THE DIFFERENCE IN LENGTH BETWEEN THE A AND B PULSES.

(B) LINE NAME -P WC 17
SYSTEM PAGE 60.30.01.1
LOCATIONS 06C2J07B

(1) HOW MUCH TIME HAS ELAPSED BETWEEN THE TIME A PULSE RISES ACROSS THE REFERENCE LINE AND WHEN THE B PULSE FALLS ACROSS THE REFERENCE LINE

(2) WAS THE TRIGGER TURNED ON OR OFF

(C) LINE NAME -N WC 17
SYSTEM PAGE 60.30.01.1
LOCATIONS 06C2H07H

(1) WHAT IS THE TOTAL DELAY BETWEEN THE RISE OF THE A PULSE AND THE FALL OF THE B PULSE

(2) HOW MUCH TIME IS THERE BETWEEN THE FALL OF A PULSE THAT TURNED THE TRIGGER ON AND THE FALL OF THE B PULSE THAT REFLECTS THIS ON STATUS

(3) EXPLAIN WHY THIS DELAY IS USED

LABORATORY PROJECT 4E
OSCILLOSCOPE APPLICATIONS

I. PURPOSE

DEMONSTRATE THE DIFFERENCE IN CHARACTERISTICS BETWEEN ALLOY CIRCUITRY AND DIFFUSED CIRCUITRY.

II. PROCEDURE

COMPARE THE OUTPUTS OF AMZZ CONVERT BLOCK WITH THE OUTPUTS OBTAINED WHEN A DAZZ CONVERT BLOCK (371282) IS PLACED IN THE SAME LOCATION.

III. MACHINE REQUIREMENTS

DATA CHANNEL AND ONE TAPE UNIT

IV. PREPARATION

REVIEW THE OPERATION AND CHARACTERISTICS OF THE AMZZ AND DAZZ CARDS. REVIEW DELAYED SWEEP.

V. REFERENCES

- A. PAGES 32 AND 64 OF CE MANUAL OF INSTRUCTION SMS COMPONENT CIRCUITS
- B. PAGES 20-21 OF CE MANUAL OF INSTRUCTION FOR TEKTRONIX OSCILLOSCOPES (SWEEP DELAYED APPLICATIONS)

VI. SCOPE SETUP

A. TIME BASE B CONTROLS

1. TRIGGERING MODE - AC
2. TRIGGERING SLOPE - MINUS EXTERNAL
3. TIME PER CENTIMETER - 2.0 MICROSECONDS

B. TIME BASE A CONTROLS

1. STABILITY AND TRIGGERING LEVEL CONTROLS FULLY CLOCKWISE
2. TIME PER CENTIMETER - 100 NANoseconds

C. SIGNAL INPUTS - PART A

1. TRIGGER INPUT B LINE NAME -N WC 2 TR
 SYSTEM PAGE 61.20.20.1
 LOCATION 06B2G09D
2. CHANNEL A LINE NAME +N WC 1 TR
 SYSTEM PAGE 61.20.20.1
 LOCATION 06B2G09D
3. LINE NAME -P WC SAMPLE
 SYSTEM PAGE 61.20.20.1
 LOCATION 06B2G09A

- (A) WHAT IS THE TIME DIFFERENCE BETWEEN THE INPUT TO THE BLOCK (PIN D) AND THE OUT OF PHASE OUTPUT (PIN A).
- (B) REMOVE THE AND CIRCUITS ON PAGE 61.20.30.1 THAT ARE DRIVEN BY THIS BLOCK. WHAT IS THE DELAY OF THE BLOCK.
- (C) REMOVE THE ALLOY CONVERT BLOCK (AMZZ) FROM THE MACHINE. PLACE THE DIFFUSED CARD (DAZZ) IN ITS PLACE. WHAT IS THE DELAY OF THE DIFFUSED CARD
- (D) REPLACE THE AND CIRCUITS DRIVEN BY THE CONVERT BLOCK. MEASURE THE DELAY AGAIN.
- (E) RESTORE THE MACHINE TO ITS ORIGINAL CONDITION BY REMOVING THE DIFFUSED CARD AND INSERTING THE ALLOY CARD.

D. SIGNAL INPUTS - PART B

- | | | | |
|----|-----------------|-------------|--------------|
| 1) | TRIGGER INPUT B | LINE NAME | -N WC 2 TR |
| | | SYSTEM PAGE | 61.20.20.1 |
| | | LOCATION | 06B2F10F |
| 2. | CHANNEL A | LINE NAME | -P WC SAMPLE |
| | | SYSTEM PAGE | 61.20.20.1 |
| | | LOCATION | 06B2G09A |
| 3. | CHANNEL B | LINE NAME | +P WC SAMPLE |
| | | SYSTEM PAGE | 61.20.20.1 |
| | | LOCATION | 06B2G09B |

- (A) WITH THE ALLOY CARD AND ALL THE AND CIRCUITS IN THE MACHINE, DETERMINE THE AMOUNT OF TIME BETWEEN THE RISE OF THE IN PHASE OUTPUT AND THE FALL OF THE OUT OF PHASE OUTPUT.
- (B) MAKE THE SAME MEASUREMENT USING THE DIFFUSED CARD.
- (C) RESTORE THE MACHINE TO ITS NORMAL CONDITION.

LABORATORY PROJECT 4F
OSCILLOSCOPE APPLICATIONS

I. PURPOSE

DEMONSTRATE THE INPUT SIGNALS OF THE FOUR TYPES OF DRIVER TERMINATOR (DT)

II. PROCEDURE

THE INPUTS OF FOUR DTS ARE SCOPED WHILE THEY ARE IN OPERATION. THE FOUR ARE, IN ORDER, N-P, P-N, P-P, N-N

III. MACHINE REQUIREMENTS

DATA CHANNEL

IV. PREPARATION

REVIEW THE CHARACTERISTICS AND OPERATION OF THE DL--, DK--, RL--, AND DJ-- CARDS.

V. REFERENCES

PAGES 89, 90, 91, AND 92 OF CE MANUAL OF INSTRUCTION SMS COMPONENT CIRCUITS

VI. SCOPE SETUP

A. TIME BASE B CONTROLS

1. TIME BASE B IS NOT USED

B. TIME BASE A CONTROLS

1. USE INTERNAL SYNC
2. USE TIME PER CENTIMETER APPROPRIATE TO PULSE OBSERVED

C. SIGNAL INPUTS

1. CHANNEL A

USING TAPE CYCLE WRITE A SMALL NUMBER OF RECORDS ON TAPE. WRITE AN END OF FILE AND REWIND THE TAPE. PLACE THE TAPE IN CONTINUOUS READ CYCLE OPERATION AND SCOPE THE FOLLOWING POINTS. IN EACH CASE USE THE ATTACHED FORM TO RECORD THE PULSE YOU OBSERVE AND THE INDICATED SCOPE SETTINGS.

- | | | |
|-----|-------------|-----------------------|
| (A) | LINE NAME | +N STEP GP CNTR |
| | SYSTEM PAGE | 60.25.13.1 |
| | LOCATION | 06C4J14C |
| (B) | LINE NAME | +N RCP CNTL GATE |
| | SYSTEM PAGE | 60.36.01.1 |
| | LOCATION | 06D3D09D |
| (C) | LINE NAME | +P CHAN RDS |
| | SYSTEM PAGE | 60.50.02.1 |
| | LOCATION | 06C3G13D |
| (D) | LINE NAME | -P SET DATA DISC GATE |
| | SYSTEM PAGE | 60.36.02.1 |

a.	DT Type _____										
	Volts/cm _____										
	Voltage _____										
	Time/cm _____										
b.	DT Type _____										
	Volts/cm _____										
	Voltage _____										
	Time/cm _____										
c.	DT Type _____										
	Volts/cm _____										
	Voltage _____										
	Time/cm _____										
d.	DT Type _____										
	Volts/cm _____										
	Voltage _____										
	Time/cm _____										

LABORATORY PROJECT 5

POWER SUPPLY

THE VOLTAGES PRESENT IN THE 7090 POWER SYSTEM ARE EXTREMELY DANGEROUS. THE LAB INSTRUCTOR WILL DROP POWER TO THE SYSTEM FROM THE WALL BOX AND ALLOW TIME FOR ALL SUPPLY CAPACITORS TO BLEED OFF BEFORE STUDENTS ARE TO BEGIN THIS PROJECT. (STUDENTS SHOULD TAKE NOTE OF HOW POWER IS DROPPED AND BROUGHT UP)

I. 7608 Power Converter

Remove the covers from the 7608 and locate the following:
(For reference use the red 7608 7618 CE Reference Manual)

A. Blower Assembly

1. Are there any lubrication fittings on the blower motor?
2. Locate the thermal switch in the M-G.
3. Are there any air filters?
4. Is the blower CB located in the MG set?

B. Drive Motor and Generator

1. Locate the grease fittings on the drive motor and generator.
2. Are there any brushes in the drive motor? In the generator? How do you change them?
3. Locate the cable that connects to the generator field.
4. Where are the 100 and 150 Amp. connectors located?
Big--aren't they?

C. M-G CB-1

1. Is the load connected to the top or bottom of CB-1?
2. Do you know how to reset CB-1?
3. Find the wires connected to CB1 "under voltage release coil."

II. 7618 Power Control Unit

A. Front control panel

1. Locate the Voltmeter. What is it for? What is the range switch for?

2. Locate:
 - a. Pwr on switch
 - b. Normal off switch
 - c. Emergency off switch
 - d. Pwr on reset switch
 - e. D.C. off switch

Do you know what these switches do?

3. Locate the 13 blower CB's. Is the memory blower CB different? Why?
4. How many convenience outlet breakers are there?
5. Locate:
 - a. CB-28 and 29, the 208V. 400 cycle line CB's
 - b. CB-30, the 60 cycle input CB
 - c. CB-31, M-G output CB
 - d. CB-32, M-G blower CB
 - e. CB-33, Mem. Temp. unit CB
 - f. CB-34, power sequence powerstat CB
 - g. CB-35, M/C powerstat CB

B. Power Cable Distribution (Rear Pane)

1. Notice how the cables are labeled for each frame to which they connect.
2. Notice that there are two connectors for each frame (except the console which has four)
3. The center 2 rows of connectors carry 400 cycle power. The outer two rows carry 60 cycle and DC power to the frames.
4. Note the pin number labeling on the connectors.
5. Note that even though these connectors carry heavy current, they are physically of light construction. Care should be taken in connection and removal of cables.

C. M/C and Power Sequencing Variacs

1. Locate and identify the three variacs
2. On the M/C variacs note the drive motor. Note the terminal board which connects the drive motor powering
3. Locate the lower limit, center limit, and upper limit cams on the MC variacs. Why is there a center limit cam?
4. Locate the output of the variable autotransformers on the sides of the assemblies.
5. Are there any brushes on any of the variacs? Are there any lubrication points? Do you know where to find, in the reference manual, any necessary adjustment procedures?

D. Generator Exciter - Regulator Panel

1. Locate the panel
2. Locate the magnetic amplifier on the panel
3. Locate the transistor Q1 on the panel

4. Locate the VR tube
5. Locate relay CR 1
6. Locate the three potentiometers. Do you know what they do? How they are adjusted?

E. -48 Volt Power Supply

1. Locate the -48V supply
2. Locate the -48V supply fuses
3. How many -48V supplies are there? How are they connected?

F. Contactors & Relays

1. Locate HR-29, the power on sequencing contactor.
2. Locate HR-30, the magnetic motor starter (Note the manual reset)
3. Locate the individual frame sequencing contactors HR-1 through HR-28. Note the bus bars on each side of these contactors. Which are the M/C bus bars? Which are the 208V variable bus bars? Which are the 208V fixed bus bars?
4. Locate DR relays 1 through 13. What are these DR relays for?
5. Locate DR 14 and 15, the memory time and interlock relays
6. Locate DR16 through 28. What are these relays for?
7. Locate DR-29, 30 and 31, the 400 cycle interlock relays. They are located on the "Relay, Fuse, and Diode Panel".
8. Locate HR 36 and 37 "D.C. On" relays also on the panel.

III. Typical Frame (CPU or Channels)

1. Locate the blowers
2. Locate the blower CB's
3. Locate the power cable connectors at the tailgate (hard to get at, aren't they?)
4. Locate CB1 for one of the modular supplies
5. Locate the Pwr. On-Off switch at the back of the frame.
6. Locate one of the gate thermal switches
7. Locate the D. C. fuse panel on an SMS supply

8. Locate DR-1 relay in the frame
9. Locate terminal boards 12, 13, 14, and 15 at the back of the frame
10. Locate Bus Bar 1 (frame bond)

IV. Console

1. Locate the SMS power supply
2. Where is CB1?
3. Locate the fuse panel
4. Locate the Marginal Control meter panel. Do you know how to bias the system?
5. Are there any blowers in the console?
6. Do you know how to open the top key panel and see the key restore motors?

V. 7302 Oil Memory

A. Core special supplies

1. Locate CB1 on the special supply
2. Locate CB2 on the special supply
3. What voltage does CB2 drop?
4. Locate the +30 volt M/C Pot. and drive motor
5. Locate the +60 volt M/C Pot. and drive motor
6. Locate the Mag. Amp card on the Special Supply
7. Locate the +82 volt supply. What is this supply used for?
8. Locate the -18 volt supply. What is this supply used for?
9. Locate the wafer switch which adjusts the output of the +82 volt supply

B. Core 2KW Modular Supply

1. Locate CB1
2. Locate the Mag. Amp. card
3. Locate the Marginal Check relays.
Why are there only two relays?
4. Locate the fuses
5. Note the selenium diode rectifiers and their heat sinks

VI. Card Machines

A. Printer

1. Locate the special 55 volt supply
2. Locate the power supply blower
3. Locate the P.S. blower fuses

4. Locate the 48 volt generator. Where are the brushes?
5. Locate the generator output adjusting resistor
6. Locate the main fuse panel. How is a blown fuse indicated?
7. Are there any power supplies in the Card Reader or Punch?

B. Card Reader and Punch Unit

1. Remove the CR covers
2. Where is the drive motor?
3. Does the drive motor need lubrication?
4. Locate the fuses in the CR
5. Do you know where to find the fuse locations, valves, and the circuits each fuse protects in the ALD's?
6. Remove the PU covers
7. Locate the drive motor in the PU
8. Locate the fuses in the PU
9. Take a good look around. Is this all you want to see?

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 1 - MANUAL STORAGE AND DISPLAY CONTROLS

PART 2 - ON-LINE TAPE OPERATION

PART 3 - OFF-LINE TAPE OPERATION

PART 4 - TAPE CYCLE OPERATION

PART 5 - CONTINUOUS STORAGE READ-IN AND READ-OUT

PART 6 - CARD MACHINE OPERATIONS

PART 7 - QUESTIONS

NOTE- DATA MAY BE TRANSMITTED BETWEEN I/O DEVICES AND CORE STORAGE ONLY WHEN THE DATA CHANNEL IS IN THE ON LINE STATUS. WHEN THE CHANNEL IS IN THE OFF LINE STATUS THE I/O DEVICES DO NOT HAVE ACCESS TO CORE STORAGE. HOWEVER, MANUAL CONTROLS OF CORE STORAGE SUCH AS STORE AND DISPLAY ARE ACTIVE REGARDLESS OF WHETHER THE CHANNEL IS ON LINE OR OFF LINE.

LABORATORY PROJECT
7617 CONSOLE ORIENTATION

PART 1 - MANUAL STORAGE AND DISPLAY CONTROLS

1. USING THE LOAD COMMAND BUTTON, PLACE AN ADDRESS IN THE CHANNEL ADDRESS COUNTER.
2. USING THE LOAD DATA REGISTER BUTTON, PLACE A NUMBER IN THE DATA REGISTER.
3. DEPRESS THE STORE DATA REGISTER BUTTON ONCE.
4. NOTE THAT THE CHANNEL ADDRESS COUNTER HAS BEEN INCREMENTED. THE DATA WORD HAS BEEN STORED AT THE ADDRESS THAT WAS SPECIFIED BY THE CHANNEL ADDRESS COUNTER (CAC). THE CAC WAS THEN STEPPED IN ORDER TO BE READY FOR ANOTHER STORE OPERATION.
5. PERFORM THE STORE SEVERAL MORE TIMES, WATCHING THE CAC STEP AND SUPPLY THE CONSECUTIVE LOCATIONS.
6. RETAINING THE CAC AS IT IS, CHANGE THE CONTENTS OF THE DATA REGISTER.
7. STORE THE NEW NUMBER IN SEVERAL LOCATIONS.
8. RESET THE CHANNEL AND RE-ENTER THE ORIGINAL ADDRESS IN THE CAC.
9. DEPRESS THE DISPLAY STORAGE BUTTON.
10. NOTICE THAT THE CAC HAS AGAIN BEEN INCREMENTED.
11. THE WORD NOW IN THE DATA REGISTER IS THE WORD YOU HAVE DISPLAYED. IT WAS STORED AT THE STARTING LOCATION.
12. DISPLAY SEQUENTIAL LOCATIONS UNTIL YOU REACH THE SECOND DATA WORD YOU USED.

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 2 - ON-LINE TAPE OPERATION

1. REWIND THE TAPE UNIT. LOAD AN IOCD COMMAND INTO THE COMMAND REG-
ISTERS FROM THE OP. PANEL KEYS ON THE DATA CHANNEL CONSOLE.
2. DEPRESS THE WRITE TAPE KEY. CHANNEL WILL WRITE ONE RECORD ON
TAPE. THIS RECORD WILL CONSIST OF THE NUMBER OF WORDS YOU HAVE
SPECIFIED BY THE WORD COUNT YOU HAVE ENTERED AS PART OF THE
COMMAND. THEY WILL COME FROM CONSECUTIVE CORE STORAGE LOCATIONS,
STARTING WITH THE ADDRESS SUPPLIED TO THE CHANNEL ADDRESS COUNTER
BY YOUR COMMAND.
3. WATCH THE STEPPING OF THE WORD COUNTER AND THE CHANNEL ADDRESS
COUNTER.
4. WHEN THE WORD COUNTER HAS BEEN STEPPED TO ZERO, THE CHANNEL
WILL DISCONNECT AND THE TAPE DRIVE WILL HALT.
5. REWIND THE TAPE. USING THE SAME COMMAND THAT YOU USED BEFORE,
DEPRESS THE READ TAPE KEY AND OBSERVE THE OPERATION.
6. REDUCE THE SIZE OF THE WORD COUNT AND READ THE SAME RECORD AGAIN.
NOTE THAT EVEN THOUGH THE WORD COUNT HAS GONE TO ZERO AND DATA
TRANSMISSION HAS BEEN HALTED, TAPE MOTION CONTINUES UNTIL THE END
OF RECORD IS REACHED.

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 3 - OFF-LINE TAPE OPERATION

1. LOAD THE DATA REGISTER WITH ALL BITS.
2. DEPRESS THE WRITE TAPE KEY. THE CHANNEL WILL WRITE A CONTINUOUS RECORD ON TAPE. EACH WORD OF THIS RECORD COMES FROM THE DATA REGISTER. THE DATA REGISTER IS NEVER RESET IN THIS MODE OF OPERATION.
3. OBSERVE THE FLOW OF INFORMATION FROM DATA REGISTER TO TAPE REGISTER TO R/W REGISTER.
4. NOTE THE OPERATION OF THE WRITE CLOCK, THE READ CLOCK, THE SKEW REGISTERS A AND B.
5. WHILE THE CHANNEL IS IN OPERATION, CHANGE THE CONTENTS OF THE DATA REGISTER (DO NOT RESET). OBSERVE ANY CHANGES IN THE APPEARANCE OF THE ABOVE REGISTERS ETC.
6. DEPRESS THE STOP WRITE BUTTON. REWIND THE TAPE, RESET THE CHANNEL, AND DEPRESS READ TAPE.
7. NOTE EACH OF THE ABOVE REGISTERS AND CLOCKS AGAIN DURING THE READ OPERATION.
8. ALSO NOTE THE TAPERING OFF EFFECT OF THE TAPE REGISTER LIGHTS DURING THE READ OPERATION. CAN YOU EXPLAIN THIS CONDITION.
9. NOTE THAT THE OP REGISTER, WORD COUNTER, CHANNEL ADDRESS COUNTER, AND LOCATION COUNTER ARE NOT USED DURING AN OFF-LINE I/O OPERATION.

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 4 - TAPE CYCLE OPERATION

1. TAPE CYCLE IS AN OFF LINE OPERATION. LOAD THE DATA REGISTER WITH ALL BITS
2. PLACE THE TAPE CYCLE TOGGLE SWITCH IN THE ON(UP) POSITION.
3. DEPRESS THE WRITE TAPE BUTTON. THE CHANNEL WILL WRITE A SERIES OF RECORDS ON THE SELECTED TAPE. EACH RECORD WILL BE ONE WORD (SIX CHARACTERS) IN LENGTH. THE DATA FLOW IS THE SAME AS FOR NON-CYCLE OPERATION.
4. EXAMINE THE REGISTERS AND CLOCKS. COMPARE THEIR APPEARANCE NOW WITH THEIR APPEARANCE DURING THE WRITING OF A CONTINUOUS RECORD.
5. STOP WRITE. DEPRESS THE WRITE END OF FILE KEY TO WRITE A TAPE MARK. REWIND THE TAPE.
6. RESET THE CHANNEL. DEPRESS THE READ TAPE BUTTON. THE TAPE WILL MOVE AND BE READ UNTIL THE END OF FILE IS DETECTED. THEN IT WILL REWIND TO LOAD POINT. WHEN LOAD POINT IS REACHED, THE READ OPERATION WILL BE RESTARTED.
7. WHILE THE READING IS IN PROCESS EXAMINE THE REGISTERS IN THE DATA PATH AND THE CLOCKS. COMPARE WHAT YOU SEE WITH WHAT YOU SAW DURING THE READING OF A CONTINUOUS RECORD.
8. READ A FEW OF THE WORDS INTO KNOWN EMPTY STORAGE LOCATIONS. DISPLAY THESE LOCATIONS.

LABORATORY PROJECT 6

7617 CONSOLE ORIENTATION

PART 5 - CONTINUOUS STORAGE READ-IN AND READ-OUT

1. PLACE IN THE CAC THE ADDRESS OF A CORE STORAGE LOCATION WHICH IS AVAILABLE BUT EMPTY.
2. PLACE THE WORD TO BE STORED IN THE DATA REGISTER.
3. PLACE THE CONTINUOUS STORAGE READ-IN TOGGLE SWITCH (CSRI) IN THE ON (UP) POSITION.
4. THE CONTENTS OF THE DATA REGISTER IS NOW BEING REPETITIVELY STORED AT THE LOCATION SPECIFIED BY THE CAC.
5. NOTE THE FLICKER OF THE BDW REQUIRED TRIGGER. ALSO NOTE THE OPERATION OF THE CST1 AND CST2. NOTE THE BDW AND DR LDD TGRS ALSO.
6. TURN OFF THE CSRI SWITCH. RESET THE CHANNEL TO CLEAR THE DATA REGISTER.
7. PLACE THE SAME ADDRESS IN THE CAC. TURN ON THE CONTINUOUS STORAGE READ-OUT SWITCH (CSRO).
8. THE CONTENTS OF THE SPECIFIED CORE STORAGE LOCATION IS NOW BEING REPETITIVELY READ OUT AND PLACED INTO THE DATA REGISTER.
9. NOTE THE BDW REQUIRED, BDW, CST1, CST2 AND DR LDD TRIGGERS.
10. TURN OFF THE CSRO SWITCH AND RESET THE CHANNEL.
11. SELECT AN AREA OF STORAGE WITH TEN EMPTY LOCATIONS. PLACE THE LOW ORDER ADDRESS OF THIS AREA IN THE CAC. ALSO, PLACE THE QUANTITY TEN IN THE WORD COUNTER.
12. PLACE A WORD TO BE STORED IN THE DATA REGISTER.
13. TURN ON THE CSRI SWITCH. THE OPERATION AT THIS TIME IS IDENTICAL TO THAT PREVIOUSLY OBSERVED.
14. PLACE THE CHANNEL INTO AUTOMATIC STATUS. THE WORD COUNTER AND CAC WILL EACH STEP UNTIL THE WORD COUNT EQUALS ZERO. THE WORD IN THE DATA REGISTER WILL BE STORED IN EACH OF THE TEN LOCATIONS ADDRESSED BY THE CAC.
15. TURN OFF THE CSRI SWITCH. RETURN THE CHANNEL TO MANUAL. DISPLAY THE TEN LOCATIONS TO VERIFY THAT THE INFORMATION WAS STORED.

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 6 - CARD MACHINE OPERATIONS

1. WITH THE CHANNEL OFF LINE FILL THE DATA REGISTER WITH BITS.
2. SET THE UNIT SELECT SWITCH TO ZERO.
3. DEPRESS THE WRITE PUNCH BUTTON. NOTE THE RESULT. THE DATA REGISTER SUPPLIES 24 WORDS FOR EACH PUNCH CYCLE. MODIFY THE DATA REGISTER AND TRY THIS AGAIN.
4. PERFORM THE OPERATIONS DESCRIBED ABOVE USING THE WRITE PRINTER BUTTON. NOTE THE RESULT.
5. TRY THE SAME OPERATION ON THE PRINTER AGAIN, BUT WITH THE PRINT BINARY TOGGLE SWITCH IN THE ON (UP) POSITION. NOTE THE RESULT. THE DATA REGISTER SUPPLIES ONLY TWO WORDS FOR EACH CYCLE IN THIS MODE OF OPERATION. (AT 1L AND 1R TIMES).
6. REPEAT THESE OPERATIONS WITH THE CHANNEL ON LINE. YOU MAY STORE THE REQUIRED IMAGES MANUALLY OR YOU MAY READ THEM INTO STORAGE FROM CARDS BY USING THE READ CARD READER BUTTON.

LABORATORY PROJECT 6
7617 CONSOLE ORIENTATION

PART 7 - QUESTIONS

1. IN AN ON LINE WRITE OPERATION, WHAT IS THE SOURCE OF THE DATA WRITTEN ON TAPE.

2. IN AN OFF LINE WRITE OPERATION, WHAT IS THE SOURCE OF THE DATA WRITTEN ON TAPE.

3. DURING AN OFF LINE READ OF A LONG RECORD CONTAINING ALL BITS THE INTENSITY OF THE TAPE REGISTER INDICATORS APPEARS TO TAPER OFF OR DIMINISH TO THE RIGHT. EXPLAIN WHY.

4. IN AN OFF LINE READ OPERATION, DATA FOLLOWS THE SAME PATHS OR FLOW AS IN AN ON LINE OPERATION EXCEPT THAT IT IS STOPPED AT SOME POINT, THEREFORE NEVER REACHING CORE STORAGE. TELL THE LAST POINT IN THE DATA FLOW THAT THE INFORMATION REACHES IN THIS TYPE OF OPERATION.

5. TELL WHAT IS GENERATED ON TAPE BY WRITING OFF LINE WITH THE TAPE CYCLE SWITCH IN THE ON POSITION.

6. CAN THE DATA FLOW BE OBSERVED UNDER THESE CONDITIONS.

7. PRESUME THAT YOU HAVE A TAPE WRITTEN IN THIS MANNER. IT IS NOW BEING READ OFF LINE WITH THE TAPE CYCLE SWITCH ON. DESCRIBE THE APPEARANCE OF THE REGISTERS.

8. GIVE THE REASON WHY THE BDW REQUIRED TRIGGER IS VISIBLE ON A CONTINUOUS STORAGE READ-IN OR READ-OUT BUT THE BDW TRIGGER IS NOT.

9. TELL HOW MANY WORDS PER LINE ARE TRANSMITTED TO THE PRINTER FOR A WRITE PRINTER BINARY OPERATION.

10. TELL HOW MANY WORDS PER LINE ARE TRANSMITTED TO THE PRINTER FOR A NORMAL (NOT BINARY) WRITE PRINTER OPERATION.

11. DESCRIBE THE QUICKEST MEANS OF LOADING THE SAME WORD INTO ALL LOCATIONS OF CORE STORAGE FROM THE 7617.

LABORATORY PROJECT 7

DIAGNOSTICS

THIS INFORMATION WILL PROVIDE YOU WITH A BASIC UNDERSTANDING OF THE APPLICATION OF DIAGNOSTICS. STUDY IT THOROUGHLY. IT IS FOLLOWED BY A PROJECT WHICH YOU WILL SOON PERFORM IN LAB. AFTER YOU UNDERSTAND THE MATERIAL CONTAINED HEREIN, READ THROUGH THE PROJECT SO YOU WILL BE PREPARED WHEN YOUR TURN ON THE SYSTEM ARRIVES. THE CONTENTS IS BROKEN INTO THE FOLLOWING PARTS -

1. LOADERS
2. 9M51
3. DEPRX
4. 9M56
5. 9T51
6. 9I0C
7. DIAGNOSTIC TAPE
8. FORCE LOAD TAPE

1. LOADERS

A LOADER IS A SHORT PROGRAM, FREQUENTLY ONLY ONE CARD, WHICH IS DESIGNED TO BE BROUGHT INTO STORAGE AS A RESULT OF DEPRESSING A LOAD KEY AND WHICH THEN LOADS A LARGER, PRODUCTIVE PROGRAM. THE LOAD BUTTON SEQUENCE ITSELF MAY BE CONSIDERED AS A SMALL LOAD PROGRAM. IT FILLS LOCATIONS 0, 1, AND 2 WITH THE FIRST THREE WORDS OF THE FIRST AVAILABLE RECORD. THIS RECORD IS USUALLY ALL OR PART OF THE LOADER. THESE THREE WORDS MAY ALSO BE CONSIDERED AS A SHORT LOAD PROGRAM. THEY USUALLY SERVE TO LOAD THE BALANCE OF THE LOADER. THE LOADER THEN PERFORMS THE FOLLOWING FUNCTIONS - -

- A. IT EXAMINES EACH RECORD OF THE PROGRAM TO DETERMINE WHERE THE FIRST WORD IS TO BE PLACED IN STORAGE. IT PLACES THE FIRST WORD THERE AND THEN PLACES SUCCEEDING WORDS OF THAT RECORD IN SUCCESSIVE STORAGE LOCATIONS.
- B. IT COMPUTES A CHECK SUM FOR EACH RECORD AND COMPARES THE COMPUTED CHECK SUM WITH THE CHECK SUM READ FROM THE RECORD. IN THE EVENT THAT THE COMPUTED CHECK SUM AND THE CHECK SUM READ ARE DIFFERENT, THE LOADER WILL HALT. CARD LOADERS WILL PROCEED AFTER DEPRESSION OF THE START KEY. TAPE LOADERS MAY PROCEED OR TRY THE SAME RECORD AGAIN. WITH SOME LOADERS, THE CHECK SUM ROUTINE MAY BE BYPASSED BY THE OPERATOR THROUGH THE USE OF CONSOLE SWITCHES.
- C. THE LOADER EXAMINES EACH RECORD TO DETERMINE IF IT IS A TRANSFER RECORD (TAPE INPUT) OR TRANSFER CARD (CARD INPUT). WHEN SUCH A RECORD IS FOUND, NO FURTHER RECORDS ARE READ. THE LOADER THEN EXECUTES A TRANSFER TO THE LOCATION DESIGNATED IN THE TRANSFER RECORD, THEREBY INITIATING THE PRODUCTIVE PROGRAM.

2. 9M51

9M51 IS THE MAIN FRAME DIAGNOSTIC. IT TESTS ALL CPU INSTRUCTIONS. IT IS ARRANGED IN SUCH A MANNER THAT NO INSTRUCTION IS USED TO TEST ANOTHER INSTRUCTION UNTIL IT ITSELF HAS BEEN TESTED.

9M51 IS DIVIDED INTO PARTS 1, 2, AND 3. PART 3 TESTS ALL FLOATING POINT INSTRUCTIONS. PART 2 TESTS ALL SENSE INDICATOR INSTRUCTIONS. PART 1 TESTS ALL OTHER CPU INSTRUCTIONS.

9M51 IS THE FIRST DIAGNOSTIC YOU WOULD RUN TO CHECK OUT THE SYSTEM.

3. 9DEPRX

9DEPRX STANDS FOR DIAGNOSTIC PRINT ROUTINE. ALL DIAGNOSTICS HAVE A NEED TO CONVERT INTERNAL RECOGNITION OF ERROR TO PRINTED OUTPUT USABLE BY THE C.E. ALSO, ALL DIAGNOSTICS HAVE A NEED FOR C.E. CONTROLS TO MAKE THEIR USE AND APPLICATION AS FLEXIBLE AS POSSIBLE. TO WRITE SUCH A ROUTINE EACH TIME A DIAGNOSTIC IS WRITTEN WOULD BE REPETITIVE AND WASTEFUL. ALSO, THERE MIGHT BE MANY VARIANCES IN PROCEDURE FROM DIAGNOSTIC TO DIAGNOSTIC BECAUSE OF THE DIFFERENT AUTHORS.

9DEPRX IS A STANDARD PROGRAM WRITTEN TO SOLVE THESE PROBLEMS. IT IS INCLUDED AS A PART OF MOST DIAGNOSTICS. IT ALWAYS OCCUPIES THE SAME LOCATIONS IN CORE STORAGE. THE SENSE SWITCH FUNCTIONS WHICH PROVIDE FLEXIBILITY ARE LISTED BELOW IN A LOGICAL SEQUENCE RATHER THAN IN NUMERIC SEQUENCE. THIS IS FOR EASE OF UNDERSTANDING.

SSW 6

UP

WHEN THE DIAGNOSTIC HAS BEEN COMPLETED, A LOAD CARDS BUTTON SEQUENCE IS SIMULATED. A PROGRAM DECK LOADED AND READY IN THE CARD READER WOULD, THEREFORE, BE READ INTO STORAGE.

DOWN

WHEN THE DIAGNOSTIC IS COMPLETED, IT IS AUTOMATICALLY RESTARTED.

SSW 4

UP

AS EACH TEST ROUTINE IS COMPLETED, THE MACHINE PROCEEDS TO THE NEXT ROUTINE IN STORAGE.

DOWN

A TRANSFER OCCURS WHICH CAUSES THE MACHINE TO RETURN TO THE START OF THE ROUTINE JUST COMPLETED. EACH SUCH TRANSFER IS COUNTED. WHEN THE COUNT BECOMES EQUAL TO THE VALUE N, THE MACHINE PROCEEDS TO THE NEXT SEQUENTIAL ROUTINE. THE NORMAL VALUE FOR N IS 40 BUT IT MAY BE MODIFIED TO ANY VALUE YOU MAY DESIRE BY A MANUAL STORE.

SSW 1

UP

AS EACH TEST ROUTINE IN THE DIAGNOSTIC IS COMPLETED, SSW 4 IS TESTED.

DOWN

WHEN THE TEST ROUTINE IS COMPLETED, A TRANSFER OCCURS WHICH RETURNS THE MACHINE TO THE START OF THE ROUTINE JUST COMPLETED.

SSW 3

UP

WHEN AN ERROR CONDITION IS RECOGNIZED BY THE PROGRAM, AN ERROR PRINTOUT DESCRIBING THE FAILURE OCCURS ON THE ON-LINE PRINTER. THE PROGRAM THEN PROCEEDS IN A MANNER DETERMINED BY THE SETTING OF SENSE SWITCHES 1 AND 4.

DOWN

WHEN AN ERROR IS RECOGNIZED, NO PRINTOUT OCCURS BUT THE MACHINE IS CAUSED TO HALT INSTEAD.

SSW 2
UP

WHEN AN ERROR IS RECOGNIZED BY THE PROGRAM, SSW 3 IS TESTED.

DOWN

WHEN AN ERROR IS RECOGNIZED, SSW 3 IS NOT TESTED AND NO INDICATION OF THE FAILURE IS GIVEN.

SSW 5

THIS SWITCH IS NOT USED BY 9DEPRX AND IS AVAILABLE FOR ANY USE DESIRED BY THE AUTHOR OF THE DIAGNOSTIC CONCERNED.

WHEN A DIAGNOSTIC IS USED TO TEST THE MACHINE, ALL OF THE SENSE SWITCHES EXCEPT 6 WOULD NORMALLY BE UP. WHEN A FAILURE OCCURS, APPROPRIATE SWITCHES WOULD BE USED TO ACHIEVE THE RESULTS YOU DESIRE.

4. 9M56

9M51, THE MAIN FRAME DIAGNOSTIC, IS BROUGHT INTO STORAGE BY A LOAD PROGRAM WHICH PERFORMS THE FUNCTIONS OUTLINED IN (1) ABOVE. SUCH A LOAD PROGRAM REQUIRES THE PROPER FUNCTIONING OF A LARGE PORTION OF THE SYSTEM. THEREFORE, WE ARE FACED WITH THE SITUATION WHERE CPU MUST BE WORKING BEFORE WE CAN LOAD A DIAGNOSTIC THAT IS INTENDED TO SHOW WHAT IS WRONG WITH CPU. OBVIOUSLY, THIS CONDITION WOULD OFTEN MAKE PART 1 OF 9M51 VERY DESIRABLE BUT NOT AVAILABLE. 9M56 SOLVES THIS PROBLEM.

THIS IS ACCOMPLISHED BY THE FACT THAT 9M56 USES NO LOAD PROGRAM. THEREFORE, CPU HAS NO PROGRAM THAT IT MUST PERFORM IN ORDER TO LOAD 9M56 INTO STORAGE. OF COURSE, WITH NO LOADER IN USE, NONE OF THE FUNCTIONS OF A LOADER ARE PERFORMED. NO CHECK SUMS ARE COMPUTED OR COMPARED. NO MEANS EXISTS FOR PLACING A GIVEN RECORD IN A DESIGNATED STORAGE LOCATION.

POSITION 9R OF THE PROGRAM DECK CARDS, WHICH USUALLY HOLDS A CHECK SUM, MAY NOW HOLD AN INSTRUCTION. POSITION 9L, WHICH USUALLY HOLDS A COMMAND (IOCD, WC=22, PLUS APPROPRIATE ADDRESS) MAY ALSO NOW HOLD AN INSTRUCTION. THEREFORE, EACH CARD HOLDS 24 RATHER THAN 22 PROGRAM WORDS. 9L AND 9R OF THE FIRST CARD HOLD THE FIRST 2 INSTRUCTIONS OF THE PROGRAM. THESE GO INTO LOCATIONS 00000 AND 00001 OF STORAGE. EXCEPT FOR THE LACK OF A LOADER AND THE RESULTANT PHYSICAL REARRANGEMENT OF THE CARDS, 9M56 IS NEARLY IDENTICAL WITH PART 1 OF 9M51.

9M56 IS BROUGHT INTO STORAGE BY WHAT IS KNOWN AS A FORCE LOAD. THE CARDS ARE PLACED IN THE CARD READER IN THE NORMAL MANNER AND THE CARD READER IS MADE READY. AN IOCD COMMAND IS MANUALLY SET INTO CHANNEL A. SINCE THE PROGRAM STARTS AT ZERO, THE ADDRESS PORTION OF THE COMMAND SHOULD BE ZERO. SINCE THIS ONE COMMAND MUST

READ THE ENTIRE DECK, A LARGE WORD COUNT IS NECESSARY. 40000 IS SUFFICIENTLY LARGE AND IS EASILY ENTERED, BEING REPRESENTED BY ONLY ONE BUTTON. BECAUSE THE ENTIRE DECK IS BEING LOADED IN SEQUENCE BY ONE COMMAND, IT IS VERY IMPORTANT THAT THE CARDS IN THE DECK ARE IN THE PROPER ORDER. IF THEY ARE OUT OF ORDER, THE INSTRUCTION IN STORAGE WILL BE OUT OF SEQUENCE.

IN A TYPICAL APPLICATION, 9M56 WOULD BE USED ONLY WHEN IT WAS NOT POSSIBLE TO LOAD 9M51 BY NORMAL MEANS. THEN 9M56 CAN BE USED TO TEST CPU. IF 9M56 PERFORMS IN A SATISFACTORY MANNER, CPU IS WORKING WELL ENOUGH TO PERFORM A LOAD PROGRAM.

5. 9T51

9T51 IS A TEST OF ALL CIRCUITRY USED IN BASIC TAPE OPERATION. IT TESTS BOTH DATA FLOW AND CONTROL CIRCUITRY. IT MAY BE USED TO TEST ANY COMBINATION OF TAPE DRIVE MODELS, IN ANY NUMBER FROM ONE TO ALL THAT ARE AVAILABLE, ON ANY OR ALL OF THE AVAILABLE CHANNELS.

IF CHANNEL TROUBLE IS SUSPECTED, 9T51 IS THE FIRST DIAGNOSTIC YOU SHOULD RUN. IF 9T51 FAILS, YOU OF COURSE HAVE AN INDICATION OF THE NATURE OF THE PROBLEM. IF 9T51 RUNS SUCCESSFULLY, THERE ARE SEVERAL ADDITIONAL DIAGNOSTICS AVAILABLE TO TEST CHANNEL OPERATION IN AREAS SUCH AS THE VARIOUS CARD MACHINES, SET DENSITY, CHANNEL TRAP, BUFFER TIMING, INTERCHANGEABILITY, ETC. THESE SHOULD BE USED AS REQUIRED.

6. 9IOC

9IOC STANDS FOR INPUT-OUTPUT CONTROL. ALL I/O DIAGNOSTICS, SUCH AS 9T51, ARE WRITTEN TO OPERATE ON UNIT ONE OF CHANNEL A. IF THIS WAS THE LIMIT OF THEIR ABILITY, THEIR USEFULNESS WOULD BE SEVERELY RESTRICTED. CONSEQUENTLY, IT IS DESIRABLE TO HAVE A MEANS BY WHICH THE USER MAY INSTRUCT THE PROGRAM AS TO WHICH UNITS ON WHICH CHANNELS ARE TO BE TESTED.

AS WAS POINTED OUT EARLIER, IF A UNIQUE PROGRAM WERE WRITTEN TO ACCOMPLISH THIS ABILITY FOR EACH DIAGNOSTIC, THERE WOULD BE A GREAT DUPLICATION OF EFFORT AND THE PROCEDURES WOULD VARY FROM DIAGNOSTIC TO DIAGNOSTIC.

9IOC IS A STANDARD PROGRAM WRITTEN FOR THIS PURPOSE. IT IS INCLUDED AS A PART OF MOST I/O DIAGNOSTIC PROGRAMS. WHEN INCLUDED, IT ALWAYS OCCUPIES THE SAME STORAGE LOCATIONS. 9IOC USES THE OP PANEL KEYS FOR INPUT. CONSULT YOUR POCKET REFERENCE MANUAL FOR INFORMATION ON THE ASSIGNMENT OF KEYS.

WHEN A PROGRAM USING 9IOC IS READ INTO STORAGE AND PLACED INTO OPERATION, IT TRANSFERS TO 9IOC BEFORE PERFORMING ANY TESTS. IN 9IOC IT STOPS AT LOCATION 75320. THIS IS TO ALLOW THE C.E. TO SET THE APPROPRIATE OP PANEL KEYS. WHEN THE KEYS HAVE BEEN SET, THE

START KEY IS DEPRESSED. 910C THEN INTERROGATES THE OP PANEL KEYS AND MODIFIES EACH I/O INSTRUCTION IN THE PROGRAM SO THAT IT WILL NOW OPERATE ON THE PROPER UNIT ON THE PROPER CHANNEL. WHEN THE TEST IS COMPLETED ON THE FIRST UNIT, THE MACHINE RETURNS TO 910C, WHICH THEN MODIFIES THE DIAGNOSTIC FOR THE NEXT UNIT TO BE TESTED. WHEN ALL UNITS SPECIFIED IN THE OP PANEL KEYS HAVE BEEN TESTED, 910C RESTORES THE DIAGNOSTIC TO THE ORIGINAL CONDITION (UNIT ONE, CHANNEL A) AND ONCE AGAIN HALTS AT 75320, AWAITING FURTHER INSTRUCTIONS.

IF THE MACHINE IS MANUALLY RESET AND START IS DEPRESSED, THE TEST IS RESUMED ON THAT UNIT WHICH WAS UNDER TEST WHEN THE RESET OCCURRED.

7. THE DIAGNOSTIC TAPE

DIAGNOSTICS ARE MOST OFTEN LOADED FROM TAPE RATHER THAN FROM CARDS. THIS IS BECAUSE OF THE GREAT DIFFERENCE IN THE AMOUNT OF TIME REQUIRED BY THE TWO METHODS. FOR EXAMPLE, 9M51 REQUIRES 5 MINUTES TO READ IN FROM CARDS - ONLY 4 SECONDS FROM TAPE.

THE DIAGNOSTIC TAPE MUST BE MOUNTED ON TAPE DRIVE A1 IN ORDER TO RESPOND TO THE LOAD TAPE BUTTON. EACH DIAGNOSTIC IS A FILE ON THIS TAPE. A SEARCH AND LOAD PROGRAM MAKE UP THE FIRST RECORD OF EACH FILE. THIS PROGRAM IS BROUGHT INTO CORE IN A MANNER SIMILAR TO 9LD01 OR 9LD02 AND PERFORMS THE SAME FUNCTIONS AS THOSE PROGRAMS PLUS SOME ADDITIONAL FUNCTIONS. WHEN AT REST, THE TAPE IS USUALLY POSITIONED SO THAT THE READ HEAD IS EITHER BETWEEN THE LOAD POINT AND THE FIRST RECORD OF THE FIRST FILE, OR BETWEEN AN END OF FILE MARK AND THE FIRST RECORD OF THE NEXT FILE. THEREFORE, WHEN THE LOAD TAPE BUTTON IS DEPRESSED, THE FIRST RECORD (SEARCH AND LOAD PROGRAM) OF SOME FILE IS READ INTO STORAGE.

IF SENSE SWITCH 6 IS UP WHEN THE LOAD TAPE BUTTON IS DEPRESSED, THE SEARCH AND LOAD PROGRAM WILL READ INTO STORAGE, THAT FILE OF WHICH IT IS A PART. IT WILL PLACE EACH RECORD IN THE APPROPRIATE STORAGE LOCATIONS, PERFORM THE USUAL CHECK SUM OPERATIONS, LOCATE THE TRANSFER RECORD (TAPE EQUIVALENT OF A TRANSFER CARD), AND EXECUTE A TRANSFER TO THE START OF THE PROGRAM. SUCCESSIVE DEPRESSIONS OF THE LOAD TAPE BUTTON THEREFORE RESULT IN THE READING IN OF SUCCESSIVE FILES.

IF SENSE SWITCH 6 IS DOWN, SIMILAR OPERATIONS WILL OCCUR. THE DIFFERENCE IS THAT THE SEARCH AND LOAD PROGRAM WILL INTERROGATE THE OP PANEL KEYS BEFORE READING IN THE DIAGNOSTIC. THE ADDRESS PORTION OF THE OP PANEL KEYS IS USED TO DESIGNATE IN OCTAL THE NUMBER OF THE FILE YOU WISH TO LOAD INTO STORAGE. THE SEARCH AND LOAD PROGRAM WILL THEN SEARCH IN THE APPROPRIATE DIRECTION, TOWARD OR AWAY FROM LOAD POINT, FIND THE REQUESTED DIAGNOSTIC, AND READ IT INTO STORAGE. THIS IS POSSIBLE BECAUSE EACH SEARCH AND LOAD PROGRAM CONTAINS A WORD WHICH TELLS THE NUMBER OF THE FILE TO WHICH IT BELONGS. THE DIFFERENCE BETWEEN THE FILE IMMEDIATELY AVAILABLE AND THE FILE REQUESTED IS COMPUTED AND CAN BE SEEN BEING STEPPED DOWN IN INDEX REGISTER C AS THE PROGRAM SEARCHES FOR THE REQUESTED FILE.

WHEN THE SEARCH AND LOAD PROGRAM INTERROGATES THE ADDRESS KEYS (SSW 6 DOWN), IT ALSO EXAMINES THE SIGN KEY. IF THE SIGN KEY IS UP, THE DIAGNOSTIC TAPE IS CAUSED TO REWIND AFTER THE REQUESTED DIAGNOSTIC IS READ. IF THE SIGN KEY IS DOWN, THE REWIND IS SUPPRESSED AND THE TAPE IS POSITIONED SO THAT THE READ HEAD IS JUST BEYOND THE END OF FILE WHICH TERMINATES THE FILE LOADED INTO STORAGE. THEREFORE, THE TAPE IS READY TO READ THE SEARCH AND LOAD RECORD OF THE NEXT FILE.

CHECK SUM HALTS ARE AT 77766. DEPRESS START TO TRY AGAIN. YOU MAY PLACE SSW 2 DOWN TO BYPASS FURTHER CHECK SUM HALTS.

8. THE FORCE LOAD TAPE

YOU HAVE ALREADY SEEN THE ADVANTAGES OF HAVING AVAILABLE A FORCE LOAD PROGRAM SUCH AS 9M56. IF CPU IS FAILING TOO BADLY TO PERFORM A LOAD PROGRAM, 9M56 MAY BE FORCE LOADED AND USED TO TEST BASIC CPU INSTRUCTIONS. 9M56 REQUIRES THREE MINUTES TO BE READ INTO STORAGE FROM CARDS. THEREFORE, IT IS USUALLY PUT ON TAPE AT SOME TIME WHEN THE MACHINE IS WORKING PROPERLY. THIS IS KNOWN AS A FORCE LOAD TAPE. IT HAS A VERY IMPORTANT ADDITIONAL ADVANTAGE OVER THE CARD INPUT BECAUSE IT CAN BE FORCE LOADED FROM ANY CHANNEL. IF CHANNEL A, FOR EXAMPLE, IS FAILING, 9M56 MAY BE FORCE LOADED FROM CHANNEL B AND USED TO PROVE OR DISPROVE THE RELIABILITY OF CPU.

CONSIDER FOR A MOMENT THE SITUATION WHERE IT WAS NOT POSSIBLE TO LOAD 9M51 FROM TAPE A1, THE DIAGNOSTIC TAPE. THE PROBLEM COULD BE THAT CPU COULD NOT PROPERLY PERFORM THE LOADER OR THAT CHANNEL A HAD FAILED IN SOME MANNER. THE FIRST STEP SHOULD BE TO FORCE LOAD 9M56. SINCE THE STATUS OF CHANNEL A IS DOUBTFUL, 9M56 IS BEST FORCE LOADED FROM SOME OTHER CHANNEL.

IF 9M56 INDICATES A CPU FAILURE, YOU MAY USE THE INDICATION AS AN AID IN REPAIRING THE MACHINE.

IF 9M56 RUNS NORMALLY WITH NO INDICATION OF FAILURE, YOU MUST PRESUME THAT THE INABILITY TO READ IN 9M51 IS BASED ON A MALFUNCTION OF CHANNEL A.

THE NEXT STEP IS OBVIOUS. USE 9T51 TO TEST CHANNEL A AND SHOW WHAT IS WRONG. UNFORTUNATELY, IF 9M51 WOULD NOT LOAD FROM CHANNEL A, 9T51 WILL IN ALL PROBABILITY NOT LOAD EITHER. UNLESS WE HAVE PREPARED FOR SUCH AN OCCASION PREVIOUSLY, WE ARE NOW IN A VERY POOR POSITION. WE HAVE AN EXCELLENT PROGRAM TO DIAGNOSE CHANNEL MALFUNCTIONS, BUT WE CANNOT BRING IT INTO STORAGE BECAUSE CHANNEL A IS MALFUNCTIONING.

THE PREPARATION IS SIMPLE BUT MUST OCCUR BEFORE THE NEED ARISES, AT A TIME WHEN THE MACHINE IS FUNCTIONING PROPERLY. IT INVOLVES THE GENERATION OF A FORCE LOAD TAPE VERSION OF 9T51. THIS MAY THEN BE LOADED FROM ANY CHANNEL. THIS SAME APPROACH MAY BE

USED WITH ANY DIAGNOSTIC. IN THE LAB WE HAVE A FORCE LOAD TAPE CONTAINING, IN SUCCESSIVE FILES, 9M56, 9T51, 9S51L, 9S51H.

THIS TAPE WAS GENERATED BY READING THE DESIRED DIAGNOSTIC INTO STORAGE BY NORMAL MEANS. IF THERE WERE CHECK SUMS AND COMMANDS INTERSPERSED WITH THE INSTRUCTIONS, THEY WERE REMOVED BY THE LOADER WHEN IT STORED EACH RECORD. NOW, IT IS POSSIBLE TO DUMP CORE ON TAPE, PRESERVING THE PROGRAM FOR FUTURE USE. THIS PROGRAM NOW CONSTITUTES A FILE ON TAPE AND IS FOLLOWED BY AN END OF FILE MARK.

DIFFERENT PROGRAMS USE DIFFERENT AMOUNTS OF STORAGE. SO THAT IT WILL NOT BE NECESSARY TO REMEMBER THE APPROPRIATE ADDRESS AND WORD COUNT FOR EACH OF SEVERAL FILES ON THE TAPE, USE ADDRESS 00000 AND WORD COUNT 77776 FOR EACH OF THEM WHEN WRITING THE TAPE. WHEN THE TAPE IS READ, USE THE SAME ADDRESS BUT USE A WORD COUNT OF 77777. THIS MEANS THE CHANNEL WILL READ THE ENTIRE FILE PLUS THE EOF FOLLOWING IT AND THE TAPE DRIVE WILL STOP, READY TO READ THE NEXT FILE. THE TWO WORDS OF THE ORIGINAL STORAGE CONTENTS WHICH ARE LOST BY THIS PROCEDURE ARE OF NO VALUE AND PRESENT NO PROBLEM.

AN ADDITIONAL CONSIDERATION IN THE PREPARATION OF A FORCE LOAD TAPE, IS THE NORMAL STARTING POINT OF THE DIAGNOSTICS CONCERNED. 9M51/56 STARTS AT LOCATION 00000. OTHER PROGRAMS START AT OTHER LOCATIONS. THEREFORE, 9M56 MAY BE PUT INTO OPERATION BY A RESET (PROGRAM COUNTER 00000) AND START. HOWEVER, OTHER PROGRAMS REQUIRE A TRANSFER TO THE STARTING LOCATION. THIS IS DONE BY THE LOAD PROGRAM, AS A RESULT OF THE TRANSFER RECORD, UNDER NORMAL CONDITIONS. (THE TRANSFER CARD RESULTS FROM THE END CARD IN THE SYMBOLIC DECK. YOU MAY DETERMINE THE TRANSFER TO LOCATION FOR ANY PROGRAM BY NOTING THE ADDRESS ASSIGNED TO THE END CARD IN THE LISTING).

INSTEAD OF REMEMBERING THE TRANSFER TO LOCATION FOR EACH PROGRAM ON THE FORCE LOAD TAPE, YOU MAY STORE A TRA TO THE APPROPRIATE ADDRESS AT LOCATION 00000, BEFORE YOU WRITE THE DIAGNOSTIC ON TAPE. THEN, ANY PROGRAM LOADED FROM THIS TAPE MAY BE PUT INTO OPERATION BY STARTING FROM LOCATION 00000 (RESET).

LABORATORY PROJECT 7
DIAGNOSTICS

1. LOCATE THE TWO LISTS OF DIAGNOSTICS ON THE 7151 CONSOLE.
 - A. THE SMALLER SHEET HAS AN OCTAL NUMBER TO THE LEFT OF EACH DIAGNOSTIC. THIS IS THE NUMBER OF THE FILE THAT THE DIAGNOSTIC OCCUPIES ON THE DIAGNOSTIC TAPE.
 - B. THE LARGER LIST SHOWS THE VARIOUS DIAGNOSTICS IN ALPHANUMERIC ORDER. TO THE RIGHT OF EACH ENTRY IS ITS NAME OR DESCRIPTION. TO THE RIGHT OF THE DESCRIPTION IS A NUMBER WHICH DESIGNATES WHICH OF 15 BINDERS HOLDS THE INSTRUCTIONS AND THE LISTING FOR THAT DIAGNOSTIC. THE BINDERS ARE LOCATED IN THE BOOKCASE BESIDE THE 716.

2. READ IN DIAGNOSTIC 9M51 FROM THE DIAGNOSTIC TAPE. (HAVE SSW 6 DOWN)
 - A. NOTICE THE HALT AT 00000. (PROGRAM COUNTER EQUALS 00001)
DEPRESS START.
 - B. NOTICE THE HALT AT 00001. (PROGRAM COUNTER EQUALS 00002)
DEPRESS START.
 - C. OBSERVE THE PRINT OUT. PART ONE OF THE DIAGNOSTIC HAS BEEN PERFORMED ONE TIME FOLLOWING THE PRINT OUT -
9M51 NOW TESTING BASIC MAIN FRAME INSTRUCTIONS
 - D. THE LONG INDEX REGISTER TEST IS NOW BEING EXECUTED. OBSERVE THE OPERATION OF THE INDEX REGISTERS. THIS TEST IS PERFORMED ONLY ON THE FIRST PASS THROUGH THE DIAGNOSTIC. IT TAKES EIGHT MINUTES TO COMPLETE. YOU MAY BYPASS IT BY A MANUAL TRANSFER TO LOCATION 52112 ANYTIME AFTER THE TEST HAS BEGUN. DO THIS NOW.
 - E. OBSERVE THE PRINT OUT. PART TWO WAS PERFORMED ONE TIME FOLLOWING THE PRINT OUT -
9M51 NOW TESTING SENSE INDICATOR INSTRUCTIONS
PART THREE WAS PERFORMED ONE TIME FOLLOWING THE PRINT OUT
9M51 NOW TESTING FLOATING POINT INSTRUCTIONS
THE MACHINE THEN BEGAN PERFORMING PARTS 1,2, AND 3 IN SEQUENCE AND WILL CONTINUE IN THIS MANNER.
 - F. NOTICE THE SUBSEQUENT PRINT OUTS.
 - G. LOOK AT THE 7151 CONSOLE SO THAT YOU MAY BECOME FAMILIAR WITH ITS APPEARANCE DURING THE NORMAL OPERATION OF THIS DIAGNOSTIC. THE I/O CHECK INDICATOR IS TESTED (TURNED ON AND OFF) DURING EACH PASS THROUGH PART TWO.
 - H. PLACE SSW 6 UP. NOTE THE FOLLOWING INDICATORS -
 1. R/W SELECT (7151)
 2. CHANNEL SELECT A (7151)
 3. CR SEL (7617)
 4. SELECT (711)

3. READ IN 9T51 FROM THE DIAGNOSTIC TAPE.
 - A. EXAMINE THE INITIAL PRINT OUT.
 - B. THE MACHINE HAS HALTED. DETERMINE THE LOCATION AT WHICH IT HAS HALTED. DETERMINE WHAT PORTION OF THE PROGRAM THIS LOCATION IS IN AND DETERMINE THE PURPOSE FOR THE HALT.
 - C. PROCEED TO TEST ONE TAPE DRIVE ON EACH CHANNEL.
 - D. OBSERVE ALL PRINT OUTS.
 - E. WHEN ALL THE DRIVES YOU SPECIFIED HAVE BEEN TESTED, THE MACHINE WILL AGAIN HALT AT LOCATION 75320.
 - F. IF YOU WISH TO TEST ADDITIONAL DRIVES, YOU MAY SET THE APPROPRIATE ENTRY KEYS AND DEPRESS START AT THIS TIME. IF YOU WISH TO TEST THE SAME UNITS AGAIN AND AGAIN, YOU MAY PLACE SSW 5 DOWN AND DEPRESS START. THE PROGRAM WILL NOT STOP AT 75320 AS LONG AS SSW 5 IS DOWN. THIS IS A SPECIAL USAGE OF THIS SWITCH IN THIS DIAGNOSTIC.
 - G. OBSERVE THE OPERATION OF A TAPE DRIVE WHILE IT IS BEING TESTED BY 9T51.

4. FORCE LOAD 9M56 FROM THE CARD READER. A WORD COUNT OF 40000 WILL BE MORE THAN SUFFICIENT. THE CHANNEL ADDRESS COUNTER MUST BE SET TO THE STARTING LOCATION OF 9M56.
 - A. WHEN ALL OF THE CARDS HAVE BEEN READ, RETURN THE CHANNEL TO AUTOMATIC STATUS.
 - B. DEPRESS RESET. THIS SETS THE INSTRUCTION COUNTER TO 00000 WHICH IS WHERE THE PROGRAM STARTS.
 - C. DEPRESS START. NOTE THE HALT AT 00000.
 - D. DEPRESS START. NOTE THE HALT AT 00001.
 - E. DEPRESS START. THE PROGRAM WILL NOW RUN CONTINUOUSLY IF SSW 6 IS DOWN.
 - F. OBSERVE ALL PRINT OUTS.
 - G. NOTE THE APPEARANCE OF THE 7151 WHILE THE DIAGNOSTIC IS BEING PERFORMED.
 - H. PLACE SSW 6 UP. THIS IS A SPECIAL USAGE OF THIS SWITCH WHICH IS UNIQUE TO THIS DIAGNOSTIC. THE CARD READER WAS NOT SELECTED. NOTICE THAT INSTEAD THE MACHINE HALTED WITH MOST OF THE CONSOLE LIGHTS ON. THIS OCCURS AT THE END OF THE TEST, SO ANY LIGHTS WHICH SHOULD BE ON BUT ARE NOT ON ARE EITHER BAD THEMSELVES OR HAVE BAD DRIVERS. THE REGISTER POSITIONS HAVE ALREADY BEEN TESTED AND FOUND TO BE FUNCTIONING PROPERLY.

- I. DISPLAY LOCATION 00001. THIS IS THE SECOND HALT AND PROCEED INSTRUCTION. NOTE THE OP CODE. ALSO NOTE THE CONTENTS OF THE ADDRESS FIELD. A HPR INSTRUCTION REQUIRES NO INFORMATION IN THE ADDRESS FIELD.
 - J. DISPLAY LOCATION 00000. THIS IS THE FIRST HPR. NOTICE THAT IT HAS AN ADDRESS PORTION OF THREE. THIS WILL HAVE NO EFFECT ON THE OPERATION OF THIS WORD AS A CPU INSTRUCTION. HOWEVER, CONSIDER THE OPERATION OF THE LOAD BUTTON SEQUENCE FOR A MOMENT. IF YOU WERE TO PLACE THIS DECK (WHICH HAS NO LOADER) IN THE CARD READER AND DEPRESS THE LOAD CARDS KEY, 00000, 00001, AND 00002 WOULD BE LOADED. CPU WOULD PROCEED TO LOCATION 00001, FIND AN HPR, AND HALT. CHANNEL WOULD BRING OUT THE CONTENTS OF LOCATION 00000 AND INTERPRET IT AS A COMMAND. THIS COMMAND WOULD BE AN IOCD, WORD COUNT OF 42000, ADDRESS OF 00003. THIS WOULD READ IN THE BALANCE OF THE DECK. THUS, 9M56 COULD BE LOADED BY USE OF THE LOAD CARDS BUTTON, STILL USING NONE OF THE ARITHMETIC OR LOGIC FUNCTIONS OF CPU. THIS IS A SPECIAL FEATURE OF 9M56 ONLY.
- 5. YOU WILL NOW GENERATE A FORCE LOAD TAPE. THIS MAY BE DONE ON EITHER CHANNEL. THE FIRST FILE ON THIS TAPE WILL BE 9M56.
 - A. USING EITHER DATA CHANNEL CONSOLE, WRITE 9M56 ON TAPE FROM CORE. USE AN IOCD WITH A WORD COUNT OF 77776. WRITE AN END OF FILE MARK.
 - B. REWIND THE TAPE AND CLEAR STORAGE.
 - C. USING AN IOCD WITH A FULL WORD COUNT (77777), READ 9M56 INTO STORAGE.
 - D. RESET AND START. THE PROGRAM SHOULD RUN IN THE NORMAL WAY.
 - 6. YOU WILL NOW ADD ANOTHER DIAGNOSTIC TO THE SAME FORCE LOAD TAPE.
 - A. LOAD 9T51 INTO STORAGE FROM THE STANDARD DIAGNOSTIC TAPE.
 - B. 9T51 DOES NOT START AT LOCATION 00000. DURING A NORMAL LOAD OPERATION, THE LOAD PROGRAM SEARCHES FOR A TRANSFER CARD OR A TRANSFER RECORD. THIS TELLS THE LOADER WHERE THE MAIN PROGRAM STARTS. THE CONTENTS OF THE TRA CARD IS THE RESULT OF THE SYMBOLIC ADDRESS SHOWN ON THE END CARD WHICH WAS INCLUDED IN THE SYMBOLIC DECK WHEN THE PROGRAM WAS ASSEMBLED. LOOK IN THE LISTING OF 9T51 TO DETERMINE THE ADDRESS ASSIGNED TO THE END STATEMENT. WE WILL CALL THIS THE STARTING LOCATION.
 - C. IF YOU WERE TO WRITE CORE ON TAPE NOW, YOU COULD FORCE LOAD IT AT ANY FUTURE TIME, PERFORM A MANUAL TRANSFER TO THE STARTING LOCATION, AND THE PROGRAM WOULD RUN IN THE NORMAL MANNER. DO NOT WRITE TAPE YET.

- D. IF YOU STORE A TRA TO THE STARTING LOCATION AT STORAGE LOCATION 00000 BEFORE WRITING CORE ON TAPE, THERE WILL BE NO NEED FOR A MANUAL TRANSFER AND NO NEED TO REMEMBER THE TRANSFER-TO LOCATION WHEN THE PROGRAM IS USED. THIS CAN BE IMPORTANT WHEN THERE ARE SEVERAL DIFFERENT PROGRAMS ON THE FORCE LOAD TAPE AND THEY ALL START AT DIFFERENT LOCATIONS.
- E. STORE SUCH A TRANSFER AT LOCATION 00000.
- F. WRITE CORE ON TAPE WITH AN IOCD WITH A WORD COUNT OF 77776.
- G. WRITE AN END OF FILE.
- H. REWIND THE TAPE AND CLEAR STORAGE.
- I. FORCE LOAD 9T51. USE AN IOCD AGAIN BUT THIS TIME USE A FULL WORD COUNT. ON THE FIRST READ SELECT 9M56 (77776 WORDS) WILL BE READ INTO STORAGE. A WORD COUNT OF ONE WILL REMAIN IN THE CHANNEL ADDRESS COUNTER, SO THE CHANNEL AND THE TAPE WILL CONTINUE IN OPERATION UNTIL THE EOF IS RECOGNIZED. RE-LOAD THE COMMAND AND READ SELECT ONCE AGAIN. 9T51 WILL NOW BE FORCE LOADED INTO CORE. TEST IT TO SEE THAT IT WILL WORK PROPERLY.

LABORATORY PROJECT 8

OIL CORE SCOPING

SET UP C (KEYS) = STO 70
 C (AC) = 377777777777
 C (PC) = 0
 C (O) = 0
 CONTINUOUS ENTER INSTRUCTION SWITCH ON

SCOPE SYNC ANY OUTPUT OF BLOCK 1H ON PAGE 01.12.00.1
 INCLUDING FOOTNOTES 4 AND 5

I. OBSERVE ALL INPUTS AND OUTPUTS OF THE FOLLOWING

- A. X DRIVER PAGE 01.09.00.1 BLK 1B
 PAGE 01.09.00.1 BLK 2B
 PAGE 01.09.01.1 BLK 1D
- B. Y DRIVER PAGE 01.10.00.1 BLK 1E
- C. Z DRIVER PAGE 01.11.06.2 BLK 5D
- D. TIMING CIRCUITS PAGE 00.01.01.0

II. TEST POINTS

- A. CHECK THE FIRST ELEVEN WAVE FORMS ON THE
 TIMING CHART AT THE LOCATIONS GIVEN AS TEST
 POINTS. LOCATE EACH TEST POINT IN SYSTEM
 PAGES.

III. STROBE GENERATOR

- A. 01.16.00.1 5H
- B. 01.16.00.1 4H
- C. 01.16.00.1 4I
- D. 01.16.00.1 2A
- E. 01.16.00.1 1A
- F. 01.16.00.1 1I

IV. SENSE AMPS.

- A. USE TYPE CA PLUG-IN UNIT ON THE SCOPE.
SET WITH INPUTS ADDED ALGEBRAICALLY TO
VIEW SEGMENT INPUTS.
- B. COMPARE STROB TO INPUT AT SA TEST POINT
(PIN H).
THE TEST POINT IS THE OUTPUT OF THE
EMITTER FOLLOWER.

V. MAR

- A. 01.02.00.1 4B

VI. MDR

- A. 01.18.03.1 3A

LABORATORY PROJECT 9
7302A FAMILIARIZATION
(AIR COOLED CORE STORAGE)

1. POWER SUPPLY CONTROL PANEL

A. TELL WHICH VOLTAGES CAN BE VARIED WITH THE RAISE-LOWER SWITCH.

B. CAN THESE VOLTAGES ALSO BE VARIED FROM THE CONSOLE.

C. TELL THE PURPOSE OF THE J1 AND J2 JACKS.

D. TELL THE PURPOSE OF THE NORMAL-LOGIC SWITCH.

2. POWER SUPPLIES

A. HOW MANY POWER SUPPLIES ARE THERE.

B. TELL THE POLARITY AND VOLTAGE OF EACH, FROM TOP TO BOTTOM.

C. ARE THESE POWERED BY THE MG SET.

D. TELL HOW MANY CIRCUIT BREAKERS ARE IN EACH SUPPLY.

E. TELL WHY SOME SUPPLIES HAVE ADJUSTING POTENTIOMETERS WHILE OTHERS DO NOT.

F. TELL HOW YOU WOULD REMOVE A POWER SUPPLY FROM THE MACHINE.

3. TAILGATES

A. TELL HOW MANY TAILGATES THERE ARE.

B. TELL HOW THEY ARE LABELED.

C. HOW MANY SIGNAL CABLES ARE REQUIRED.

D. DESCRIBE THE PHYSICAL LOCATION OF PIN NUMBER 01G01B17.

4. PHYSICALLY LOCATE AND OBSERVE EACH OF THE FOLLOWING.

A. SENSE AMPLIFIERS

B. MDR TRIGGERS (DR).

C. Z DRIVERS

D. MAR TRIGGERS

E. X AND Y DRIVERS

F. LOAD SHARING SWITCHES (HOW MANY ARE THERE.)

G. SEGMENT DRIVERS

H. CONTROL CIRCUITS

5. OBSERVE THE OPERATION OF THE CF TEST PANEL.

6. IN RELATION TO THE CORE ARRAY, LOCATE AND OBSERVE EACH OF THE FOLLOWING-

A. HEATING ELEMENTS

B. FANS

C. THERMOSTATS

D. THERMOMETER

E. DUMMY PLANES

F. Z DRIVER TERMINATING RESISTORS

G. X WINDING TERMINATING RESISTORS

H. Y WINDING TERMINATING RESISTORS

I. INTER-PLANE JUMPER CONNECTIONS
(FOLLOW SKEW THRU SEVERAL PLANES)

J. IS IT POSSIBLE TO REMOVE A SINGLE PLANE- _____

K. WHAT ARE INTER-PLANE AIR BAFFLES- _____

WHY ARE THEY NEEDED- _____

L. X PRIMARY RESISTORS

M. Y PRIMARY RESISTORS

N. RESISTOR COOLING FANS

O. AIR FLOW SENSING SWITCHES

7. OBSERVE THE WAVE FORMS WHICH ARE DEPICTED ON PAGES 6 THRU 10 OF THE REFERENCE MANUAL.

7090 SYSTEM PROJECT 10
PRELIMINARY INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

SLIGHT MOTION OF THE DIAGNOSTIC TAPE WAS OBSERVED. THE MACHINE HUNG UP AS SHOWN ON PICK-SHEET 1. EXAMINE THE PICK-SHEET.

9M51 IS THE MAINFRAME DIAGNOSTIC AND TESTS EVERY CPU INSTRUCTION. HOWEVER, BECAUSE THE MOTION OF THE DIAGNOSTIC TAPE WAS NOT NORMAL, IT IS POSSIBLE THAT THE LOAD PROGRAM FAILED TO OPERATE PROPERLY. IT WAS THEREFORE DECIDED THAT IT WOULD BE A WISE MOVE TO TRY TO ELIMINATE THE LOAD PROGRAM AND ASSOCIATED CIRCUITRY BY USING 9M56. FOR A DESCRIPTION OF 9M56 SEE THE WRITE UP FOR 9M51.

FORCE LOADED 9M56 FROM TAPE B2.

APPEARED TO LOAD IN NORMAL MANNER.
DEPRESSED RESET KEY.

DEPRESSED START KEY.
PROGRAM STOP AT LOCATION 00001

DEPRESSED START KEY.
PROGRAM STOP AT LOCATION 00002

DEPRESSED START KEY.
MACHINE HUNG UP. SEE PICK-SHEET 2.

DETERMINE THE REASON FOR THE PROGRAM STOPS AT LOCATIONS 00001 AND 00002.

IN THE CASE OF A HANG UP SUCH AS IS SHOWN ON PICK-SHEET 2, THE PROGRAM COUNTER SHOWS THE LOCATION OF THE INSTRUCTION BEING EXECUTED PLUS ONE. THE MACHINE IS IN L TIME. DURING THE PREVIOUS I CYCLE THE INSTRUCTION NOW BEING EXECUTED WAS BROUGHT FROM CORE STORAGE AND THE PROGRAM COUNTER WAS STEPPED ONE TIME IN PREPARATION FOR THE NEXT I TIME. TURN TO THE APPROPRIATE LOCATION IN THE LISTING OF 9M51/56 AND ANALYZE THE TEST ROUTINE IN ORDER TO DETERMINE THE NATURE OF THE FAILURE.

7090 SYSTEM PROJECT 10
PRELIMINARY INSTRUCTIONS

PICK-SHEET NUMBER 4

PROGRAM COUNTER		000 000 000 000	●●●																	
ADDRESS REGISTER		000 000 000 000	●●●																	
INDEX REGISTER A		000 000 000 000	000																	
INDEX REGISTER B		000 000 000 000	000																	
INDEX REGISTER C		000 000 000 000	00●																	
PROGRAM REGISTER	O S	●●● ●●● 00●		SHIFT COUNTER		00 000 000														
STORAGE REG.	O S	00 ●●● ●●● 00●	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000
ACCUMULATOR	O O O S Q P	00 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000
M Q CONTENTS	O S	00 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000	000 000 000 000 000 000

O I/O CHECK	O FP	SENSE	CYCLE TIME
		1 2 3 4	I L E B M
O TRAP	O 9 CARRY	0 0 0 0	0 ● 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
		1 2 3 4 5	A B C D E F G H
O ACC. OVERFLOW	O Q CARRY	● 0 0 0 0	0 0 0 0 0 0 0 0
O QUOT OVERFLOW	O MST STOP	COMMAND TRAP	TAPE CHECK TRAP
		A B C D E F G H	A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK	
		A B C D E F G H	
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0	
O T2	O X CARRY		

7090 SYSTEM PROJECT 10
 PRELIMINARY INSTRUCTIONS

PICK-SHEET NUMBER 2

PROGRAM COUNTER		000 ●00 00● 000 0●●	
ADDRESS REGISTER		000 000 000 000 000	
INDEX REGISTER A		000 ●00 00● 000 000	
INDEX REGISTER B		000 0●0 000 000 000	
INDEX REGISTER C		000 000 000 000 000	
PROGRAM REGISTER	●	●●● ●00 0●●	SHIFT COUNTER 00 000 000
	S		
STORAGE REG.	●	00 ●●● ●00 0●● 000 000 000 000 000 000 000 000	
	S	-----	-----
ACCUMULATOR	O O O	00 000 000 000 000 000 000 000 000 000 000 000	
	S Q P	-----	-----
M Q CONTENTS	O	00 000 000 000 000 000 000 000 000 000 000 00●	
	S	-----	-----

O I/O CHECK	O FP	SENSE	CYCLE TIME
		1 2 3 4	I L E B M
O TRAP	O 9 CARRY	0 0 0 0	0 ● 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
		1 2 3 4 5	A B C D E F G H
O ACC. OVERFLOW	O Q CARRY	● 0 0 0 0	0 0 0 0 0 0 0 0
O QUOT OVERFLOW	O MST STOP	COMMAND TRAP	TAPE CHECK TRAP
		A B C D E F G H	A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK	
		A B C D E F G H	
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0	
O T2	O X CARRY		

7090 SYSTEM PROJECT 11
PRELIMINARY INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
PROGRAM STOP OCCURED AT LOCATION 00001.

DEPRESSED START KEY.
PROGRAM STOP OCCURED AT LOCATION 00002.

DEPRESSED START KEY.
PROGRAM STOP OCCURED. SEE PICK-SHEET 1.

DEPRESSED START KEY.
PROGRAM STOP OCCURED. SEE PICK-SHEET 2.

BECAUSE EACH 7090 CPU INSTRUCTION IS TESTED BY 9M51 BEFORE IT IS USED BY 9M51 TO TEST SOME OTHER INSTRUCTION, IT IS ALMOST ALWAYS NECESSARY TO TRACE OUT THE TROUBLE ON THE BASIS OF THE FIRST ERROR INDICATION. TO USE AN ERROR INDICATION OTHER THAN THE FIRST, IN THE ACTUAL TROUBLE-SHOOTING, CAN LEAD TO NEEDLESS CONFUSION, DIFFICULTY, AND DELAY. HOWEVER, IT IS FREQUENTLY HELPFUL TO EXAMINE ONE OR MORE OF THE LATER ERROR INDICATIONS IN ORDER TO GAIN A BETTER UNDERSTANDING OF THE FIRST ERROR INDICATION. THE MORE INFORMATION YOU CAN GATHER ABOUT THE ERROR THE BETTER YOUR ANALYSIS WILL BE.

7090 SYSTEM PROJECT 11
PRELIMINARY INSTRUCTIONS

PICK-SHEET NUMBER 1

PROGRAM COUNTER		000 000 000 ●●● ●00	
ADDRESS REGISTER		000 000 000 000 000	
INDEX REGISTER A		000 000 000 000 000	
INDEX REGISTER B		000 000 000 000 000	
INDEX REGISTER C		000 000 000 000 000	
PROGRAM REGISTER	O	●00 000 000	SHIFT COUNTER 00 000 000
	S		
STORAGE REG.	O	00 ●00 000 000 000 000 000 000 000 000 000 000	
	S	-----	-----
ACCUMULATOR	O O O	00 000 000 000 000 000 000 000 000 000 000 000	
	S Q P	-----	-----
M Q CONTENTS	O	00 00● 000 000 000 000 000 000 000 000 000 000	
	S	-----	-----

O I/O CHECK	O FP	SENSE	CYCLE TIME
		1 2 3 4	I L E B M
O TRAP	O 9 CARRY	0 0 0 0	0 ● 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
		1 2 3 4 5	A B C D E F G H
● ACC. OVERFLOW	O Q CARRY	● 0 0 0 0	0 0 0 0 0 0 0 0
O QUOT OVERFLOW	● MST STOP	COMMAND TRAP	TAPE CHECK TRAP
		A B C D E F G H	A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK	
		A B C D E F G H	
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0	
O T2	O X CARRY		

7090 SYSTEM PROJECT 11
PRELIMINARY INSTRUCTIONS

PICK-SHEET NUMBER 2

PROGRAM COUNTER		000	000	000	000	000							
ADDRESS REGISTER		000	000	000	000	000							
INDEX REGISTER A		000	000	000	000	000							
INDEX REGISTER B		000	000	000	000	000							
INDEX REGISTER C		000	000	000	000	000							
PROGRAM REGISTER	O	000	000	000			SHIFT COUNTER		00	000	000		
	S												
STORAGE REG.	O	00	000	000	000	000	000	000	000	000	000	000	000
	S												
ACCUMULATOR	O O O	00	000	000	000	000	000	000	000	000	000	000	000
	S Q P												
M Q CONTENTS	O	00	000	000	000	000	000	000	000	000	000	000	000
	S												

O I/O CHECK	O FP	SENSE		CYCLE TIME
		1 2 3 4		I L E B M
O TRAP	O 9 CARRY	0 0 0 0		0 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER		CHANNEL SELECT
		1 2 3 4 5		A B C D E F G H
● ACC. OVERFLOW	O Q CARRY	● 0 0 0 0		0 0 0 0 0 0 0 0
O QUOT OVERFLOW	● MST STOP	COMMAND TRAP		TAPE CHECK TRAP
		A B C D E F G H		A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0		0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK		
		A B C D E F G H		
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0		
O T2	O X CARRY			

7090 SYSTEM PROJECT 12
PRELIMINARY INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
PROGRAM STOP OCCURED AT LOCATION 00001.

DEPRESSED START KEY.
PROGRAM STOP OCCURED AT LOCATION 00002.

DEPRESSED START KEY.
PRINT-OUT A OCCURED.

FURTHER PRINT-OUTS SUPPRESSED BY SSW 2.

EXAMINE THE PRINT-OUT AND NOTE EACH OF THE FOLLOWING ITEMS--

FIRST LINE--

TEST LOCATION - TELLS YOU THE LOCATION OF THE FIRST INSTRUCTION OF THE
ROUTINE THAT FAILED. (24740)

OPERATION - TELLS YOU THE INSTRUCTION THE ROUTINE WAS DESIGNED TO TEST.
(STP)

ERROR LOCATION - TELLS YOU THE LOCATION WITHIN THE ROUTINE AT WHICH THE
ERROR WAS RECOGNIZED. SINCE SOME ROUTINES CHECK FOR SEVERAL KINDS
OF FAILURE, THEY HAVE SEVERAL POSSIBLE ERROR LOCATIONS. THIS
PORTION OF THE PRINT-OUT TELLS YOU THE ERROR LOCATION AT WHICH THIS
ERROR WAS DETECTED. (24744)

0 LOCATION - TELLS YOU THE CONTENTS OF LOCATION 00000 AT THE TIME THE
ERROR WAS DETECTED. THIS IS USEFUL IN THE CASE OF PROBLEMS IN-
VOLVING TRAP.

SWITCHES - TELLS YOU THE SETTING OF THE SENSE SWITCHES AT THE TIME THE
ERROR WAS DETECTED.

EXAMINE THE SECOND AND THIRD LINES OF THE ERROR PRINT-OUT. IF THERE IS
ANY PORTION OF THESE THAT YOU DO NOT UNDERSTAND, ASK YOUR INSTRUCTOR.

7090 SYSTEM PROJECT 12

PRELIMINARY INSTRUCTIONS

***** PRINT OUT A *****

9M51 NOW TESTING BASIC MAIN FRAME INSTRUCTIONS

TEST LOC 24740, OPN STP ,ERROR LOC 24744, O LOC 042000000003,SW 000001
LITE 0000, MQ 377777777777, XRA 24700, XRB 77401, XRC 53034, TRAP TGR 0
ACC +,Q 0,P 1,300000000000, DIV CK 0, ACC OVFL 1, KEYS 000000000001

TEST LOC 42165, OPN STP ,ERROR LOC 42201, O LOC 042000000003,SW 000001
LITE 0000, MQ 777777777400, XRA 77777, XRB 77777, XRC 35577, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 1, ACC OVFL 1, KEYS 000000000001

TEST LOC 42254, OPN STP ,ERROR LOC 42274, O LOC 042000000003,SW 000001
LITE 0000, MQ 000000000000, XRA 77777, XRB 34214, XRC 35504, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 1, ACC OVFL 1, KEYS 000000000001

TEST LOC 47335, OPN STP ,ERROR LOC 47355, O LOC 042000000003,SW 000001
LITE 0000, MQ 600000000000, XRA 00004, XRB 07777, XRC 30423, TRAP TGR 0
ACC +,Q 0,P 0,076100051400, DIV CK 0, ACC OVFL 0, KEYS 000000000001

TEST LOC 50664, OPN 7/0000,ERROR LOC 50676, O LOC 002000050544,SW 000001
LITE 0000, MQ 377777777777, XRA 00004, XRB 34214, XRC 27102, TRAP TGR 0
ACC +,Q 0,P 0,300000000000, DIV CK 0, ACC OVFL 1, KEYS 000000000001

7090 SYSTEM PROJECT 13

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9T51 FROM TAPE A1

APPEARED TO LOAD IN NORMAL MANNER.
PRINT OUT A OCCURED

EXAMINE THIS PRINT OUT. CONSULT THE
INSTRUCTION PORTION OF THE 9T51 DIAG-
NOSTIC WRITE UP AND DETERMINE ITS
SIGNIFICANCE.

PROGRAM STOP OCCURED
SEE PICK-SHEET 1.

CONSULT THE 9T51 DIAGNOSTIC WRITE UP
IN ORDER TO DETERMINE THE PURPOSE OF
THIS HALT. ALSO SEE THE WRITE UP FOR
9IOC WHICH IS AVAILABLE IN THE LAB.

THE OP PANEL KEYS ON THE CPU CONSOLE
WERE SET FOR 9IOC AS FOLLOWS-
KEYS 19, 20, 22, 24, 28, 31 DOWN.

DEPRESSED START KEY.
PRINT-OUT^B OCCURED.
FURTHER PRINT-OUTS SUPPRESSED BY SSW 2.

A PROGRAM STOP OCCURED ONCE MORE AS IS
SHOWN ON PICK-SHEET 1. THIS INDICATES
THAT THE TEST HAS BEEN COMPLETED ON THE
UNITS SPECIFIED AND THAT THE PROGRAM
IS NOW READY TO PERFORM FURTHER TESTS
BASED ON NEW INFORMATION PLACED IN
THE OP PANEL KEYS.

7090 SYSTEM PROJECT 13

DATA CHANNFL

***** PRINT OUT A *****

NOW PERFORMING DIAGNOSTIC TEST 9T51

WRITE -AT-LOAD-POINT-DELAY-**-LOW TOLERANCE- -HI-TOLERANCE-

FOR - 729 MODS -II- AND -V-	46.56 MILSEC	53.96 MILSEC
FOR - 729 MODS -IV- AND -VI-	31.04 MILSEC	37.64 MILSEC

-READ-AT-LOAD-POINT-DELAY-**-

FOR - 729 MODS -II- AND -V-	23.28 MILSEC	29.48 MILSEC
FOR - 729 MODS -IV- AND -VI-	15.52 MILSEC	21.32 MILSEC

7090 SYSTEM PROJECT 13

DATA CHANNEL

PICK-SHEET NUMBER 4

PROGRAM COUNTER		●●● ●●● ●●● ●●● ●●●	
ADDRESS REGISTER		●●● ●●● ●●● ●●● ●●●	
INDEX REGISTER A		●●● ●●● ●●● ●●● ●●●	
INDEX REGISTER B		●●● ●●● ●●● ●●● ●●●	
INDEX REGISTER C		●●● ●●● ●●● ●●● ●●●	
PROGRAM REGISTER	O S	●●● ●●● ●●● ●●●	SHIFT COUNTER ●●● ●●● ●●●
STORAGE REG.	O S	●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●●	
ACCUMULATOR	O O O S Q P	●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●●	
M Q CONTENTS	O S	●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●●	

O I/O CHECK	O FP	SENSE	CYCLE TIME
O TRAP	O 9 CARRY	1 2 3 4	I L E B M
O SIMULATE	O 9 OVERFLOW	0 0 0 0	0 ● 0 0 0
O ACC. OVERFLOW	O Q CARRY	TALLY COUNTER	CHANNEL SELECT
O QUOT OVERFLOW	● MST STOP	1 2 3 4 5	A B C D E F G H
O R/W SELECT	O END OP	● 0 0 0 0	0 ● 0 0 0 0 0 0
O DIVIDE CHECK	O AND	COMMAND TRAP	TAPE CHECK TRAP
O TRAP CONTROL	O CAQ	A B C D E F G H	A B C D E F G H
O T2	O X CARRY	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
		CHANNEL TAPE CHECK	
		A B C D E F G H	
		0 0 0 0 0 0 0 0	

7090 SYSTEM PROJECT 13

DATA CHANNEL

***** PRINT OUT A *****

9T51 PASS COMPLETE CHN A TF 2 MOD 2

TEST LOC 00472, OPN SCHB 5,ERROR LOC 00526, 0 LOC 002000013366,SW 000001
LITE 0000, MQ 77777277777, XRA 00001, XRB 00001, XRC 77252, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 1,
INDS 000100001154 KEYS 000000324223

TEST LOC 00472, OPN SCHB 5,ERROR LOC 01276, 0 LOC 002000013366,SW 000001
LITE 0000, MQ 77777277777, XRA 00001, XRB 00001, XRC 76502, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 1,
INDS 000100001154 KEYS 000000324223

TEST LOC 00472, OPN SCHB 5,ERROR LOC 01324, 0 LOC 002000013366,SW 000001
LITE 0000, MQ 77777277777, XRA 00001, XRB 00001, XRC 76454, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 1,
INDS 000100001154 KEYS 000000324223

TEST LOC 00472, OPN SCHB 5,ERROR LOC 01374, 0 LOC 002000013366,SW 000001
LITE 0000, MQ 77777277777, XRA 00001, XRB 00001, XRC 76404, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 1,
INDS 000100001154 KEYS 000000324223

TEST LOC 00472, OPN SCHB 5,ERROR LOC 01432, 0 LOC 002000013366,SW 000001
LITE 0000, MQ 77777277777, XRA 00001, XRB 00001, XRC 76346, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 1,
INDS 000100001154 KEYS 000000324223

TEST LOC 01466, OPN TRCB 5,ERROR LOC 01503, 0 LOC 002000013366,SW 000001

TEST LOC 01714, OPN BSRB 5,ERROR LOC 01747, 0 LOC 002000013366,SW 000001
MSE 0000, COMP ERROR WORD GENERATED 00001000001000001000001000001000001
REC 00001, WORD 00001, WORD READ 00
INDS 000100001154 KEYS 000000324223 1 1 1 1 1 1

TEST LOC 01753, OPN BSFB 5,ERROR LOC 02003, 0 LOC 002000013366,SW 000001
MSE 0000, COMP ERROR WORD GENERATED 000001000001000001000001000001000001
REC 00001, WORD 00001, WORD READ 00
INDS 000100001154 KEYS 000000324223 1 1 1 1 1 1

7090 SYSTEM PROJECT 14

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

ALLED 9T51 FROM TAPE A1

APPEARED TO LOAD IN NORMAL MANNER.
PRINT-OUT A OCCURED.
NORMAL STOP FOR 9I/OC OCCURED AT 75320.

ENTERED FOLLOWING 9I/OC INFORMATION-
KEYS 19, 20, 24 DOWN.

DEPRESSED START KEY.
TAPE DRIVE UNDER TEST RAN AWAY.
SEE PICK SHEETS 1 AND 2.

7090 SYSTEM PROJECT 14

DATA CHANNEL

PICK-SHEET NUMBER 1

PROGRAM COUNTER		000 000 000 000 000	
ADDRESS REGISTER		000 000 000 000 000	
INDEX REGISTER A		000 000 000 000 000	
INDEX REGISTER B		000 000 000 000 000	
INDEX REGISTER C		000 000 000 000 000	
PROGRAM REGISTER	O S	000 000 000	SHIFT COUNTER 00 000 000
STORAGE REG.	O S	00 000 000 000 000 000 000 000 000 000 000 000	
ACCUMULATOR	O O O S Q P	00 000 000 000 000 000 000 000 000 000 000 000	
M Q CONTENTS	O S	00 000 000 000 000 000 000 000 000 000 000 000	

O I/O CHECK	O FP	SENSE	CYCLE TIME
		1 2 3 4	I L E B M
O TRAP	O 9 CARRY	0 0 0 0	0 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
		1 2 3 4 5	A B C D E F G H
O ACC. OVERFLOW	O Q CARRY	0 0 0 0 0	0 0 0 0 0 0 0 0
O QUOT OVERFLOW	O MST STOP	COMMAND TRAP	TAPE CHECK TRAP
		A B C D E F G H	A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK	
		A B C D E F G H	
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0	
O T2	O X CARRY		

7090 SYSTEM PROJECT 14

DATA CHANNEL

***** PRINT OUT A *****

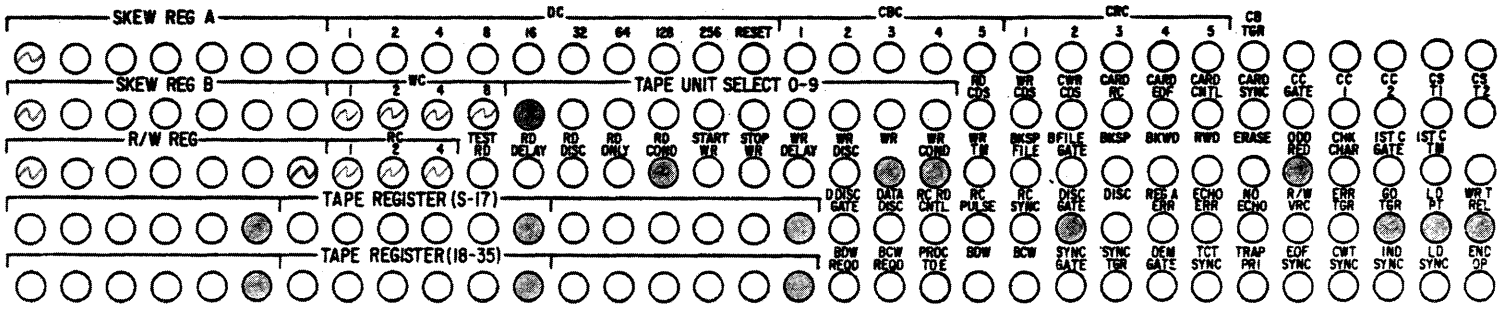
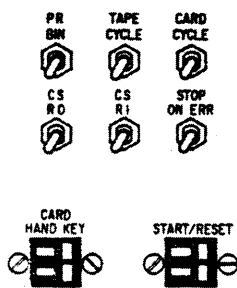
LOW PERFORMING DIAGNOSTIC TEST 9T51

WRITE -AT-LOAD-POINT-DELAY-**-LOW TOLERANCE- -HI-TOLERANCE-

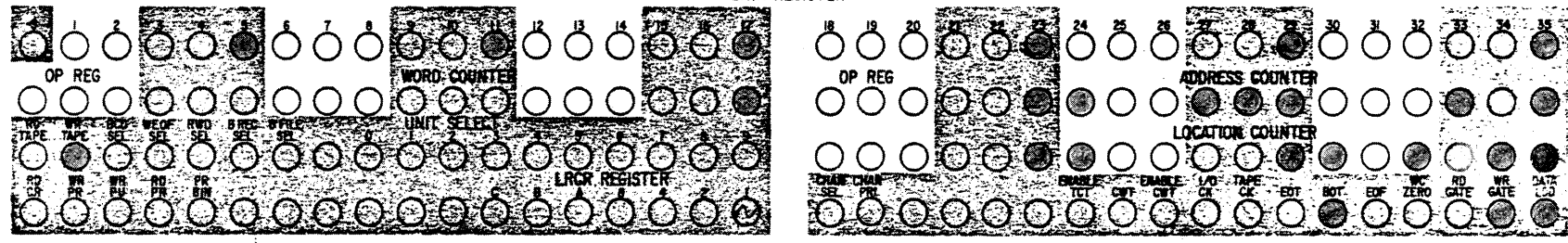
FOR - 729 MODS -II- AND -V-	46.56 MILSEC	53.96 MILSEC
FOR - 729 MODS -IV- AND -VI-	31.04 MILSEC	37.64 MILSEC

READ-AT-LOAD-POINT-DELAY-**-

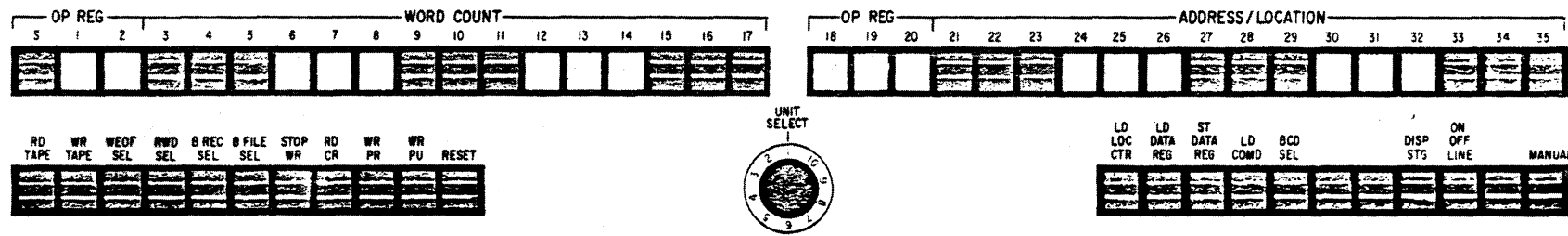
FOR - 729 MODS -II- AND -V-	23.28 MILSEC	29.48 MILSEC
FOR - 729 MODS -IV- AND -VI-	15.52 MILSEC	21.32 MILSEC



DATA REGISTER



401



7090 SYSTEM PROJECT 15

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9T51 FROM TAPE A1

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL PRINT OUTS OCCURED.
NORMAL STOP FOR 9I/OC OCCURED AT 75320.

ENTERED FOLLOWING 9I/OC INFORMATION-
KEYS 19, 20, 21 DOWN.

DEPRESSED START KEY.
MACHINE HUNG UP.
SEE PICK SHEETS 1 AND 2.

7090 SYSTEM PROJECT 15

DATA CHANNEL

PICK-SHEET NUMBER 1

PROGRAM COUNTER		000 000 000 000 000	
ADDRESS REGISTER		000 000 000 000 000	
INDEX REGISTER A		000 000 000 000 000	
INDEX REGISTER B		000 000 000 000 000	
INDEX REGISTER C		000 000 000 000 000	
PROGRAM REGISTER	O S	000 000 000	SHIFT COUNTER 00 000 000
STORAGE REG.	O S	00 000 000 000 000 000 000 000 000 000 000 000	
ACCUMULATOR	O O O S Q P	00 000 000 000 000 000 000 000 000 000 000 000	
M Q CONTENTS	O S	00 000 000 000 000 000 000 000 000 000 000 000	

O I/O CHECK	O FP	SENSE	CYCLE TIME
		1 2 3 4	I L E B M
O TRAP	O 9 CARRY	0 0 0 0	0 0 0 0
O SIMULATE	O 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
		1 2 3 4 5	A B C D E F G H
O ACC. OVERFLOW	O Q CARRY	0 0 0 0 0	0 0 0 0 0 0 0 0
O QUOT OVERFLOW	O MST STOP	COMMAND TRAP	TAPE CHECK TRAP
		A B C D E F G H	A B C D E F G H
O R/W SELECT	O END OP	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
O DIVIDE CHECK	O AND	CHANNEL TAPE CHECK	
		A B C D E F G H	
O TRAP CONTROL	O CAQ	0 0 0 0 0 0 0 0	
O T2	O X CARRY		

7090 SYSTEM PROJECT 16

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOPS OCCURED AT 00001 AND 00002.

DEPRESSED START KEY.
PRINT-OUT A OCCURED.

PERFORMED A MANUAL TRANSFER TO LOCATION
52112 TO BYPASS LONG INDEX REG. TEST.
PRINT-OUT B OCCURED.

CALLED 9P51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOP FOR 9I/OC OCCURED AT 75320.
KEYS 19; AND 34 DOWN.

DEPRESSED START KEY.
PRINT-OUT C OCCURED.

7090 SYSTEM PROJECT 16

DATA CHANNEL

***** PRINT OUT A *****

9M51 NOW TESTING BASI IN FRAME INSTRUCTIONS

7090 SYSTEM PROJECT 16

DATA CHANNEL

***** PRINT OUT B *****

9M51 NOW TESTING SENS DICATOR INSTRUCTIONS

9M51 TESTING FLOATIN INT INSTRUCTIONS

7090 SYSTEM PROJECT 16

DATA CHANNEL

***** PRINT OUT C *****

PERFORMING PRINTER DIAGNOSTIC TEST P51-A ON CHANNEL A.
ION AA. PRINTER DISCONNECT TEST

TER DISCONNECT TEST COMPLETE.	
ION AB. CURSORY TEST COLUMNS 1-7	DER WPR.
56789012345678901234567890123456	12345678901234567890123456789012
ION AC. CURSORY TEST COLUMNS 73-	UNDER WPR.
78901234567890123456789012345678	34567890

7090 SYSTEM PROJECT 17

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

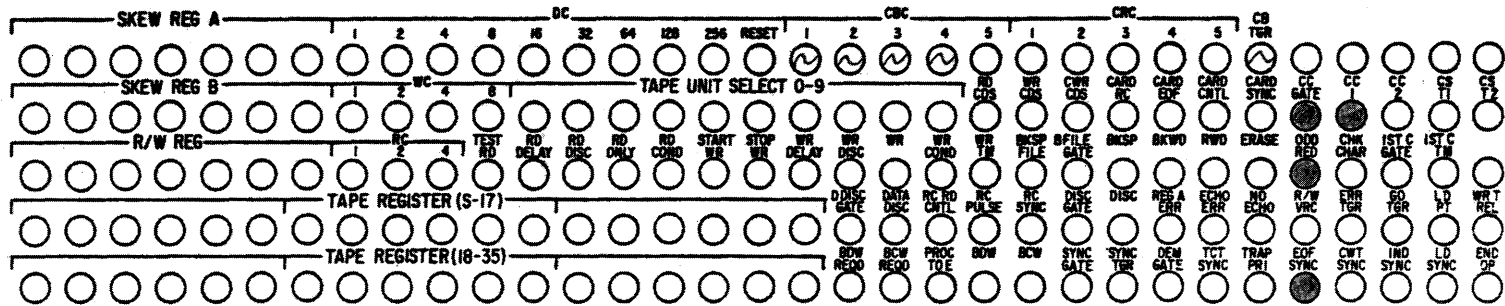
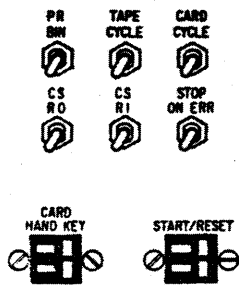
CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOPS OCCURED AT 00001 AND 00002.

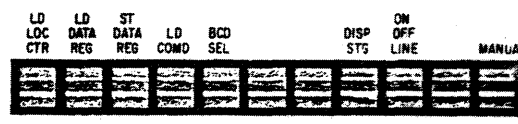
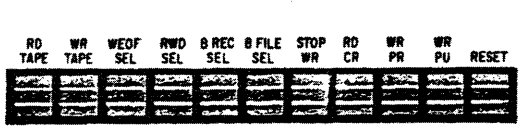
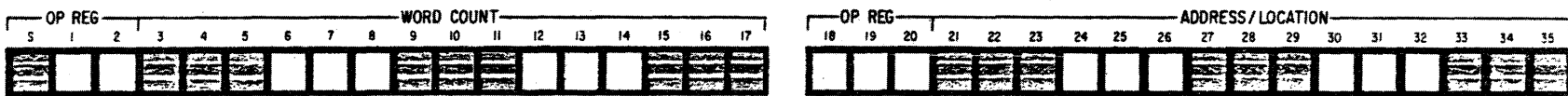
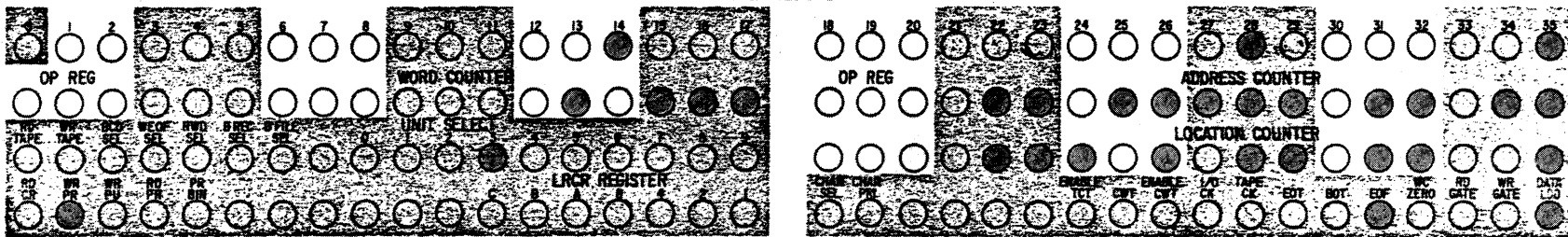
DEPRESSED START KEY.
PRINT-OUT A OCCURED.
SEE PICK SHEET 1.

CPU APPEARS TO BE EXECUTING LON INDEX
REGISTER TEST IN THE NORMAL MANNER.

DEPRESSED RESET KEY TO TERMINATE
PRINT OUT.



DATA REGISTER



410

SYSTEM PROJECT 17
Pick Sheet

7090 SYSTEM PROJECT 18

DATA CHANNEL

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M56 FROM CARD READER.

CARD READER TOOK ONE CARD FEED CYCLE.
MACHINE HUNG UP.
SEE PICK SHEETS 1 AND 2.

7290 SYSTEM PROJECT 18

DATA CHANNEL

PICK-SHEET NUMBER /

```

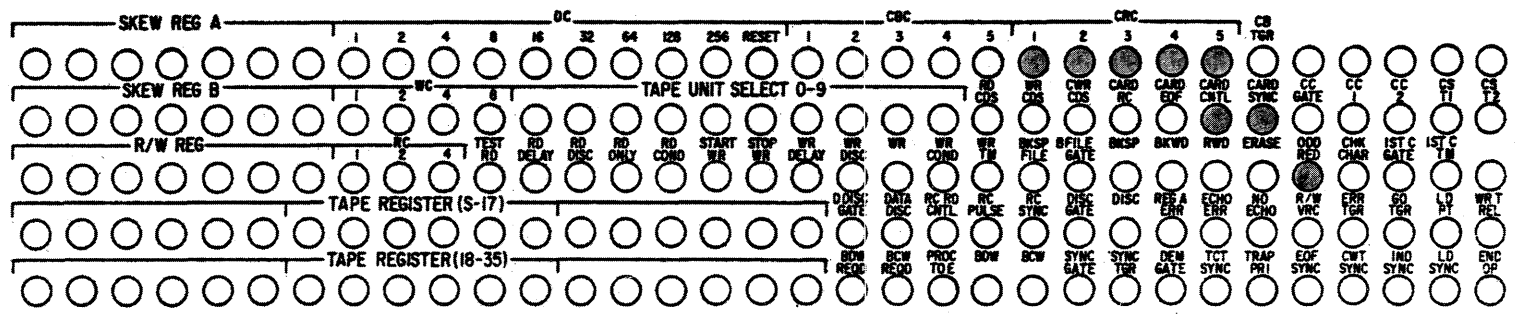
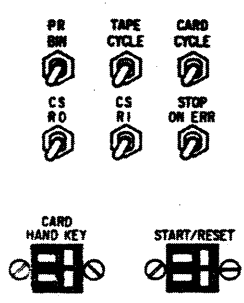
PROGRAM COUNTER      000 000 000 000 000
ADDRESS REGISTER    000 000 000 000 000
INDEX REGISTER A    000 000 000 000 000
INDEX REGISTER B    000 000 000 000 000
INDEX REGISTER C    000 000 000 000 000
PROGRAM REGISTER    0 000 000 000      SHIFT COUNTER    00 000 000
                   S
STORAGE REG.       0 00 000 000 000 000 000 000 000 000 000 000
                   S  -----
ACCUMULATOR       0 0 0 00 000 000 000 000 000 000 000 000 000
                   S Q P -----
M Q CONTENTS      0 00 000 000 000 000 000 000 000 000 000 000
                   S  -----

```

```

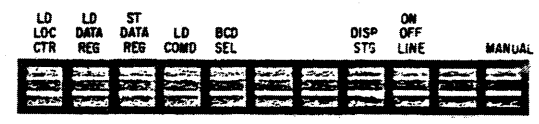
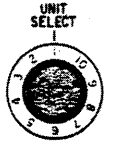
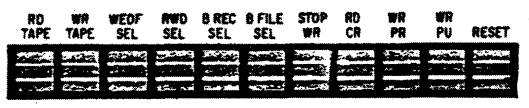
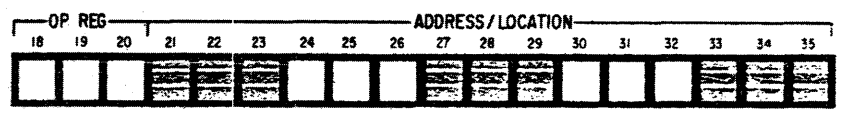
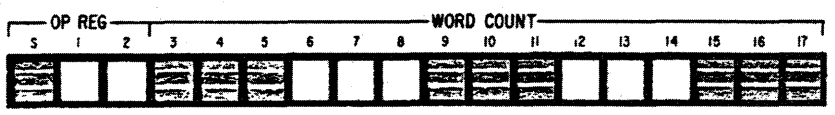
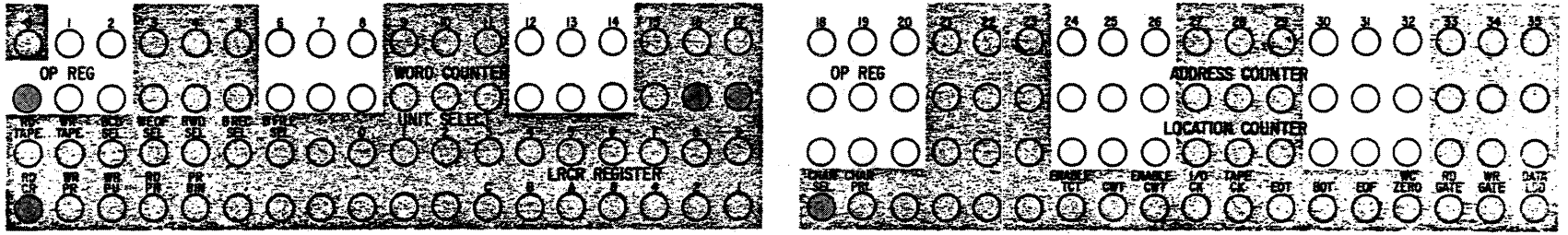
O I/O CHECK      O FP          SENSE          CYCLE TIME
O TRAP           O 9 CARRY     1 2 3 4    I L E B M
O SIMULATE       O 9 OVERFLOW  0 0 0 0    ● 0 0 0 0
O ACC. OVERFLOW O Q CARRY     TALLY COUNTER
O QUOT OVERFLOW ● MST STOP    1 2 3 4 5  CHANNEL SELECT
● R/W SELECT     ● END OP      0 0 0 0 0  A B C D E F G H
O DIVIDE CHECK   O AND        COMMAND TRAP  ● 0 0 0 0 0 0 0
O TRAP CONTROL  O CAQ         A B C D E F G H
O T2            O X CARRY     0 0 0 0 0 0 0  TAPE CHECK TRAP
                   CHANNEL TAPE CHECK
                   A B C D E F G H
                   0 0 0 0 0 0 0 0

```



414

DATA REGISTER



SYSTEM PROJECT 18
Pick Sheet 2

7090 SYSTEM PROJECT 19

ADVANCED INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOPS OCCURED AT 00001 AND 00002.

DEPRESSED START KEY.

PERFORMED A MANUAL TRANSFER TO LOCATION
52112 TO BYPASS LONG INDEX REG. TEST.

PRINT-OUT A OCCURED.
FURTHER PRINT-OUTS SUPPRESSED BY SSW 2.

7090 SYSTEM PROJECT 19

ADVANCED INSTRUCTIONS

***** PRINT OUT A *****

9M51 NOW TESTING SENSE INDICATOR INSTRUCTIONS

9M51 TESTING FLOATING POINT INSTRUCTIONS

CORRECT FP EXECUTION VALUES AND CONDITIONS

ACC +00000000001 MQ +000400000000 LOC ZERO ALL ZERO NO FP TRAP

ERROR FP EXECUTION VALUES AND CONDITIONS

ERROR IN ACC 1 THRU 35

TEST LOC 65067, OPN FRN ,ERROR LOC 65117, 0 LOC 00000000000,SW 000001
LITE 0000, MQ 000400000000, XRA 00000, XRB 12662, XRC 12661, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 0,
INDS 040000000000 KEYS 002000052112

CORRECT FP EXECUTION VALUES AND CONDITIONS

ACC -100200000001 MQ -377777777777 LOC ZERO ALL ZERO NO FP TRAP

ERROR FP EXECUTION VALUES AND CONDITIONS

ERROR IN ACC 1 THRU 35

TEST LOC 65123, OPN FRN ,ERROR LOC 65155, 0 LOC 00000000000,SW 000001
LITE 0000, MQ 777777777777, XRA 00000, XRB 12624, XRC 12623, TRAP TGR 0
ACC -,Q 0,P 0,400000000000, DIV CK 0, ACC OVFL 0,
INDS 040000000000 KEYS 002000052112

CORRECT FP EXECUTION VALUES AND CONDITIONS

ACC -377777777777 MQ +001400000000 LOC ZERO ALL ZERO NO FP TRAP

ERROR FP EXECUTION VALUES AND CONDITIONS

ERROR IN ACC 1 THRU 35

TEST LOC 65161, OPN FRN ,ERROR LOC 65211, 0 LOC 00000000000,SW 000001
LITE 0000, MQ 001400000000, XRA 00000, XRB 12570, XRC 12567, TRAP TGR 0
ACC -,Q 0,P 0,400000000000, DIV CK 0, ACC OVFL 0,
INDS 040000000000 KEYS 002000052112

7000 SYSTEM PROJECT 20

ADVANCED INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOPS OCCURED AT 00001 AND 00002.

DEPRESSED START KEY.

PERFORMED A MANUAL TRANSFER TO LOCATION
52112 TO BYPASS LONG INDEX REG. TEST.

PRINT-OUT A OCCURED.
MACHINE HUNG UP. SEE PICK-SHEET 1.

7090 SYSTEM PROJECT 20

ADVANCED INSTRUCTIONS

***** PRINT OUT A *****

9M51 NOW TESTING SENSE INDICATOR INSTRUCTIONS

9M51 TESTING FLOATING POINT INSTRUCTIONS

7090 SYSTEM PROJECT 20

ADVANCED INSTRUCTIONS

PICK-SHEET NUMBER 1

PROGRAM COUNTER		●●● ●●● ●●● ●●● ●●●	
ADDRESS REGISTER		●●● ●●● ●●● ●●● ●●●	
INDEX REGISTER A		000 000 000 000 000	
INDEX REGISTER B		000 000 000 000 000	
INDEX REGISTER C		000 000 000 000 000	
PROGRAM REGISTER	0 S	●●● ●●● ●●●	SHIFT COUNTER 00 000 000
STORAGE REG.	0 S	●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●●	
ACCUMULATOR	● 0 0 S Q P	●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●● ●●●	
M Q CONTENTS	● S	●● 000 000 000 000 000 000 000 000 000 000 000	

0 I/O CHECK	0 FP	SENSE	CYCLE TIME
0 TRAP	● 9 CARRY	1 2 3 4 0 0 0 0	I L E B M 0 ● 0 0 0
0 SIMULATE	● 9 OVERFLOW	TALLY COUNTER	CHANNEL SELECT
0 ACC. OVERFLOW	0 Q CARRY	1 2 3 4 5 0 0 ● 0 0	A B C D E F G H 0 0 0 0 0 0 0 0
0 QUOT OVERFLOW	0 MST STOP	COMMAND TRAP	TAPE CHECK TRAP
0 R/W SELECT	0 END OP	A B C D E F G H 0 0 0 0 0 0 0 0	A B C D E F G H 0 0 0 0 0 0 0 0
0 DIVIDE CHECK	0 AND	CHANNEL TAPE CHECK	
0 TRAP CONTROL	0 CAQ	A B C D E F G H 0 0 0 0 0 0 0 0	
0 T2	0 X CARRY		

7090 SYSTEM PROJECT 21

ADVANCED INSTRUCTIONS

DIAGNOSTICS

PROCEDURES AND RESULTS

CALLED 9M51 FROM TAPE A1.

APPEARED TO LOAD IN NORMAL MANNER.
NORMAL STOPS OCCURED AT 00001 AND 00002.

DEPRESSED START KEY.

PERFORMED A MANUAL TRANSFER TO LOCATION
52112 TO BYPASS LONG INDEX REG. TEST.

PRINT-OUT A OCCURED.
FURTHER PRINT-OUTS SUPPRESSED BY SSW 2.

7090 SYSTEM PROJECT 21

ADVANCED INSTRUCTIONS

***** PRINT OUT A *****

9M51 NOW TESTING SENSE INDICATOR INSTRUCTIONS

9M51 TESTING FLOATING POINT INSTRUCTIONS

CORRECT FP EXECUTION VALUES AND CONDITIONS
ACC +203700000000 MQ +150Y00000000

ERROR FP EXECUTION VALUES AND CONDITIONS
ERROR IN ACC 1 THRU 35
ERROR IN MQ S,1 THRU 35

TEST LOC 67766, OPN FAD ,ERROR LOC 70010, 0 LOC 000000000000,SW 000001
LITE 0000, MQ 000000000000, XRA 00000, XRB 07771, XRC 07770, TRAP TGR 0
ACC +,Q 0,P 0,000000000000, DIV CK 0, ACC OVFL 0,
INDS 060000000000 KEYS 002000052112

CORRECT FP EXECUTION VALUES AND CONDITIONS
ACC -032600000000 MQ -377000000000 LOC ZERO +000001070263 FP TRAP

ERROR FP EXECUTION VALUES AND CONDITIONS
ERROR IN ACC 1 THRU =5
ERROR IN MQ S,1 THRU 35
FP TRAP ERROR CONDITION
ERROR IN POS 17 LOC ZERO
ERROR IN ADR OF LOC ZERO

TEST LOC 70257, OPN FSB ,ERROR LOC 70310, 0 LOC 000000000000,SW 000001
LITE 0000, MQ 400000000000, XRA 00000, XRB 07471, XRC 07470, TRAP TGR 0
ACC -,Q 0,P 0,400000000000, DIV CK 0, ACC OVFL 0,
INDS 070600000000 KEYS 002000052112

"PROGRAMMING INSTRUCTION MANUAL"
"PIM"

ADVANCED COMPUTER PROGRAMMING

Reference Material

The material contained in this text is not to be considered complete, but is only the information thought to be needed with the Advanced Computer Programming course. The material is not to be reproduced without consent by Education Research Department, Poughkeepsie, New York.

Your comments on the completeness, arrangement, and over-all usefulness of the material will be appreciated. Please address comments to:

O. W. Perry
Education Research
Education Building
Poughkeepsie, New York

CONTENTS

Computer Characteristics	426
Introduction	426
Instructions And Data	426
Register	429
Counter	431
Adder	432
Machine Cycles	432
Processing Unit Data Flow	435
Data Representation	436
Punched Cards	437
Standard IBM Card Code	437
Seven-Bit Alphameric Code	438
IBM 7090/7094 Card Reading	440
Row Binary Card Representation	443
Column Binary Card Representation	444
BCD Characters in 7090/7094 Storage	445
Computer Instructions	447
Instruction Specifications	447
Fixed-Point Arithmetic Instructions	450
Floating-Point Operations	452
Shifting Operations	453
Logic Operations	455
Packing and Unpacking	457
Word Transmission Operation	459
Indexing Concept	460
Complement Arithmetic	462
Multiple Tags	464
Index Transmission Operations	465
Indexing Operations	467
Sense Indicator Operations	473
Control Instructions	474
Arithmetic Control Instructions	475
Storage Testing Control Instructions	477
Index Register Testing Instructions	477
Sense Light and Indicator Testing Instructions	479
Instructions used With Trapping	480

CONTENTS

Input/Output Devices and Operations	482
Introduction	482
Reading and Writing	482
Data Buffering	483
Data Channel Operation	484
Magnetic Tape and Data Channel (7607) Addresses	486
Data Channel Registers (7607)	487
IBM 7607 Data Channels	491
Magnetic Tape Units	491
Character Alteration in BCD Mode	491
Data Channel Select Registers	492
Card Reader	494
Card Punch	494
Printer	494
Input-Output Transmission Operations	495
Data Channel Commands	497
Program Example	501
Unit Addresses	502
Data Channel Trapping	504
Floating Point Trap	506
Subroutines	507
Introduction	507
Open Subroutine	507
Closed Subroutine	508
Calling Sequence	509
IBM Programming Systems	513
Introduction	513
Operating Systems	513
Over-All Operation	516
FORTRAN Assembly Program	517
Elements of the FAP Language	518
Symbols	518
Symbolic Expressions	519
Literals	520
Data Generating Operations	522
Pseudo-Operations (extended operation codes)	524
Storage Allocation Operations	525
Symbol Defining Operations	526
Updating Symbolic Tapes	526
UPDATE Pseudo-Operations	527
Example of UPDATE	528
REWIND Pseudo-Operation	528
DELETE Pseudo-Operation	528
LBL (Label) Pseudo-Operation	530

CONTENTS

COUNT Pseudo-Operation	530
PCC (Print Control Card) Pseudo-Operation	531
Program Linking Pseudo-Operation	531
FORTRAN II Monitor Programming System	532
Monitor Control Cards	533
IBSYS - The Basic Monitor System	535
Functions of the Basic Monitor	536
Basic Monitor Sections	537
Specializing the Basic Monitor	538
The Nucleus of Basic Monitor	538
Communication Region and One-Word Entry Table	542
Unit Control Block	545
Unit Control Block Word 1 (UCW 1)	547
Unit Control Block Word 2 (UCW 2)	548
Unit Control Block Word 3 (UCW 3)	548
Available Units	549
Units That Are Not Available	551
Detached Units	551
Function Units	551
Reserve Units (Intersystem)	554
Symbolic Unit Designation	555
Program Execution Under Basic Monitor Control	556
The IBSYS Library	558
The Job Input	561
Running Under Basic Monitor Control	563
Control Cards	564
Basic Monitor Control Card Format	564
Control Card Categories	564
Operational Control Cards	565
Unit Assignment Control Cards	566
Tape Manipulation Control Cards	567
Miscellaneous Control Cards	567
The Input-Output Executor (IOEX)	568
Functions of IOEX	568
Advantages Offered To The User	568
The Use of IOEX by a Program	569
The Request Queue	569
Initiating the I/O Operation	569
Completion of the I/O Operation	570
A Detailed Investigation of IOEX Routines	571
Information Provided by IOEX	571
ACTIV	572
Save - XTRAP	573
BGI	575
The Sample Program	575

COMPUTER CHARACTERISTICS

INTRODUCTION

The processing unit controls and supervises the entire computer system and performs the actual arithmetic and logic operations on data. From a functional viewpoint, the processing unit consists of two sections: control and arithmetic-logic.

The control section directs and coordinates all operations called for by instructions. This involves control of input/output devices, entry or removal of information from storage, and routing of data between storage and the arithmetic-logic section. Through the action of the control section, automatic integrated operation of the entire computer system is achieved.

In many ways, the control section can be compared to a telephone exchange. Data transfer paths exist, just as there are connecting lines between all telephones serviced by a central exchange. The telephone exchange has a means to control the movement of sound pulses from one phone to another, to ring bells, to connect and disconnect circuits, and so on. The path of conversation between one telephone and another is set up by controls in the exchange itself.

In the computer, execution of an instruction involves the opening and closing of many paths or gates for a given operation. The control section can start or stop an input/output device, turn a signal indicator on or off, rewind a tape reel, or direct some process of calculation.

The arithmetic-logic section contains the circuitry to perform arithmetic and logic operations. The arithmetic portion calculates, shifts numbers, sets the algebraic sign of results, compares, and so on. The logic portion carries out the decision-making operations to change the sequence of instruction execution.

Instructions and Data

The only distinction between instructions and data in core storage is the time when they are brought into the processing unit. Information brought into the processing unit during an instruction cycle is interpreted as an instruction. Information brought into the processing unit during any other type of computer cycle is considered to be data. Consequently, the computer can readily operate on its own instructions if those instructions are supplied as data (that is, if an instruction is brought into the processing unit during any cycle other than in instruction cycle). Also, the computer can be instructed to alter its own instructions according to conditions encountered during the handling of a procedure.

It is this ability to process instructions that provides the almost unlimited flexibility and the so-called logical ability of the stored program system.

In the computer information (both data and instructions) is handled in fixed groups of 36 positions (bits) each. Each group is called a word. Each position within a word is named with the S (sign) position followed by positions 1 through 35. Computer instructions with an address in the operand part indicate the core storage location to be subjected to some arithmetic or logic operation. This address part, or field, always occupies bit positions 21 through 35 of the word (Figure 1).



Figure 1 Address Field of an Instruction

Capacity of the largest core storage available on the computer is 32,768 words of 36 positions each. The 15-position address field (positions 21-35) is just large enough to hold or indicate the largest core storage address. This address, expressed in the computer's language (code), is simply 15 consecutive 1's (Figure 2).



Figure 2 Maximum Address

Other core storage capacities available with the computer are: 16,384; 8,192; and 4,096 words. In a system with a 16,384-word capacity, the largest address is contained in 14 positions of the address field. The left-most position (position 21) is ignored. Similarly, a 8,192-word system uses 13 positions (23-35), and a 4,096-word system uses 12 positions (24-35) as its address field.

The operation part of most instructions is contained in word positions S, 1-11. Positions 21-35 of the same word would then contain the address of the operand to be used with the instruction. For example, assume that two factors, A and B, are to be added. In the computer, one of these factors is always taken from core storage by the add instruction; the other factor is already in a processing unit register called the accumulator. The accumulator factor must have been placed there by a previous instruction. Figure 3 shows the format of an add instruction

when A is the contents of core storage location 00001 and factor B is the contents of the accumulator.



Figure 3 Add Instruction Format

When this instruction is executed, factor A is added to factor B and the resulting sum is returned to the accumulator. Actual computer coding is used (binary code), and 36 positions are shown in groups of three for easier reading and conversion to the octal number system.

As an example of computer operation, assume that the accumulator contains the number + 1. If the number in location 00001 is a +2, the result of executing the add instruction is a +3. This is shown in Figure 4. A 0-bit in the S position indicates a plus number; a 1 indicates a minus number

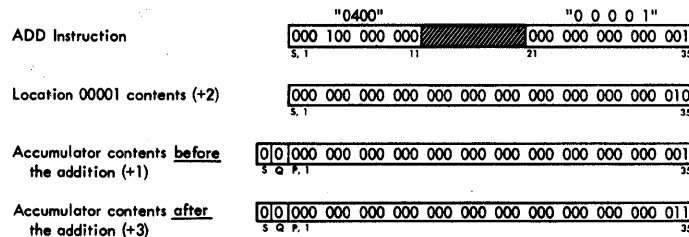


Figure 4 Add Example

Register

A register is a device capable of receiving information, holding it, and transferring it as directed by control circuits. The electronic components used may be magnetic cores, transistors, or similar components.

Registers are named according to function: an accumulator register accumulates results, a multiplier-quotient register holds either multiplier or quotient, a storage register contains information taken from core storage or sent to core storage, and address register holds the address of a storage location or device, and an instruction register contains the instruction code (operation part) of the instruction being executed (Figure 5).

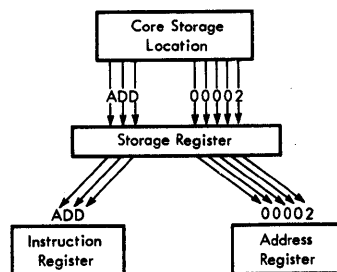


Figure 5 Register Nomenclature and Function

Registers differ in size and use. In some cases, extra register positions are used to detect overflow conditions during an arithmetic operation. The accumulator register is made up of 38 positions; 36 are used for data and two positions (P and Q) are used to remember overflow conditions. If, for example, two 36-bit binary numbers are added, it is possible that the result is a 37-bit answer.

In Figure 6 the accumulator register holds one factor, from storage, is in the storage register. The two factors are added and the result is placed back into the accumulator register, where the overflow is indicated by the presence of a 1 bit in the first (P) overflow position. The accumulator might then be shifted right one place and a record kept of the lost

low-order bit.

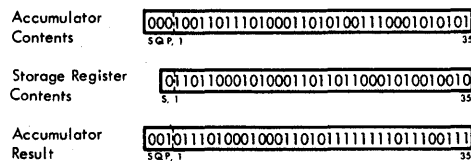
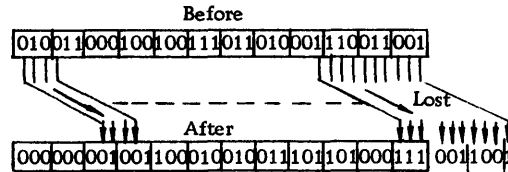


Figure 6 Execution of Add with Overflow

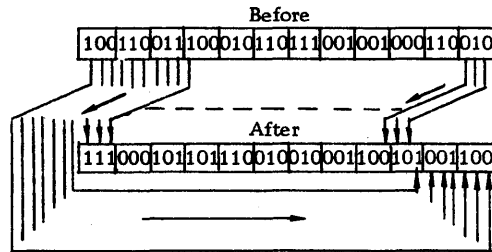
The contents of other registers can be shifted right or left within the register and, in some cases, even between registers. The effect, when shifting from one register to another, is the same as if the two registers were one large register. Figure 7 shows three types of shifting. With shifting within a register, data shifted out of the register may, or may not be lost, depending on the instruction used. With double register shifting, data shifted out of the registers are lost. In the types of shift operations where data loss is possible, vacated positions of the registers are filled with 0's.

In order uses, a register holds data while associated circuits analyze the data. For example, an instruction can be placed in a register, and circuits can determine the operation to be performed and locate the data to be used. Data within specific registers may also be checked for validity.

Single Register Shifting: (shift right seven places) Note: left-hand positions are filled with zeros; data shifted out of position 35 are lost. Dependent on the instruction, the sign position may or may not be shifted.



Single Register Shifting: (rotate left seven places) Note: data are not lost when shifted out of the S position; they are reentered into position 35.



Double Register Shifting: (shift right seven places) Note: data are shifted from position 35 of the first register into position S of the second register. Data shifted out of position 35 of the second register are lost. Vacated positions are filled with zeros.

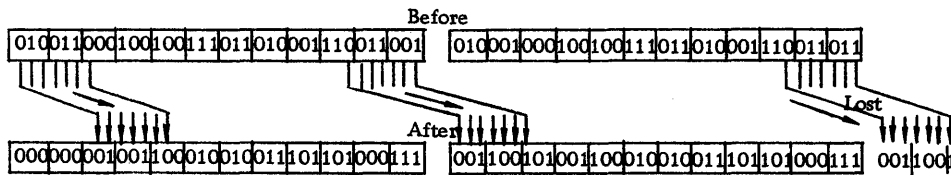


Figure 7 Register Shifting

The main registers of a system, particularly those involved in normal data flow and core storage addressing, have small lights associated with them. These lights are located on the operator's console for visual indication of register contents and various program conditions. If a light is on, a 1 bit is indicated for that position. If the light is off, a 0 bit is indicated.

Counter

The counter is closely related to a register and usually performs the same functions. In addition, its contents can be increased or decreased by some amount. The action of a counter is related to its design and use within

the computer system. Like a register, it may also have visual indicators on the operator's console.

Adder

The adder receives data from two or more sources, performs addition, and sends the result to a receiving register. Figure 8 shows two positions of an adder circuit with inputs from registers like the accumulator and storage register.

The sum is developed in the adder. A carry from any position is sent to the next higher-order position. The final sum goes to the corresponding position of the receiving register.

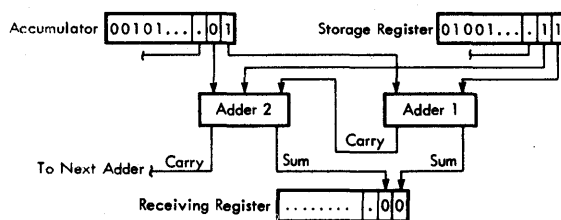


Figure 8 Adders in a Computer System

Machine Cycles

To receive, interpret, and execute instructions, the central processing unit must operate in a prescribed sequence. The sequence is determined by the specific instruction and is carried out during a fixed interval of time pulses. These intervals are measured by regular pulses emitted from an electronic clock at frequencies as high as a million or more per second. A fixed number of pulses determines the time of each basic machine cycle.

Within a machine cycle, the computer can perform a specific machine operation. The number of operations required to execute a single instruction depends on the instruction.

All instructions have one instruction (I) cycle and some require only an I cycle for complete execution. Other instructions require both I and an execute (E) cycle. Various machine operations are thus combined to execute each instruction.

Instruction Cycle. The first cycle required to execute an instruction is called an instruction (I) cycle. The time for this cycle is instruction or I time. During I time:

1. The instruction is taken from a main storage location and brought to the processing unit.
2. The operation part is decoded in an instruction register; this tells the machine what is to be done.
3. The operand is placed in an address register; this tells the machine what it is to work with.
4. The location of the next instruction to be executed is determined.

At the beginning of a program, the instruction counter is set to the address of the first program instruction. This instruction is brought from storage and, while it is being executed, the instruction counter automatically advances (steps) to the location corresponding to the space occupied by the next stored instruction. By the time one instruction is executed, the counter has located the next instruction in the program sequence. The stepping action of the counter is automatic; in other words, when the computer is directed to a series of instructions, it will execute these instructions one after another until instructed to do otherwise.

Assume that an instruction is given to add the contents of storage location 00002 to the contents of the accumulator register. Figure 9 shows the main registers involved and the information flow lines.

I time begins when the instruction counter transfers the location of the instruction to the address register. This instruction is selected from storage and placed in a storage register. From the storage register, the operation part is routed to the instruction register and the operand to the address register. Operation decoders then condition circuit paths to perform the instruction while the address register locates the operand.

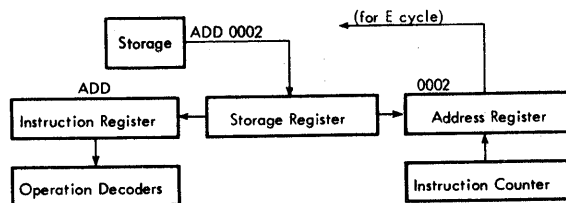


Figure 9 Computer I Cycle Flow Lines

Execution of instructions does not have to proceed sequentially. Certain instructions alter the process of sequential execution unconditionally. In this case, an instruction brought from storage indicates that the next sequential instruction is not to be executed but that one located in another position is next; the normal stepping of the instruction counter is altered accordingly. For instance, the instruction counter can be reset back to the beginning of the program so that the entire program can be repeated for another incoming group of data.

This transfer (branch) to alternative instructions may also be conditional. The computer can be directed to examine some indicating device and then transfer if the indicator is on or off. Such an instruction can say: "Look at the sign of the quantity in the accumulator; if this sign is minus, take the next instruction from location 5000; if the sign is plus, proceed to the next instruction in sequence." The instruction counter is set according to one of the two possible storage locations (5000, or the location of the next instruction in sequence). The logic path followed by the computer (that is, the precise sequence of instructions executed) may be controlled either by unconditional transfers or by a series of conditional tests applied at various points in the program. The arrangement of instructions in storage, however, is not normally altered.

Execute Cycle. I time is usually followed by one or more computer cycles that complete the operation being done by the computer. Execution of an E cycle results in bringing a word into the processing unit from core storage or in taking a word from the processing unit and placing it in core storage. Any word brought into the processing unit during an E cycle is

treated as data for the operation decoded by the previous I cycle. Figure 10 shows the data flow following the I time illustrated by Figure 9.

The E cycle starts by removing from storage the information located at the address (00002) indicated by the address register. This information is placed in the storage register. In this case, the core storage factor is then placed in the adders, together with the number from the accumulator. The contents of the storage register and accumulator are combined in the adders, and the sum is returned to the accumulator.

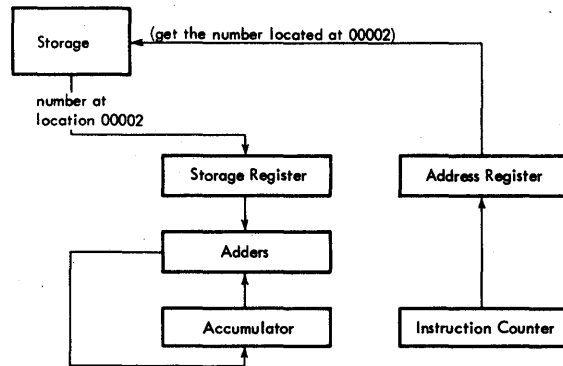


Figure 10 Computer E Cycle Flow Lines

The address register may contain information other than the storage location of data. It can indicate the address of an input/output device or a control function to be performed. The operation part of the instruction tells the computer how to interpret this information.

Processing Unit Data Flow

Instruction flow charts are included with many of the instruction descriptions to assist in understanding data and instruction flow through the processing unit. Figure 11 shows a simplified processing unit data flow. In this

figure, the positions of the word that are placed in a register or counter are shown below the component.

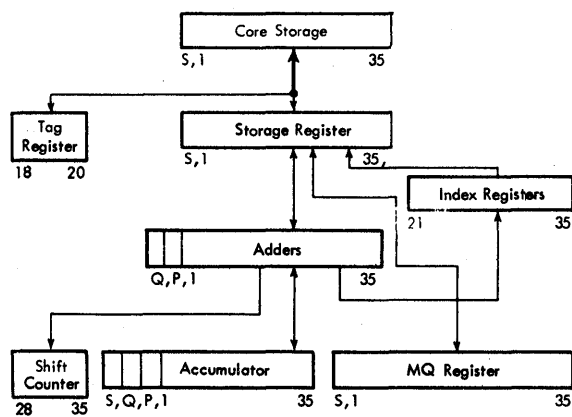


Figure 11 Simplified Processing Unit Data Flow

DATA REPRESENTATION

Symbols convey information; the symbol itself is not the information but merely represents it. The printed characters on this page are symbols and, when understood, convey the writer's meaning.

The meaning of symbols is one of convention. A symbol may convey one meaning to some persons, a different meaning to others, and no meaning to those who do not know its significance.

Presenting data to the computer system is similar in many ways to communicating with another person by letter. The intelligence to be conveyed must be reduced to a set of symbols. In the English language, these are the familiar letters of the alphabet, numbers, and punctuation. The symbols are recorded on paper in a prescribed sequence and sent to

another person who reads and interprets them.

Similarly, communication with the computer system requires that data be reduced to a set of symbols that can be read and interpreted by data processing machines. The symbols differ from those commonly used by people, because the information to be represented must conform to the design and operation of the machine. The choice of these symbols and their meaning is a matter of convention on the part of the designers. The important fact is that information can be represented by symbols, which become a language for the communication between people and machines.

Punched Cards

IBM cards provide 80 vertical columns with twelve punching positions in each column. The twelve punching positions form twelve horizontal rows across the card. One or more punches in a single column represent a card character. The number of columns used depends on the amount of data to be represented.

The card is often called a unit record because the data are restricted to the 80 columns and the card is read or punched as a unit of information. The actual data on the card, however, may consist of part of a record, one record, or more than one record. If more than 80 columns are needed to contain the data of a record, two or more cards may be used. Continuity between cards may be established by punching identifying information in designated columns of each card.

Information punched in cards is read or interpreted by a card reader and is recorded in a card by a card punch. Data are transcribed from source documents to punched cards by manually operated card punch machines.

Standard IBM Card Code

The standard IBM card code (originally called the "Hollerith" code) uses the twelve possible punching positions of a vertical column on a card to represent a numeric, alphabetic, or special character (Figure 12). The twelve punching positions are divided into two areas: numeric and zone. The first nine positions from the bottom edge of the card are the numeric punching positions and have an assigned value of 9, 8, 7, 6, 5, 4, 3, 2, and 1, respectively. The remaining three positions 0, 11, and 12, are the zone positions (the 0 position is considered to be both a numeric and zone position.)

The numeric characters 0 through 9 are represented by a single hole in a vertical column. For example, 0 is represented by a single hole in the 0 zone position of the column.

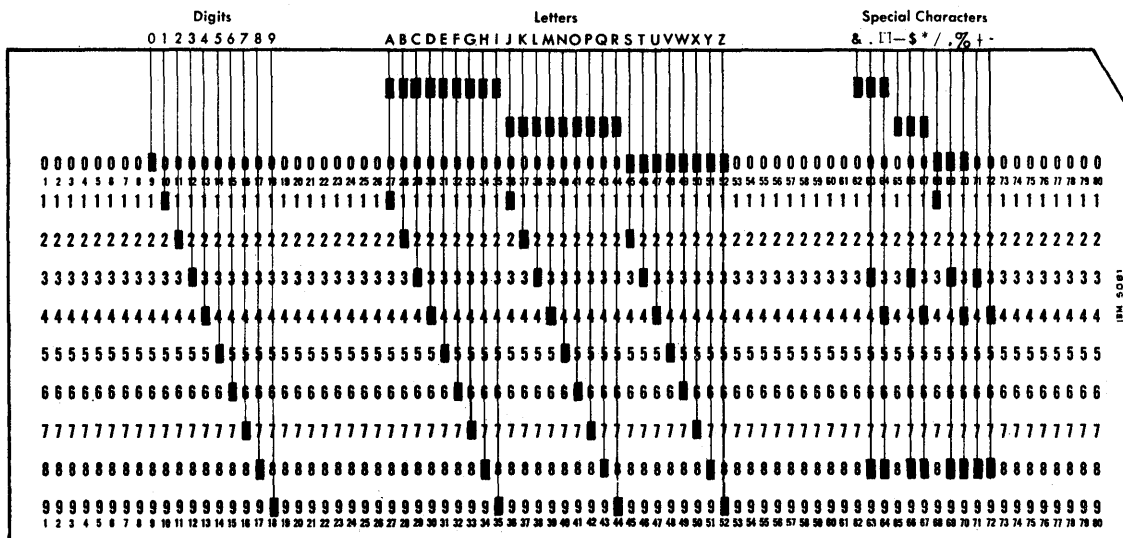


Figure 12. Standard IBM Card Code

The alphabetic characters are represented by two holes in a single vertical column: one numeric hole and one zone hole. The alphabetic characters A through I use the 12 hole punch and numeric holes 1 through 9, respectively. The alphabetic characters J through R use the 11 zone hole and a numeric hole 1 through 9, respectively. The alphabetic characters S through Z use the 0 zone hole and a numeric hole 2 through 9, respectively.

The standard special characters \$ * % and so on, are represented by one, two, or three holes in a column of the card and consist of hole configurations not used to represent numeric or alphabetic characters.

Seven-Bit Alphameric Code (Binary Coded Decimal)

In this code, all characters -- numeric, alphabetic, and special -- are represented (coded) using seven positions of binary notation. These positions are divided into three groups; one check position, two zone positions, and four numeric positions (Figure 13).

Check Bit	Zone Bits		Numeric Bits			
	B	A	8	4	2	1
C						

Figure 13. Bit Positions, Seven-Bit Alphameric Code

The four numeric positions are assigned decimal values of 8, 4, 2, and 1 and represent, in binary coded decimal form, the numeric digits 0 through 9 as shown in Figure 14.

Decimal Digit	Place Value			
	8	4	2	1
0	1	0	1	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Figure 14. Numeric Bits for Decimal Digits 0 through 9, BCD Code

Note that 0 is represented as 1010, actually the binary number for 10. The B and A zone bits are not present (00) when the numeric digits 0 through 9 are represented.

Combinations of zone and numeric bits represent alphabetic and special characters. The B and A bits provide for four possible bit combinations: 00, 10, 01, and 11. Figure 15 shows the zone and numeric bit combinations used to represent numeric, alphabetic, and special characters in what is loosely called the BCD systems (705, 1401, 1410, 7080, etc. data processing systems.)

The C position, known as the check bit, is used for code checking only. The check bit is present (is a 1-bit) in a character when the sum of the zone and numeric bits used to represent the character is odd.

CHAR.	C	BA	B421	CHAR.	C	BA	B421	CHAR.	C	BA	B421	Storage Mark and Drum Mark		
&	0	11	0000	-	1	10	0000	Blank	1	01	0000	0 0 00 0000		
ALPHABETIC	A	1	11	0001	J	0	10	0001	/	0	01	0001	CH. C BA B421 T 1 00 0001	
	B	1	11	0010	K	0	10	0010	S	0	01	0010	2 1 00 0010	
	C	0	11	0011	L	1	10	0011	T	1	01	0011	3 0 00 0011	
	D	1	11	0100	M	0	10	0100	U	0	01	0100	4 1 00 0100	
	E	0	11	0101	N	1	10	0101	V	1	01	0101	5 0 00 0101	
	F	0	11	0110	O	1	10	0110	W	1	01	0110	6 0 00 0110	
	G	1	11	0111	P	0	10	0111	X	0	01	0111	7 1 00 0111	
	H	1	11	1000	Q	0	10	1000	Y	0	01	1000	8 1 00 1000	
	I	0	11	1001	R	1	10	1001	Z	1	01	1001	9 0 00 1001	
	Plus Zero	0	0	11	1010	Minus Zero	0	1	10	1010	Record Mark	1	01	1010
SPECIAL CHAR	.	1	11	1011	\$	0	10	1011	,	0	01	1011	# 1 00 1011	
	h	0	11	1100	*	1	10	1100	%	1	01	1100	@ 0 00 1100	
Group Mark	0	11	1111									Tape Mark 0 00 1111		

Figure 15. Standard IBM BCD Character Code Chart

IBM 7090/7094 Card Reading

Cards to be read into the computer are placed in the card read hopper 9-edge first, face-down. This means that information punched in the 9-row of the card is read first, then the 8-row, 7-row, and so on, until the 12-row has been read. Only 72 of the possible 80 card columns are used; the first 72 columns will be used in this discussion.

Each of the 12 rows of the card is split into two parts, the left half consisting of columns 1 through 36 and the right half of columns 37 through 72; each half row can be treated as a 36-bit word and read into a core storage location. Figure 16 shows how the card is divided. Thus, there are 24 half-rows in the card or 24 possible words of 36-bits each.

The card reader regards any punched holes as a binary one. No hole indicates a binary zero. Thus, an 8-hole in column 36 of the card is regarded by the computer as a binary 1 in the low-order position of the word punched in the 8-row left. The high-order or left-most position of each half-row is reserved for the sign bit of the word. A binary 1 represents a negative sign, while a binary 0 represents a positive sign.

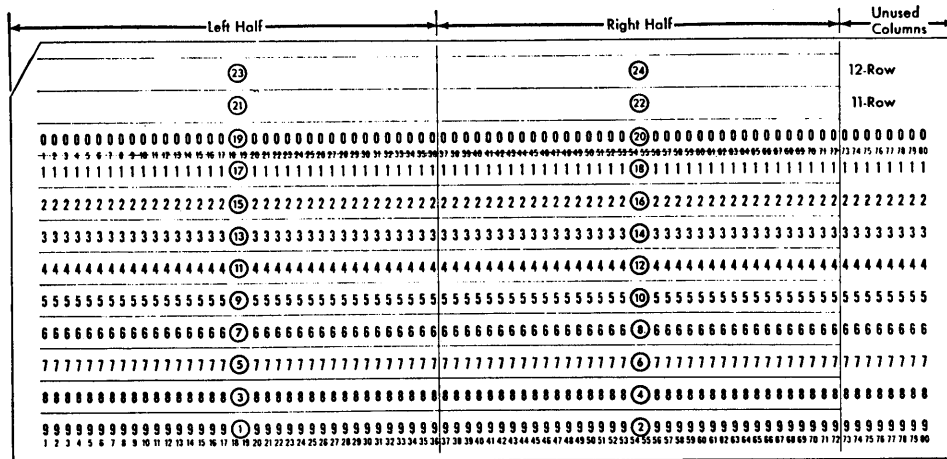


Figure 16. Card Data Positioning

Observe that this card representation of 24 binary words does not mean that the cards must always be punched with true binary information. The holes in the card can just as well be numerical punching in the standard Hollerith card code, alphabetic punching, or any configuration of data. It is necessary only to provide a suitable program for the computer which will convert from the binary reading method (that is built into the machine) to the particular code used on the card. This is not quite as easy as it sounds.

If the computer is programmed to read one entire 72 column card, and is told to put the first word into location 1000, data punched in the 9-row left goes into location 1000. Data from the 9-row right is placed into location 1001, 8-row left data goes into 1002, 8-row right into 1003, and so on until the 12-row right is placed into location 1027. This method of reading occurs despite the recording code used in the card.

For example, assume a card with the first 12 columns recorded in the Hollerith code are read into core storage starting with location 1000. The top half of Figure 17 shows how these 12 columns are placed into the 24 word card image in core storage. The contents of the odd locations of the core image are not shown since they are concerned with card columns 37 through 72 only.

With proper programming, the Hollerith characters can be converted into BCD words. These BCD words could then be converted to binary words (with more programming). The bottom half of Figure 17 shows the proper conversion from the card image to the BCD image, and from the BCD image to the binary image. Only the first six characters (032768₁₀) are converted to binary (1000000₈) since it would be meaningless to convert alphabetic characters to a binary number. It is, however, possible to do so.

032678A1 JRSZ

II

II

0000000000 00000000000000000000

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
1111111111 11111111111111111111111111

22 22222222 2222222222222222222222

3 3333333333333333333333333333333333

444444444444444444444444444444444444

5555555555555555555555555555555555!

666 6666666666666666666666666666666666

7777 7777777777777777777777777777777777

88888 888888888888888888888888888888888888

999999999 9999999999999999999999999999999999

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

72 column Hollerith card

24 Word Card
Image in Core

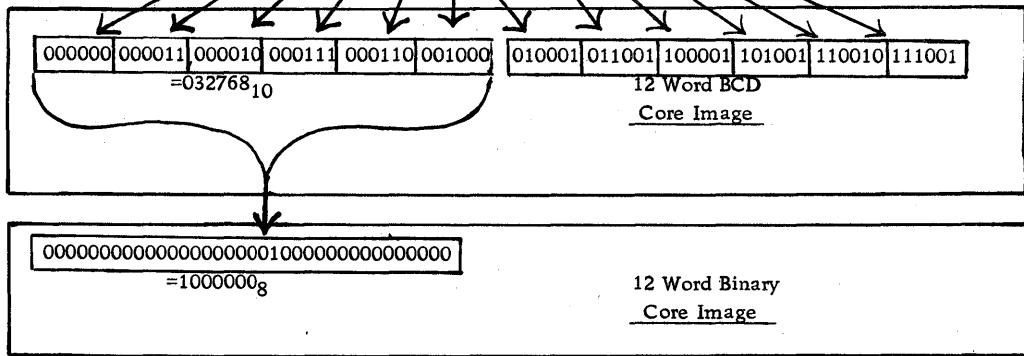
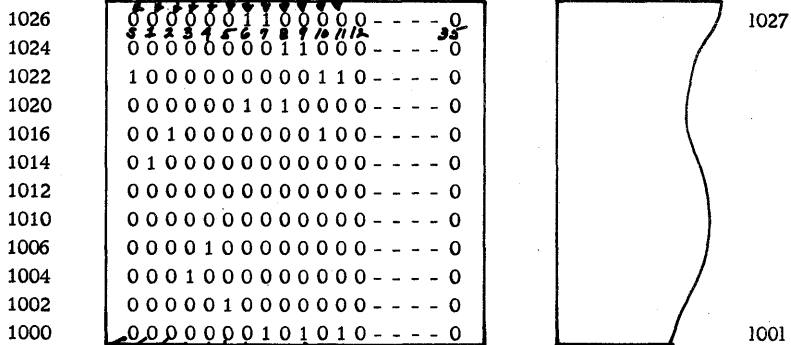


Figure 17. Data Images in Core Storage

Row Binary Card Representation

Row binary describes one method of recording binary information on cards. With this method, the information is arranged serially across each row of the card (left to right) starting at the 9-row, continuing to, and including the 12-row. This is the same arrangement as shown in Figure 16.

Row binary cards, when used with an assembly program such as the FORTRAN Assembly Program (FAP), use the entire 9-row of the card to contain information about the card (Figure 18). For example:

1. Card columns 4-18 contain a count of the number of words on the card, excluding the 9-row itself.
2. Columns 22-36 contain the address where the first word on the card will be placed in storage (the 8-row left word).
3. Columns 37-72 contain a binary number called a checksum that is computed with the add and carry logical instruction, of all words on this card except the 9-row right (checksum itself).

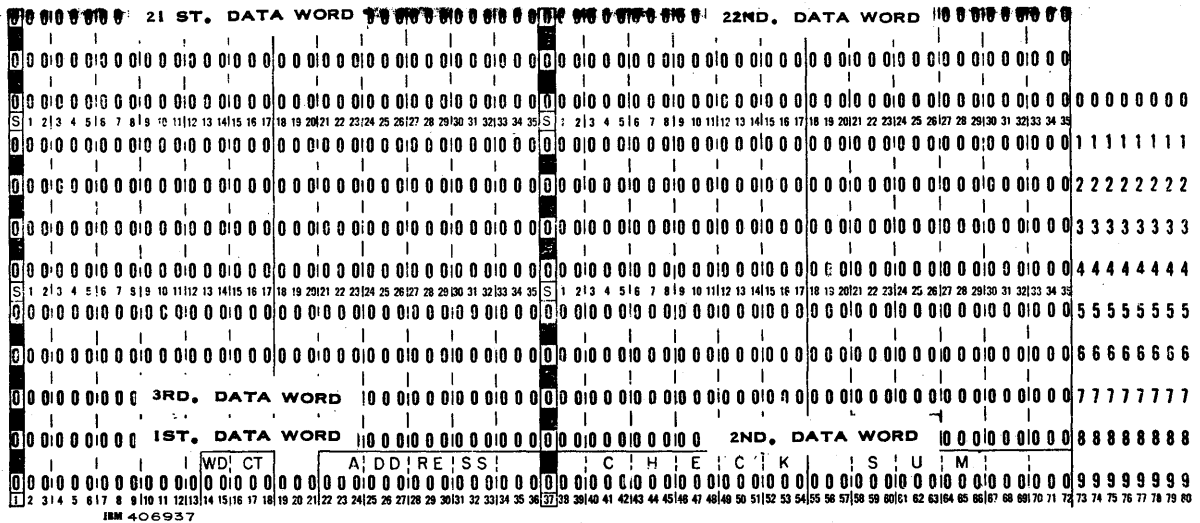


Figure 18. Row Binary Card

Column Binary Card Representation

Binary information may also be recorded in a column binary fashion. With this method, data are arranged in parallel with each column of the card containing 12 information bits. Thus, one full word (36 bits) would require three card columns. A card normally holds 24 words, but, not all are used for data. The first six card columns are used in a manner similar to the row binary card. For example (Figure 19):

1. Card column 1 contains a punch in the 7 and in the 9 row to designate the card as a column binary card.
2. Card column 2, rows 9 through 7 and card column 3, rows 9 through 12 contain the address of the first word when it is placed in core storage.
3. Card column 2, rows 3 through 11 contain a word count of the number of words on the card (excluding the first 6 card columns).
4. Card columns 4 through 6 contain a check sum that is computed in the same manner as with a row binary card.
5. Card column 7 through 12 contain relocation bits, assigned by the assembly program, that flag the decrement and address parts of instructions that will have to be changed because of relocation of the program.
6. First data word is contained in card columns 13, 14, and 15.

	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17	20	21	22	23	24	25						
0	C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
1	H	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
2	E	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	K																											4
5																												4
6	S																											5
7	U	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
8	M	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
9		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	9

Figure 19. Column Binary Card

Representation of BCD Characters in the 7090/7094 Storage

Both the 7090 and 7094 computers have the ability to process data coded in binary or BCD format. The 36-bit word of the computer allows the placement of six BCD characters within each computer word.

A sample representation of six BCD characters, as read from a strip of magnetic tape into storage, is shown in Figure 20. Note that the check bits for each character do not enter storage and that some of BCD characters on tape are changed as they are entered into storage. This change or alteration is automatic.

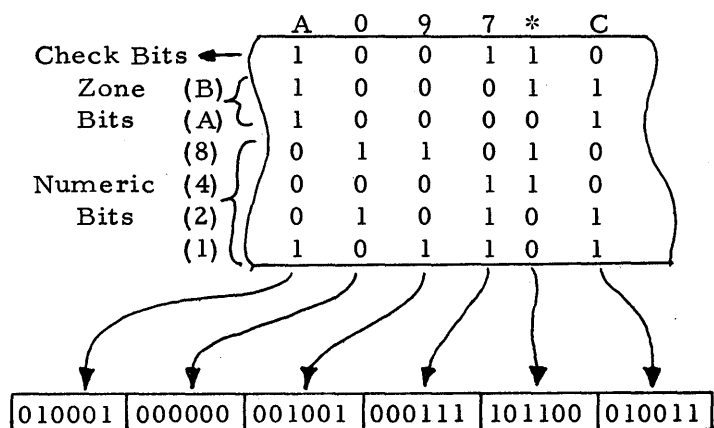


Figure 20 . BCD Character Representation

Figure 21 shows all BCD characters as they appear in main storage (9 code) and on magnetic tape (5 code).

Character	(9 Code)		Character	(9 Code)	
	In Storage	On Tape		In Storage	On Tape
0	00 0000	00 1010	-	10 0000	10 0000
1	00 0001	00 0001	J	10 0001	10 0001
2	00 0010	00 0010	K	10 0010	10 0010
3	00 0011	00 0011	L	10 0011	10 0011
4	00 0100	00 0100	M	10 0100	10 0100
5	00 0101	00 0101	N	10 0101	10 0101
6	00 0110	00 0110	O	10 0110	10 0110
7	00 0111	00 0111	P	10 0111	10 0111
8	00 1000	00 1000	Q	10 1000	10 1000
9	00 1001	00 1001	R	10 1001	10 1001
#	00 1011	00 1011	ō	10 1010	10 1010
@	00 1100	00 1100	\$	10 1011	10 1011
&	01 0000	11 0000	*	10 1100	10 1100
A	01 0001	11 0001	Blank	11 0000	01 0000
B	01 0010	11 0010	/	11 0001	01 0001
C	01 0011	11 0011	S	11 0010	01 0010
D	01 0100	11 0100	T	11 0011	01 0011
E	01 0101	11 0101	U	11 0100	01 0100
F	01 0110	11 0110	V	11 0101	01 0101
G	01 0111	11 0111	W	11 0110	01 0110
H	01 1000	11 1000	X	11 0111	01 0111
I	01 1001	11 1001	Y	11 1000	01 1000
ō	01 1010	11 1010	Z	11 1001	01 1001
.	01 1011	11 1011	#	11 1010	01 1010
□	01 1100	11 1100	,	11 1011	01 1011
			%	11 1100	01 1100

Figure 21. BCD Characters in Storage and on Tape

COMPUTER INSTRUCTIONS

INSTRUCTION SPECIFICATIONS

A symbolic instruction consists of four major divisions: location field, operation field, variable field, and comments field. The location field contains a name by which other instructions may refer to the instruction named. The operation field contains the machine operation or program system pseudo-operation, and the variable field normally contains the location of the operand. The comments field exists for the convenience of the programmer and plays no part in directing the computer.

Symbolic instructions are printed on the coding form (Figure 22) or are punched on a card form in the following format (one instruction to a line or to a card):

The location field, which may be blank, occupies columns 1 through 6.

Column 7 is always blank.

The operation field begins in column 8 and is from three to seven characters long.

A blank column separates the operation field and the variable field, which may begin in column 12 but may not begin after column 16.

The variable field does not normally extend beyond column 71 and must be followed by a blank column to separate it from the comments field.

The comments field follows the variable field and extends through column 80. If there is no variable field, the comments field may not begin before column 17.

Columns 73-80 are normally used for identification and serialization.

This section defines the computer instructions. The instruction format, shown in MAP language, appears with each instruction description. Preceding the format is the full name of the instruction. For example, the first instruction described appears:

Clear and Add -- CLA Y, T

This means that the instruction is a clear and add instruction with its operation symbolically expressed as CLA. The number of spaces between the operation part (CLA) and the variable field (Y, T) is four on the coding form. This gives the possible total of seven symbolic characters for the operation code. If the operation code were four characters long, only three spaces would separate the code from the variable field.

The comma (,) symbol in the variable field may be seen in the field heading in Figure . The Y symbol is used when the instruction requires an address part. A comma follows the Y symbol if indexing (specified by the T symbol) is to be used with the instruction. A second comma symbol follows the T symbol if decrementing or counting (specified by the V symbol) is a part of the instruction. For example, the transfer on index instruction format is TIX Y, T, V.

Instructions using indirect addressing are designated by use of the (*) asterisk symbol immediately following the operation code. For example, a normal add instruction is expressed as ADD Y, T; an add instruction using indirect addressing is expressed as ADD* Y, T.

Instruction descriptions use special terms and abbreviations:

1. C (Y) denotes the contents of storage location Y, where C (AC), C(MQ), and C(SR) denote the contents of the accumulator, multiplier-quotient, and storage registers. For example, C(MQ) S, 1-17 is read "the contents of positions S, 1 through 17 of the MQ register." When subscripts are not used, the entire register is implied. For example, C(AC) denotes the contents of accumulator positions S, Q, P, 1-35, inclusive.
2. When a register or part of a register, or a core storage location is cleared, the cleared part is reset to zeros.
3. The negative of a number is the number with its sign position reversed.
4. The magnitude of a number is the number with its sign position considered positive (a zero in position S corresponds to a positive sign).
5. In the alphabetic code of the instruction:
 - a. The letter Q designates the MQ register.

- b. The letter X in the second or third position designates use of an index register.
- c. The first letter of all transfer instructions is a T.

Fixed-Point Arithmetic Instructions

When dealing with fixed-point numbers, the first of the 36 data bits contains the algebraic sign of that number. A 0 signifies a positive number and a 1 signifies a negative number. The remaining 35 positions contain the magnitude of the number. When fixed-point instructions are used, the programmer must decide where the point is to be located. On the computer, the point that separates the integral part from the fraction part is termed a binary point.

Before any arithmetic operations can be executed, one of the numbers involved in the operation must be taken from core storage and placed in the appropriate CPU register. The arithmetic instruction is then given and the second number is brought from core storage and placed in the storage register.

The problem of $A + B$ could be solved with two instructions. The first would clear (or destroy) the formed contents of the accumulator register and place the first number in that register. The second would be brought from core storage, placed in the storage register, and then combined (or added) with the number in the accumulator. The actual adding occurs in the adders. When addition is complete, the result is placed back in the accumulator (which destroys the first number). The format of the first instruction (clear and add) is shown to demonstrate relationship between the symbols and the form.

All succeeding instruction formats are shown with spacing between the operation field and the variable field. The location field is not shown.

Add-ADD Y, T

Location				Operation	Address, Tag, Decrement/Count	Comments
1	2	6	7	8		
				ADD	Y, T	

The C(Y) are algebraically added to the C(AC). The resulting sum is placed in the AC. The C(Y) are unchanged. Numbers of the same magnitude but different signs give a zero result, whose sign is the same as the sign of the original AC. A carry from position 1 turns on the AC overflow indicator.

Change Accumulator Sign--CHS

If the sign of the AC is plus, it is made minus. If minus, it is made plus. The C(AC) Q, P, 1-35 are unchanged.

Clear and Subtract--CLS Y, T

The negative of C(Y) replaces the C(AC) S, 1-35. Positions P and Q of the AC are set to zero.

Subtract -- SUB Y, T

The C(Y) are algebraically subtracted from the C(AC). The difference replaces the C(AC) and the C(Y) are unchanged. As with the ADD instruction, overflow is possible from position 1 of the AC to position P and from P to Q, but carries from position Q are lost.

Multiply and Divide Operations

The arithmetic operations multiply and divide are accomplished in much the same manner as with add or subtract, except that the multiplier-quotient (MQ) register is used in addition to the accumulator register and the address.

With a multiply operation, the multiplier factor must be placed in the MQ register before execution of the actual multiply instruction.

Multiply --MPY Y, T

The C(Y) are multiplied by the C(MQ). The 35 most significant (high order) bits of the 70-bit product replace the C(AC) 1-35, and the least significant (low order) bits replace the C(MQ) 1-35. C(AC) Q, P positions are cleared to zero. The sign of the AC and MQ are set to the algebraic sign of the product. The number of bits to the right of the binary point of the first factor added to the number of bits to the right of the binary point of the second factor give the total number of bits to the right of the binary point in the product. The C(Y) are unchanged.

The programmer must know the size of the product that is possible for his problem. If this product cannot exceed 35 bits, the complete product will be in the MQ at the end of the MPY. In this case, a store multiplier-quotient (STQ) instruction may be used to get the product into core storage.

Divide or Proceed -- DVP Y, T

The divide operation assumes prior loading of the MQ and AC registers with the dividend. Maximum possible dividend is 70 bits. Dividend loading may be accomplished with a LDQ instruction if the dividend is 35 bits or less and it is known that the entire AC is set to zero, or with a CLA and a LDQ if the dividend exceeds 35 bits.

The C(AC) Q, P, 1-35 and the C(MQ) 1-35 are treated as a 70-bit dividend, plus sign, and the C(Y) as a 35-bit divisor. If the magnitude of C(Y) is greater than the magnitude of C(AC), division takes place. A 35-bit quotient replaces the C(MQ) 1-35 and the remainder replaces the C(AC) 1-35. The MQ sign is the algebraic sign of the quotient, and the AC sign is the sign of the dividend. If the magnitude of the C(Y) is less than or equal to the magnitude of the C(AC), division does not take place, the divide check indicator is turned on, and the computer program proceeds to the next sequential instruction. The C (Y) are unchanged.

Add and Carry Logical Word -- ACL Y, T

The C(Y) are added to the C(AC) P, 1-35 and the resultant sum replaces the C(AC) P, 1-35. The sign of Y is added to ACP and a carry from ACP is added to AC35. Positions S and Q of the AC are not affected and the C(Y) are unchanged.

Floating-Point Operations

When the range of numbers anticipated during a calculation is either large or unpredictable, it becomes difficult to work with fixed-point arithmetic instructions. An alternative set of floating-point instructions is available for such calculations. These instructions maintain the binary point automatically.

In the computer, the two numbers are expressed as binary fractions, each having an eight-bit binary characteristic to represent the exponent 2. The "lining up" is done by shifting from the AC into the MQ. The result of an addition or multiplication is normalized by shifting the fractions in the AC and MQ left while making compensating changes in the characteristic of the sum or product.

In the computer, a floating-point number is stored in a word location as shown in Figure 23.

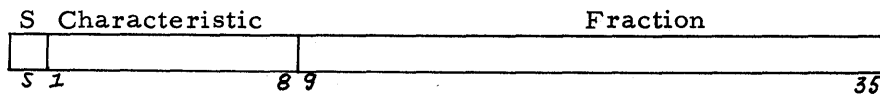


Figure 23 - Floating Point Word Format

The fraction is contained in bit positions 9 through 35. The sign of the fraction is contained in the S position of the word, and position 1 of the characteristic may be considered the sign of the characteristic. For example, an exponent of -32_{10} would be represented by a characteristic of 200_8 minus 40_8 or 140_8 . An exponent of 100_{10} would be represented by a characteristic of 200_8 plus 144_8 or 344_8 . Since 128_{10} is equal to 200_8 , the characteristic of a non-negative exponent always has a 1 bit in position 1 of the floating-point word, while the characteristic of a negative exponent always has a 0 bit in position 1. A normal zero has no bits in either the

characteristic or the fraction, and is the smallest possible zero available in this notation.

Floating Add -- FAD Y, T

The floating-point numbers located in Y and the AC are added together. The most significant portion of the result appears as a normal floating-point number in the AC. The least significant portion of the result appears in the MQ as a floating-point number with a characteristic 33 (octal) less than the AC characteristic. The signs of the AC and MQ are set to the sign of the larger factor. The sum in the AC and MQ is always normalized whether the original factors were normal or not. If C(AC)35 contain zeros, the FAD may be used to normalize an unnormal floating-point number.

Floating Subtract -- FSB Y, T

This instruction algebraically subtracts the number located in Y from the number in the AC and normalizes the result. The C(Y) are unchanged.

Floating Multiply -- FMP Y, T

The C(Y) are multiplied by the C(MQ). The most significant part of the product appears in the AC and the least significant part appears in the MQ. The product of two normalized numbers is in normalized form. If either of the numbers is not normalized, the product may or may not be in normalized form. The C(Y) are unchanged.

Floating Divide or Proceed -- FDP Y, T

The C(AC) are divided by the C(Y). The quotient appears in the MQ and the remainder appears in the AC. If the magnitude of the AC fraction is greater than or equal to twice that of the C(Y) 9-35, or if the magnitude of the C(Y) 9-35 is zero, division does not occur and the computer takes the next instruction in sequence. The quotient is in normal form if both the dividend and divisor are in normal form. The sign of the MQ is the algebraic sign of the quotient. If the AC fraction is zero, the C(AC) Q, P, 1-35 are cleared and the AC sign is set plus. The C(Y) are unchanged.

Shifting Operations

Shift instructions are used to move the contents of the accumulator and the MQ register either to the right or the left of their original positions. Except for the rotate MQ left instruction, zeros are automatically inserted in the vacated positions of a register. Thus, a shift larger than the bit capacity of the register causes the contents of the register to be replaced by zeros.

When a shift instruction is decoded during the I cycle, the amount of the shift is determined by the contents of bit positions 28-35 of the

shift instruction. This provides a maximum shift of 377_8 places. Any number larger than 377_8 is interpreted as modulo 400_8 , which means that, given any shift count, the actual number of positions shifted with the instruction is the remainder after dividing the shift count by 400_8 .

When the contents of a register are shifted right, the result is equivalent to dividing the original contents by a power of 2. Likewise, shifting to the left is equivalent to multiplying by a power of 2 (as long as no significant bits are lost).

In the following description of shift instructions, the number of positions to be shifted is specified by positions 28-35.

Accumulator Left Shift -- ALS Y, T

This instruction causes the C(AC)Q, P, 1-35 to be shifted left the number of places specified by Y. For example, ALS 3 would shift the C(AC) three places to the left. The sign position of the AC is not shifted (Figure 24). If a 1 bit is shifted into position P from position 1, the AC overflow indicator is turned on. Bits shifted past position Q are lost and vacated positions are filled with zeros.

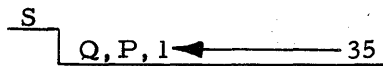


Figure 24 . ALS Schematic

Accumulator Right Shift -- ARS Y, T

The C(AC) Q, P, 1-35 are shifted right the number of places specified by Y. The sign position is not shifted (Figure 25), bits shifted from position 35 are lost, and vacated positions are filled with zeros. Bits shifted from position Q enter position P and bits from P enter position 1.



Figure 25. ARS Schematic

Logical Left Shift -- LGL Y, T

The C(AC)Q, P, 1-35 and C(MQ)S, 1-35 are treated as one register and are shifted left the number of places specified by Y. The sign of the AC is unchanged. Bits enter the MQS from MQ1 and go from MQS to AC35. If a 1 bit is shifted through AC position P, the AC overflow indicator is turned on. Bits are shifted from P to Q and bits shifted from Q are lost. Vacated positions are filled with zeros (Figure 26).

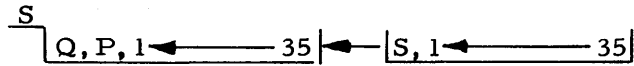


Figure 26 LGL Schematic

Logical Right Shift -- LGR Y, T

The C(AC)Q, P, 1-35 and C(MQ)S, 1-35 are treated as one register and shifted right the number of places specified by Y. The AC sign is unchanged. Bits enter MQS from AC35, and from MQS they are placed in MQ1. Bits shifted past MQ35 are lost and vacated positions are filled with zeros (Figure 27).

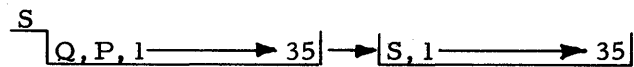


Figure 27. LGR Schematic

Rotate MQ Left -- RQL Y, T

The C(MQ) are shifted left the number of places specified by Y. Bits from MQS are routed to MQ35 and from MQ 1 into MQS, in effect, making the MQ register a circular register (Figure 28). For example, RQL 6 takes the six high-order bits (S, 1-5) of the MQ and places them in the low-order six positions (30-35). With the RQL, no bits are lost.

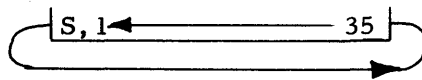


Figure 28. RQL Schematic

Logic Operations

Logic operations provide means for working on a 36-bit unsigned word or an individual character within a word. All logic operations interpret the sign position of the storage location addressed by the instruction as a numeric bit corresponding to position P of the accumulator. The sign position of the accumulator is either ignored or cleared.

The AND and OR concept is used, together with a process called masking, to accomplish the packing and unpacking of parts of words. When two numbers are combined with an AND operation, they are matched bit-

for-bit. If the same position in each word contains a 1 bit , the result is a 1 bit. If in one word the position is a 0 bit and in the other word it is a 1 bit, the result is a 0 bit. If the same position in both words is a 0 bit, the result is a 0 bit. For example:

This value, logically added to	101101011011
this value	<u>101001001101</u>
gives the resulting AND sum of	101001001001

An OR function (sometimes called inclusive OR) also matches two numbers bit-for-bit. The difference, however, when compared with an AND, is: (1) if the same position in either word contains a 1 bit, the result is a 1 bit; (2) if the same position in both words is a 1 bit, the result is again a 1 bit; (3) only if the same position in both words is a 0 bit, is the resulting position a 0 bit. For example:

This value, combined with	011010110101
this value by the inclusive OR	<u>001100100100</u>
gives the resulting OR of	011110110101

One other function of the logical operations is an exclusive OR. In this operation, only those positions which do not match result in a 1. If the same position in each word is a zero or if the same position in each word is a 1, the result is a zero. If the same position in one word is a 1 and in the other a 0, then the result is a 1. For example:

This value, combined with	101101100101
this value by the exclusive OR operation	<u>001011001101</u>
gives the resulting OR sum of	100110101000

AND to Accumulator -- ANA Y, T

Each bit of the C(Y)S, 1-35 is matched with the corresponding bit of the C(AC)P, 1-35. The sign position of Y is matched with the ACP. When the corresponding bits of both Y and the AC are 1 bits, a 1 bit replaces the contents of that position in the AC. When the corresponding bit of either location Y or AC, or both, is a 0 bit, a 0 bit replaces the contents of that position of the AC. The S and Q positions of the AC are set to zero and the C(Y) are unchanged.

AND to Storage -- ANS Y, T

Each bit of the C(AC)P, 1-35 is matched with the corresponding bit of the C(Y)S, 1-35, C(AC)P being matched with C(Y)S. When the corresponding bits of both the AC and location Y are 1's, a 1 replaces the contents of that position in location Y. When the corresponding bit of either the AC or the location Y, or both, is a 0, a 0 replaces the contents of that position in

location Y. The C(AC) are unchanged.

Exclusive OR to Accumulator -- ERA Y, T

Each bit of the C(Y)S, 1-35 is matched with the corresponding bit of the C(AC)P, 1-35, C(Y)S being matched with C(AC)P. When the corresponding position of the AC matches the position in location Y, a 0 replaces the contents of that position in the AC. When the corresponding position of the AC does not match the position in location Y, a 1 replaces the contents of that position in the AC. Positions S and Q of the AC are cleared to zeros and the C(Y) are unchanged.

OR to Accumulator -- ORA Y, T

Each bit of the C(Y)S, 1-35 is matched with the corresponding bit of the C(AC)P, 1-35. The sign of Y is matched with ACP. When the corresponding bit of either location Y or of the AC, or both, is a 1 bit, a 1 bit replaces the contents of that position in the AC. When the corresponding bits of both location Y and the AC are 0 bits, a 0 bit replaces the contents of that position of the AC. The C(Y) and the S and Q positions of the AC are unchanged.

OR to Storage -- ORS Y, T

Each bit of the C(AC)P, 1-35 is matched with the corresponding bit of the C(Y)S, 1-35, C(AC)P being matched with C(Y)S. When the corresponding bit of either the AC or location Y, or both, is a 1, a 1 replaces the contents of that position in location Y. When the corresponding bits of both the AC and location Y are 0's, a 0 replaces the contents of that position in location Y. The C(AC) are unchanged.

Packing and Unpacking

There are many cases where the information to be handled by the computer is made up of individual items, each of which is less than the size of a computer word. For example, it may be necessary to work with numbers no larger than three decimal digits. To conserve storage space, three such numbers can be stored in the same word as illustrated in Figure 27, where positions S, 14, and 25 are the sign positions of the numbers N1, N2, and N3 respectively. Handling of information in this manner is called "packing."

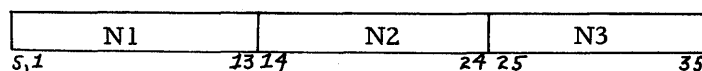


Figure 27. Diagram of a Packed Word

In addition to conserving storage space, packing also increases the entry and extra speed of information by reducing, for instance, the amount of magnetic tape which must be read or written.

Assume that a word in core storage has the form shown in Figure 27, and the number N2 is to be operated upon. Before arithmetic operations can be performed on this item, it must be separated from the other data in the word. The method of doing this is called "unpacking." The logical operations are employed in this type of operation, as they provide a flexible tool for carrying out the method. The number N2 is to be unpacked from the word without destroying the numbers N1 and N3. Therefore, the packing is done in the accumulator register, saving the packed word in core storage.

The program shown in Figure 28 will accomplish this. The mask used in the program contains 1's in position 14-24 and 0's elsewhere. The result of using this mask with the ANA instruction will place the number N2 in positions 14-24 of the accumulator. By varying the format of the mask, any of the three numbers could be unpacked (extracted) from the packed word.

Location	Operation	Address	Comments
	CAL	PAKWD	Place packed word into AC P, 1-35
	ANA	MASK	N2 is left in AC as a result of ANA
	ALS	14	Shift N2 until the sign occupies position P
	SLW	LOC2	Store N2 in location N2

MASK	OCT	000017774000	Mask configuration to obtain N2 only

Figure 28. Unpacking Program Example

After performing the desired arithmetic operations on the number N2, a new number N4, is the result. This number is the same size as N2 and is to be placed or packed (inserted) in location PAKWD replacing N2. Numbers N1 and N3 are to remain unchanged. The program shown in Figure 28 will accomplish this. The program assumes that the number N4 occupies positions S, 1-10 of location LOCN4. The mask used with the ANS instruction preserves the numbers N1 and N3 while replacing N2 with 0's.

Location	Operation	Address	Comments
	CAL	MASK	Place mask in AC P, 1-35
	ANS	PAKWD	Erase N2 from packed word in AC
	CAL	LOCN4	Place N4 in positions P, 1-10 of the AC
	ARS	14	Shift N4 into positions 14-24 of the AC
	ORS	PAKWD	Insert N4 into positions 14-24 of PAKWD
			location. Positions S, 1-13 and 25-35 remain unchanged.

MASK		OCT 777760003777	Mask to remove N2 from PAKWD

Figure 29 . Packing Program Example

Masking may be used to extract a full number or portion of a word from a given location instead of shifting and adjusting the result. Another example is shown in Figure 30 where a number located in positions 12-35 of location 00100 is to be extracted and stored in location 00200. The mask used is located in 00050 and consists of zeros in positions S, 1-11 and 7's in position 12-35.

Location	Instruction	Address	Comments
	CLA	00100	Put the number in the accumulator
	ANA	00050	Extract positions 12-35 of the AC
	STO	00200	Store the result, properly aligned.
	HTR		
00050		OCT 0000 7-----7	Mask used to extract positions 12-35

Figure 30. Masking Program Example

Word Transmission Operation

The instructions described in this section are concerned with the movement of words or parts of words from one core location or register to another.

Load Multiplier-Quotient -- LDQ Y,T

The C(Y) are loaded into the MQ and the C(Y) are unchanged. After the MQ register is loaded with the multiplier, the multiply instruction may be executed.

Store MQ -- STQ Y,T

This instruction places the C(MQ) into the location specified by Y. The C(MQ) remain unchanged.

Clear and Add -- CLA Y, T

The C(AC)S, 1-35 are replaced with the C(Y). Positions P and Q of the AC are set to zeros and the C(Y) remain unchanged.

Store--STO Y, T

The C(AC)S, 1-35 replaces the C(Y) and the C(AC) remain unchanged.

Clear and Add Logical Word--CAL Y, T

The C(Y) replaces the C(AC)P, 1-35. The sign of Y appears in ACP and accumulator positions S and Q are set to zero. The C(Y) are unchanged.

Store Logical Word--SLW Y, T

The C(AC)P, 1-35 replace the C(Y). The P position of the AC is sent to YS and the C(AC) remain unchanged.

Store Prefix--STP Y, T

The C(AC)P, 1, 2 replace the C(Y)S, 1, 2. The C(Y)3-35 and the C(AC) remain unchanged.

Store Address -- STA Y, T

The C(AC)21-35 replaces the C(Y)21-35. The C(Y)S, 1-20 and the C(AC) remain unchanged.

Store Decrement -- STD Y, T

The C(AC) 3-17 (decrement part) replace the C(Y)3-17. The C(Y)S, 1, 2, 18-35 and the C(AC) remain unchanged.

Store Zero --STZ Y, T

The store zero instruction may be used to change the contents of an entire core storage location to zeros. The C(Y) are replaced by 0 bits (sign of Y is made plus).

Store Instruction Location Counter--STL Y, T

The location of the STL instruction plus one replaces the C(Y) 21-35. The C(Y) S, 1-20 are unchanged.

Exchange AC and MQ -- XCA

The C(AC) S, 1-35 are exchanged with the C(MQ) S, 1-35. Positions P and Q of the AC are set to zeros.

Exchange Logical AC and MQ -- XCL

The C(AC) P, 1-35 are exchanged with the C(MQ) S, 1-35. Positions S and Q of the AC are set to zeros.

Zeros to Accumulator -- ZAC

When the ZAC pseudo-instruction is executed, all positions of the accumulator are replaced with zeros.

INDEXING CONCEPT

Indexing instructions may be used to modify addresses of existing instructions, reducing the number of core storage locations needed for instruction storage.

Indexing is the ability of a computer to combine the contents of an index register with the address portion of an instruction before the instruction is executed. There are two main reasons for indexing: (1) the

instruction as it appears in core storage is never changed and therefore its address never has to be initialized (set at the beginning of the program run), and (2) many addresses can be modified by the same index register's contents.

The index registers may be loaded with either true or complement numbers. When combined with the address part of an instruction, the address may be either increased or decreased depending on the type of number (true or complement) the index register started with.

The computer has three index registers. These registers are termed A, B, and C or 1, 2, and 4. The latter terminology is more convenient for the programmer working in machine language because the numbers 1, 2, and 4 are the octal representation of the addresses of the registers. Index register addresses are specified in a part of the instruction word known as the tag field. The tag field tells the computer whether an instruction is going to use an index register and, if so, which register is to be used. The tag field is located in bit-positions 18, 19, and 20 of the instruction word (Figure 31). By having more than one tag bit in the tag field, two or more index registers may be used by a single instruction. Thus, the contents of the index registers would be combined and the resultant OR (see "Packing and Unpacking") would be used. With some indexing instructions, the omission of 1 bits in the tag field simulates an index register of all zeros.

Register	Operation Part	Tag Field	Address Part
A or 1		001	
B or 2		010	
C or 4		100	

8,1 11 12 17 18 20 21 35

Figure 31. Index Register Tag Bits

The computer index registers are 15 positions long - large enough to hold the largest possible storage address. Instructions are available to test index register contents and control program instruction execution depending on that content. The contents of index registers may also be reduced or increased by variable amounts.

Complement Arithmetic

When index registers are used for address modification, the contents of an index register is always subtracted from an instruction's address. Since neither the address of the instruction nor the contents of the index register is associated with any algebraic sign, it is not possible to accomplish effective address modification by addition in any direct manner. However, addition is accomplished by using complement arithmetic. The following definitions apply to this type of arithmetic:

The 1's Complement of a binary number is the number that results by replacing each 1 in a number with a 0 and each 0 with a 1. For example, given the binary number of 101; the 1's complement would be 010. Also, the sum of a binary number and its 1's complement is a binary number composed of all ones (101 010 = 111).

The 2's Complement of a binary number is defined as the 1's complement of a number increased by one. Thus, for the preceding example, the 2's complement of a number (101) would be 011. If the 2's complement of a number occupies an index register and is used to modify an address, the effective address is the sum of index register contents and the address portion of the instruction. If the true number occupies the index register, the effective address is the difference between index register contents and the address part of the instruction.

Note that since both the contents of an index register and an instruction address are 15-bit numbers, all carries out of the leftmost position are lost.

As an example of the arithmetic involved when index registers are used, assume that index register (XR) 1 contains the binary number 2 and that an ADD instruction with a tag of 1 and an address of 200_g is to be executed (Figure 32). When the ADD instruction is decoded, the tag bit in position 20 specifies XR1. The contents of XR1 are complemented (2's complement) and placed in the adders. Note that index register contents are always automatically (2's) complemented when taken to the adders. This feature results in subtracting the contents of the XR from the address part of the instruction. The address part of the ADD instruction is also placed in the adders; after adding the two numbers, the result (called the effective address) is used in execution of the ADD instruction instead of the actual address. In this case, the effective address is 176_g.

If the programmer wishes to increase the effective address, the number placed in the XR is inserted in 2's complement form by instruction. Thus, when the address of the instruction and XR contents are combined, the result is an additive process. Using the same facts (as in Figure 32) with the XR contents in 2's complement form, the effective address is now 202_g instead of 176_g (Figure 33.)

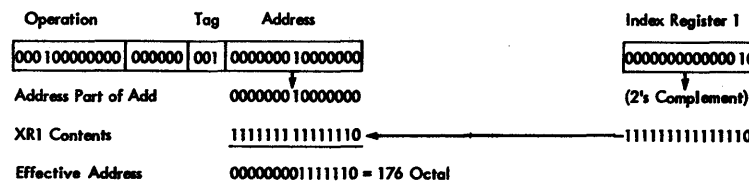


Figure 32. Index Register Arithmetic, Subtracting

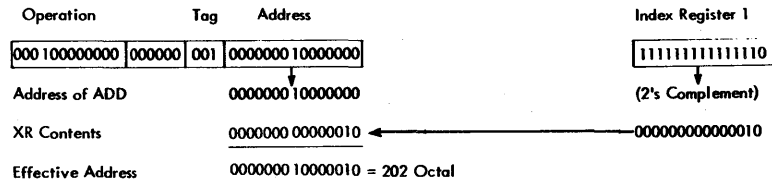


Figure 33. Index Register Arithmetic, Adding

Multiple Tags

As previously stated, an instruction may refer to more than one index register by placing multiple 1 bits in the tag field (Figure 34). Thus, a tag of 3_8 specifies index registers 1 and 2. Care must be exercised when multiple tags are used. The use of multiple tags results in a "logical OR" (see "Packing and Unpacking") of the contents of the specified index registers. For example, if a tag of 3 is given, the 15 positions of index register 1 are matched against the corresponding 15 positions of index register 2. If corresponding positions of each register contain 1 bits, the resultant logical sum is a 1 bit. If both positions are 0 bits the logical sum for that position is a 0 bit.

Tag Field		Index Registers Specified	
Binary	Octal	Octal	Alpha
000	0	None	None
001	1	1	A
010	2	2	B
011	3	1 and 2	A and B
100	4	4	C
101	5	1 and 4	A and C
110	6	2 and 4	B and C
111	7	1, 2, and 4	A, B, and C

Figure 34. Multiple Index Register Tags

Assume that index register 1 contains 03204_8 (000 011 010 000 100) and index register 2 contains 03061_8 (000 011 000 110 001). The instruction ADD 06521_8 , with an index tag field of 3, causes the "inclusive OR" (see "Packing and Unpacking") of the contents of the two registers as shown in Figure 35.

The OR'ed result of 03265_8 is subtracted from the address of the ADD 06521_8 instruction. The effective address received from the subtraction is 03234_8 , which the ADD instruction uses.

Index register 1 contents	000 011 010 000 100
Index register 2 contents	<u>000 011 000 110 001</u>
Inclusive OR'ed result	000 011 010 110 101 or
	03265 ₈

Figure 35. Inclusive OR Example

Index Transmission Operations

This section of operations deals with the loading and storing of the contents of index registers.

The operations always involve one or more index registers and either the address or decrement field of some location in storage or the accumulator register. The following 15-bit fields may serve as one of the agents in an index transmission operation: the address or decrement of the accumulator, the address or decrement of any location in storage, or the address part of the index transmission instruction itself. In addition, the number to be loaded may be placed in the specified index register in either true or complement form.

Single registers or any combination of index registers may be specified. If more than one register is specified in an unloading operation, their contents are "OR'ed" together to produce the effective number. OR'ing matches the registers position-for-position. If there is a bit in either or both of the registers, the result is a bit. For example:

XRA	101100
XRB	<u>011000</u>
RESULT	111100

Load Index from Address -- LXA Y,T

The C(Y) 21-35 replace the contents of the specified XR. The C(Y) are unchanged. A tag of zero results in a no-operation.

Load Complement of Address in Index -- LAC Y,T

The 2's complement of the C(Y)21-35 replaces the contents of the specified XR. For example, LAC 5,2 takes positions 21-35 of core location 5 and places the 2's complement of this value in index register 2. The C(Y) remain unchanged. A tag of zero results in a no-operation.

Load Index from Decrement -- LXD Y, T

The C(Y)3-17 replace the contents of the specified index register. The C(Y) are unchanged. A tag of zero results in a no-operation.

Address to Index True -- AXT Y, T

The value specified in the Y portion of this instruction replaces the contents of the index register specified by the T portion of this instruction. For example, AXT 30, 1 places the decimal value 30 (coded in binary format) in index register 1. The instruction itself remains unchanged. A tag of zero results in a no-operation.

Address to Index Complemented -- AXC Y, T

The 2's complement of positions 21-35 of this instruction replaces the contents of the specified index register. The instruction is unchanged. A tag of zero results in a no-operation.

Place Address in Index -- PAX , T

The C(AC)21-35 replaces the contents of the specified XR. The C(AC) are unchanged. A tag of zero results in a no-operation.

Place Decrement in Index -- PDX , T

The C(AC)3-17 replace the contents of the specified XR. The C(AC) are unchanged. A tag of zero results in a no-operation.

Place Index in Address -- PXA , T

The entire accumulator is cleared to zero, and the contents of the specified XR are placed in AC 21-35. With a tag of zero, the C(AC) are set to zero. The C(XR) are unchanged.

Place Index in Decrement -- PXD , T

The entire accumulator is cleared and the contents of the specified XR are placed in AC3-17. With a tag of zero, the C(AC) are set to zero. The C(XR) are unchanged.

Store Index in Address -- SXA Y, T

Positions 21-35 of the location specified by Y are replaced by the contents of the specified XR. The C(Y)S, 1-20 and the C(XR) are unchanged. With a tag of zero, the C(Y)21-35 are set to zero.

Place Complement of Address in Index -- PAC Y, T

The 2's complement of the C(AC) 21-35 replaces the contents of the specified XR. The C(AC) are unchanged. A tag of zero results in a no-operation.

Store Index in Decrement -- SXD Y, T

The C(Y) 3-17 are replaced by the contents of the specified XR. The C(Y)S, 1, 2, 18-35 and the C(XR) are unchanged. With a tag of zero, the decrement (positions 3-17 of the specified Y) is replaced with zeros.

Load Complement of Decrement in Index -- LDC Y, T

The 2's complement of the C(Y) 3-17 replaces the contents of the specified index register. The C(Y) are unchanged. A tag of zero results in a no-operation.

Figure 36 shows the data flow paths between storage, accumulator, and index registers for index transmission instructions. The action of each instruction, the registers concerned, and whether a true or complement value is used is also summarized.

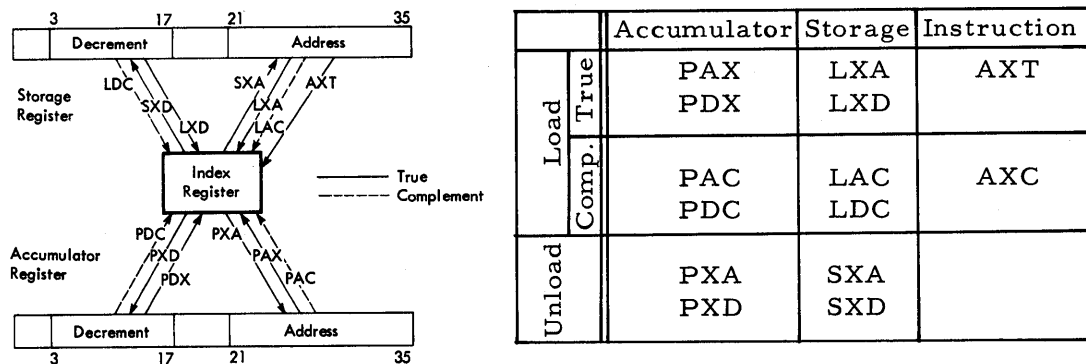


Figure 36. Index Transmission Data Flow and Summary

Indexing Operations

Several techniques may be used to increase program efficiency. One technique is address modification; another is indirect addressing. Two approaches to address modification are considered here: the destructive type and the indexing type.

Destructive Address Modification. The term destructive means that the original address of the instruction being modified is destroyed as it is modified. Regardless of the computer used, this is the method of address modification used unless the computer is equipped with index registers and indexing instructions. If an application required an instruction to be repeated many times, that instruction would have to be duplicated in the program and stored in core storage. For example, if the contents of 50 word locations were to be added together, 50 add instructions would have to be placed in the stored program. Each add instruction would have as its address part, the storage location for one of the 50 words.

The technique of modifying an instruction's address may be used to reduce the number of stored instructions. This technique, however, does increase over-all execution time for the problem. Using the same example as above, assume that the 50 word locations are designated A, A1, A2, etc. Figure 37 shows a program that could add the contents of these locations.

Note the use of the * (asterisk) symbol in the address part. When used this way, the * means the location of the instruction itself. Thus, the `cla *-2` means to bring into the accumulator the contents of the location that is two locations in front of the `cla` instruction location (Add First 1).

		Date	Page	of
Location	Operation	Address, Tag, Decrement/Count	Comments	
	ORG		Locate the PROGRAM	
START	CLA	FIRST	Locate FIRST Number	
	ADD	FIRST+2	The address of this instruction is changed	
	STO	TEMP	Temporary STORAGE location	
	CLA	*-2	BRING instruction into Accumulator	
	ADD	=1	(ONE) literal.	
	STO	*-9	STORE ALTERED instruction	
	SUB	COUNT	Reduce the Number Counter	
	IZE	END	TEST FOR END	
	CLA	TEMP	TEMPORARY STORAGE location	
	TRA	START+1	RETURN to Add Next Number	
COUNT	ADD	FIRST+49	Constant for Number Counter	
FIRST	BSS	50	Reserved STORAGE AREA	
	END		LAST Symbolic instruction	

Figure 37 . Address Modification Example (Destructive)

Indirect Addresses. All instruction addresses discussed in preceding illustrations are classified as direct, that is, they refer directly to the location of data or other instructions in storage, they select a system component, or they specify the type of control to be exercised.

Addresses may also be indirect. Such an address can refer only to a storage location that contains another address. The second address in turn refers to the location of data, a system component, or a control function.

Indirect addressing is particularly useful in performing address modification. For example, in a program it may be necessary to address a number of instructions to a single device, such as a magnetic tape unit. The same unit may be selected for reading, for error routines, in restart procedures, or in other transfer routines. If this unit is to be alternated with some other unit, the address part of all instructions involving that unit must be modified. Without indirect addressing, a number of modification instructions would be needed.

However, if the select instructions involving a tape unit are indirectly addressed to one storage location, that location can contain a single address, the address of the tape unit being used. Therefore, to change or modify all select instruction addresses, it is only necessary to modify the single effective address to which the select instructions refer (Figure 38.) In this test, the asterisk (*) is used with the operation code to designate indirect addressing. Any number of indirect addresses throughout a program may refer to a single effective address.

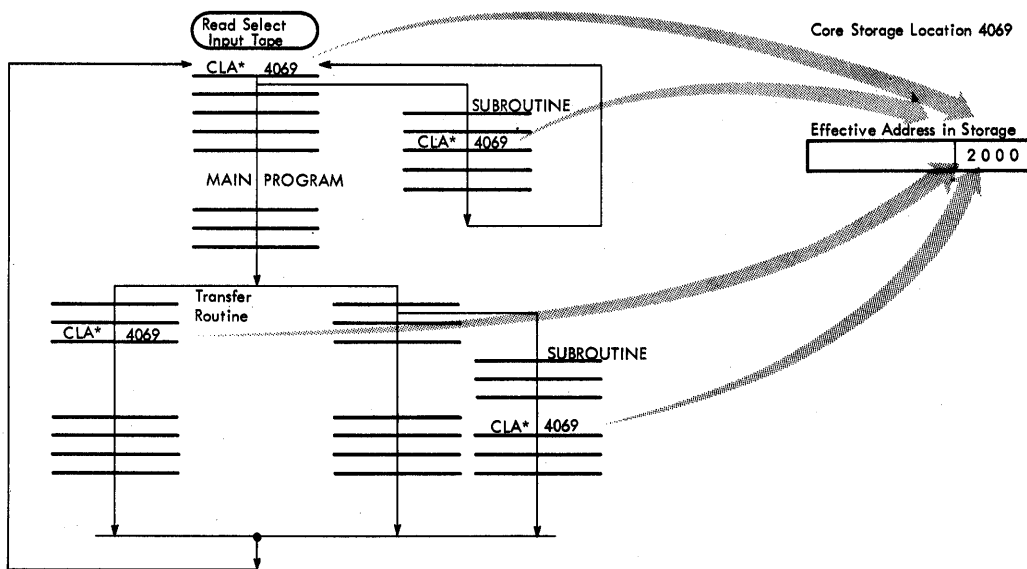


Figure 38. Indirect Address

Figure 39 shows a sample program using indirect addressing (the program shown in Figure 37 is altered to an indirectly addressed program.)

Problem							
Indirectly Addressed Checksum Routine							
Coder				Date		Page of	
1	2	6	7	8	Location		
Operation					Address, Tag, Decrement/Count		
Comments							
					BLKSM	CLA	2,4
						SXA	SAVE, 1
						PAX	0, 1
						ADD	1, 4
						STA	ADDER
						CLM	
					ADDER	ACL	0, 1
						TIX	ADDER, 1, 1
						SLW *	3, 4
					SAVE	AXT	0, 1
						TRA	4, 4

472

Figure 39. Indirect Addressing Example

Sense Indicator Operations

The following 24 instructions make reference to the 36-bit sense indicator (SI) register. The 36 bits of the SI may be thought of as switches which may be turned on or off and tested either singly or in groups by the program.

The contents of the SI register are manipulated through the use of a mask. The mask is a bit pattern comprised of 1's and 0's which may appear in an instruction, the AC, or any storage location. All masks for SI operations are used in the same way; that is, each position in the mask is compared with the corresponding position in the SI register.

For the positions in the mask which contain a 1, the corresponding position in the SI is either modified or tested depending upon the SI operation used. For the zero bits in the mask, the corresponding positions of the SI are not affected.

Four of the sense indicator operations are concerned with the transmission of full 36-bit words between the SI and either the AC or core storage. The remaining 20 operations are used to test or modify the C(SI). These 20 operations may be classified by the following five functions:

1. Set or Logical OR. These operations replace with a 1, the contents of each SI position selected by the mask.
2. Reset. These operations replace with a 0 the contents of each SI position selected by the mask.
3. Invert. These operations replace the contents of each SI position selected by the mask with its complement; i.e., 1's are replaced by 0's and 0's are replaced by 1's.
4. On Test. These operations examine the contents of each SI position selected by the mask. If all examined positions contain a 1, the calculator will either transfer to a location Y or skip the next instruction, depending upon the testing operation used.
5. Off Test. These operations examine the contents of each SI position selected by the mask. If all examined positions contain a 0, the calculator either transfers to location Y or skips the next instruction, depending upon the testing operation used.

Load Indicators -- LDI Y, T

The C(Y)S, 1-35 replace the C(SI) 0-35. The C(Y) are unchanged.

Store Indicators -- STI Y, T

The C(SI) 0-35 replace the C(Y)S, 1-35. The C(SI) are unchanged.

Set Indicators of Left Half -- SIL V

Each bit in positions 18-35 (V) of this instruction is matched with the corresponding bit of the C(SI) 0-17. The C(SI) 18-35 and V are unchanged. When the corresponding bit of either, or both V or the SI is a 1, a 1 replaces the contents of that position in the SI. When the corresponding bit of both the V and the SI is a 0, a 0 replaces the contents of that SI position.

Set Indicators of Right Half -- SIR V

Each bit in position 18-35 (V) of this instruction is matched with the corresponding bit of the C(SI) 18-35. The C(SI) 0-17 and V are unchanged. When the corresponding bit of either, or both, V or the SI is a 1, a 1 replaces the contents of that position in the SI. When the corresponding bit of both the V and the SI is a 0, a 0 replaces the contents of that SI position.

Place Indicator in Accumulator -- PIA

The C(SI) 0-35 replace the C(AC) P, 1-35. Positions S and Q of the AC are cleared to zeros and the C(SI) are unchanged.

OR Storage to Indicators -- OSI Y, T

Each bit of the C(Y)S, 1-35 is matched with the corresponding bit of the C(SI) 0-35. The C(Y) are unchanged. When the corresponding bit of either location Y or the SI, or both, is a 1, a 1 replaces the contents of that position of the SI. When the corresponding bit of both Y and SI is a 0, a 0 replaces the contents of that SI position.

Control Instructions

Instructions that govern the flow of a program and, in particular, those that cause an alteration in the computer's normal process of taking its instructions from sequential core storage locations are called control instructions.

Unconditional transfer instructions specify the location Y from which the computer is to take the next instruction. Conditional transfer instructions also specify a location Y; whether the computer takes its next instruction from location Y or the next sequential location depends on the outcome of a test of some kind. This test is specified by the operation code of the instruction.

Test instructions are similar to conditional control instructions in that they cause some test to be performed. Unlike conditional transfer instructions, however, test instructions do not specify a location Y to which control may be transferred. Instead, the alternative location to which control may be transferred is fixed relative to the location of the test instruction.

Transfer -- TRA Y, T

This instruction causes the computer to take its next instruction from the location specified by Y and proceed from there.

Halt and Transfer -- HTR Y, T

This instruction causes the computer to halt. The instruction counter contains the location of the HTR instruction. Depression of the start key on the operator's console causes the computer to transfer to location Y and execute that instruction.

No Operation -- NOP

This instruction causes the computer to take the next instruction in sequence.

Execute --XEC Y, T

This instruction causes the computer to perform or "execute" the instruction at location Y.

Since the instruction counter is not altered (when Y contains any instruction other than a successful transfer or test instruction), the program advances to the next sequential instruction following the execute instruction after performing the instruction at location Y.

If location Y contains a transfer instruction, it will be executed and program control will be altered from the sequential process.

If location Y contains a test instruction, the instruction following XEC will be located relative to the XEC rather than the test instruction.

Arithmetic Control Instructions

Low Order Bit Test--LBT

If the C(AC) 35 is a 1, the computer skips the next instruction and proceeds from there; if 35 is a 0, the next instruction is taken.

P Bit Test--PBT

If the C(AC) P is a 1, the computer skips the next instruction and proceeds from there; if P is a 0, the next instruction is taken.

Transfer on Plus -- TPL Y, T

If the sign position of the AC is a zero, the computer takes its next instruction from the location specified by Y and proceeds from there. If the sign position is a one, the computer takes the next sequential instruction.

Transfer on Zero -- TZE Y, T

If the C(AC)Q, P, 1-35 are zero, the computer takes its next instruction from the location specified by Y. If they are not zero, the computer takes the next sequential instruction.

Transfer on Minus -- TMI Y, T

If the sign position of the AC is negative (1 bit), the computer takes its next instruction from the location specified by Y and proceeds from there. If the sign position is positive (0 bit), the computer takes the next sequential instruction.

Transfer on MQ Plus -- TQP T, Y

If the sign position of the MQ is plus, the computer takes its next instruction from location Y. If the sign position is minus, the computer takes the next sequential instruction.

Transfer on No Zero -- TNZ Y, T

If the C(AC) Q, P, 1 - 35 are not zero, the computer takes its next instruction from location Y and proceeds from there. If they are zero, the next sequential instruction is taken.

Transfer on Overflow -- TOV Y, T

If the AC overflow indicator is on, it is turned off and the computer takes its next instruction from the location specified by Y. If the indicator is off, the computer takes the next sequential instruction.

Transfer on No Overflow -- TNO Y, T

If the AC overflow indicator is off, the computer takes its next instruction from location Y. If the indicator is on, it is turned off and the computer takes the next sequential instruction.

Storage Testing Control Instructions

Logical Compare Accumulator with Storage -- LAS Y, T

The C(AC) Q, P, 1-35 are treated as an unsigned 37-bit number and are compared with the C(Y)S, 1-35, which are treated as a 36-bit unsigned number. If the C(AC) are greater than the C(Y), the computer takes the next sequential instruction. If the C(AC) are equal to the C(Y), the computer skips the next instruction and proceeds from there. If the C(AC) are less than the C(Y), the computer skips the next two instructions and proceeds from there.

Compare Accumulator with Storage -- CAS Y, T

If the C(AC) are algebraically greater than the C(Y), the computer takes the next sequential instruction. If the C(AC) are algebraically equal to the C(Y), the computer skips the next instruction and proceeds from there. If the C(AC) are algebraically less than the C(Y), the computer skips the next two instructions and proceeds from there. A plus zero is considered greater than a minus zero. NOTE: The comparison is made on all positions of the AC (including positions P and Q) and the contents of location Y.

Storage not Zero Test -- NZT Y, T

If the C(Y) 1-35 are not 0, the computer skips the next instruction and proceeds from there. If the C(Y) 1-35 are 0, the computer takes the next sequential instruction. The C(Y) are not changed.

Storage Zero Test -- ZET Y, T

If the C(Y) 1-35 are 0, the computer skips the next instruction and proceeds from there. If the C(Y) 1-35 are not zero, the computer takes the next sequential instruction. The C(Y) are not changed.

Index Register Testing Instructions

Transfer and Set Index -- TSX Y, T

The 2's complement of the location of the TSX is placed in the specified XR. The computer takes its next instruction from location Y.

This instruction may be used to set up a return address to the main program when it is necessary to transfer to a subroutine and, after finishing with the subroutine, to come back to the main program. For example, assume that arithmetic operations are tested for error conditions and, when these conditions are found, a transfer to a fixup routine is to be executed. The last instruction of the fixup routine could be a TRA 00001 instruction tagged for the same index register as used by the TSX instruction. If the TSX is located at core location 1000 and program return to location 1001 is required, the program could be as shown in Figure 40.

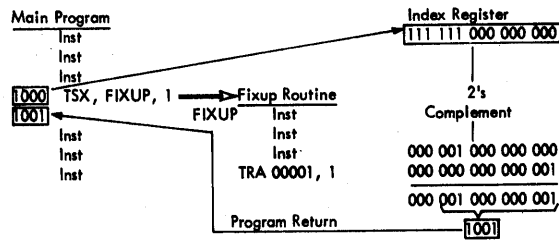


Figure 40. Possible Use Of The TSX Instruction

Transfer On Index -- TIX Y, T, V

If the C(XR) specified by T are greater than V, the C(XR) are reduced by V and the computer takes its next instruction from Y. If the C(XR) are less than or equal to V, the C(XR) are unchanged and the next sequential instruction is taken. With a tag of zero, no transfer occurs.

Transfer On No Index -- TNX Y, T, V

If the C(XR) specified by T are greater than V, the C(XR) are reduced by V and the next sequential instruction is taken. If the C(XR) are less than or equal to V, no reduction is made but the computer transfers to location Y. With a tag of zero, a transfer occurs.

Transfer On Index High -- TXH Y, T, V

If the C(XR) specified by T are greater than V, the next instruction is taken from Y. If the C(XR) are less than or equal to V, the next sequential instruction is taken. With a tag of zero, no transfer occurs.

Transfer With Index Incremented -- TXI Y, T, V

The decrement (V) is added to the C(XR) and replaces the C(XR). The next instruction is then taken from location Y.

Transfer on Index Low Or Equal -- TXL Y, T, V

If the C(XR) specified by T are less than or equal to V, the next instruction is taken from location Y. If the C(XR) are greater than V, the next sequential instruction is taken. With a tag of zero, a transfer occurs.

Figure 41 summarizes the indexing instructions by type and action.

Actions	Conditions	
	If $C(XR) > V$	If $C(XR) \leq V$
Test and Modify: TIX TNX	$C(XR) = XR - V$ and transfer to Y $C(XR) = XR - V$ and take next inst.	Take next instruction Transfer to Y
Test Only: TXL TXH	Take next instruction Transfer to Y	Transfer to Y Take next instruction
Modify Only: TXI	The V are added to the C(XR) and replace the C(XR) Transfer to Y for next instruction	

Figure 41. Index Transfer Instruction Summary

Sense Light and Indicator Testing Instructions

Sense Lights On -- SLN Y

The address part of this instruction (Y) determines which of the four sense lights are to be turned on. The lights are addressed as: 1, 2, 3, or 4.

Sense Light Test -- SLT Y

If the corresponding sense light is on, the light is turned off and the computer skips the next instruction and proceeds from there. If the light is off, the computer takes the next sequential instruction.

Transfer when Indicators On -- TIO Y, T

For each bit in the C(AC)P, 1-35 that is a 1, the corresponding position of the C(SI) 0-35 is examined. If all the examined positions in the SI contain a 1, the computer takes its next instruction from location Y. If any of the examined positions is not a 1, the computer takes the next sequential instruction. If the C(AC) P, 1-35 are zero, the computer takes its next instruction from location Y. The C(AC) and C(SI) are unchanged.

Transfer when Indicators Off -- TIF Y, T

For each bit of the C(AC) P, 1-35 that is a 1, the corresponding position of the C(SI) 0-35 is examined. If all examined positions contain 0's, the computer takes its next instruction from location Y. If any of the examined

positions does not contain a 0, the computer takes the next sequential instruction. If the C(AC) P, 1-35 are zero, the computer takes its next instruction from location Y. The C(AC) and C(SI) are unchanged.

Instructions used With Trapping

The following instructions are used with the trap feature. A complete description of trapping is found under "Data Channel Trapping".

Enter Trapping Mode --- ETM

This instruction causes the computer to enter the transfer trapping mode. A trapping indicator on the operator's console is turned on.

When the computer is in the trapping mode and any transfer except a trap transfer (TTR) is executed, the location of the transfer instruction replaces the address part of location 00000 whether the condition for transferring is met or not. If the transfer condition is met, the computer takes its next instruction from location 00001 and proceeds from there.

Only instructions which have "transfer" in their title are affected by the transfer trapping mode. Address modification may change the operation since positions 23-35 of the ETM instruction are a part of the operation code.

Leave Trapping Mode --- LTM

This instruction turns off the trap mode indicator and causes the computer to leave the transfer trapping mode. Transfer instruction, therefore, will not be trapped again until an ETM instruction is executed.

The computer leaves the trap mode until the LTM instruction is executed or the clear or reset keys on the operator's console are depressed. Since positions 23-35 of the LTM are a part of the operation code, any modification by an index register may result in the changing of the operation itself.

Trap Transfer --- TTR Y, T

This instruction causes the computer to take its next instruction from location Y and to proceed from there whether in the transfer trap mode or not. This makes it possible to have an unconditional transfer in the transfer trapping mode.

Restore Channel Traps --- RCT

This instruction will allow traps to occur as specified by the previous enable instruction. The RCT cancels the inhibiting effect of an executed trap. Since the address part of the RCT is a part of the operation code, modification by an index register may change the operation itself.

Enable from Y --- ENB Y, T

When this instruction is executed, the contents of location Y determine which signals may cause a trapping operation. Execution of each ENB cancels the effect of previous ENB instructions. All channels may be disabled (traps will not occur) by executing an ENB whose operand contains all zeros.

Execution of a trap will inhibit all further traps until a new ENB is executed or a RCT is executed. Depression of the reset or clear key on the operators console will also disable all channels.

Trapping signals are controlled as follows:

<u>Signal Due to</u>	<u>Channel</u>	<u>Effective if a 1-bit in position</u>
Channel command or EOF	A	35
Channel command or EOF	B	34
Channel command or EOF	C	33
Channel command or EOF	D	32
Channel command or EOF	E	31
Channel command or EOF	F	30
Channel command or EOF	G	29
Channel command or EOF	H	28
Tape check	A	17
Tape check	B	16
Tape check	C	15
Tape check	D	14
Tape check	E	13
Tape check	F	12
Tape check	G	11
Tape check	H	10

INPUT/OUTPUT DEVICES AND OPERATIONS

INTRODUCTION

To use the fast, versatile processing ability of the processing unit, the user must be able to put raw data into the computer system, tell the system what to do, and take the processed data from the system and record it in a form that can be used. This chain of input, processing and output always begins and ends with some input/output device.

An input/output device is a machine linked in directly to the data processing system. Each device operates under control of the processing unit as directed by the stored program. A data channel unit, placed between the processing unit and the input/output device, serves as a control or synchronizer unit. The channel not only controls the devices attached to it but serves as an assembly-disassembly device for the data passing through it. This is necessary since the internal processing speeds are much faster than the input devices that read data or the output devices that record the results. Figure 42 shows the relationship between processing unit, channel and input/output devices.

Input devices sense or read data from IBM cards, magnetic tape, paper tape, or may simply supply data to the processing unit in the form of electronic pulses. The data are then placed in the core storage of the system. Output devices record or write the data from storage on IBM cards, magnetic and paper tape, or prepare printed copy. Output may also be in the form of electronic pulses (for transmission over communications networks).

Reading and Writing

Reading takes place as the input medium physically moves through an input device. The information is sensed or read and is converted to a form that may be used by the computer system. The information is then sent to core storage.

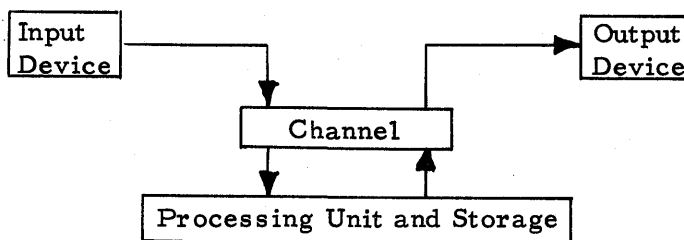


Figure 42. Input-Output Device Relationship

Writing involves converting data from storage to a form or language compatible with an output medium and recording the data using an output device.

Most input/output devices are automatic; once started, they continue to operate as directed by the stored program. Instructions in the program select the required device, direct it to read or write, and indicate the storage location that data will be put into or taken from. A few input devices are manually operated, and no medium for recording data is involved. Instead, data are entered directly into the computer using a keyboard or switches, which are usually a part of an operator's console.

Data Buffering

All data processing procedures involve input, processing, and output. Each phase of the procedure takes a specific time. The usefulness of a computer is often directly related to the speed at which it can complete a given procedure. Ideally, the configuration and speed of the various input/output devices should be so arranged that the processing unit is always kept busy with useful work. The efficiency of any computer system can be increased to the degree in which input, output, and internal processing can be overlapped or allowed to occur simultaneously.

Figure 43 shows the basic time relationship between input, processing, and output with no overlap of operations. In this type of data flow, processing is suspended during reading or writing operations.

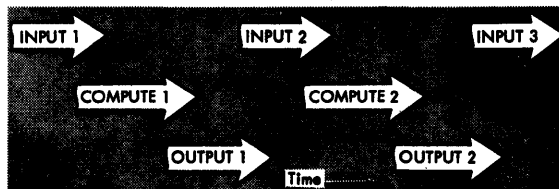


Figure 43. Non-Overlap Operation

Figure 44 shows an overlapped time relationship. The figure assumes that there are two buffers, one for input and another for output. With this type of data buffering system, data are first collected in an input buffer. When called for by the program, the contents of the input buffer are sent to core storage. The transfer takes only a fraction of the time that would be required to read the data directly from an input device. Also, while data are being assembled in the buffer, processing can occur in the processing unit. Likewise, complete data from storage can be placed in an output buffer at high speed. The output device is then instructed to write out the contents of this buffer. While writing occurs, the processing unit

is free to continue with other work.

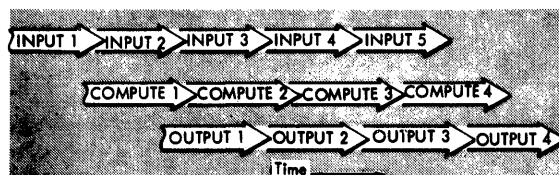


Figure 44. Overlap Operation

Data Channel Operation

Data being transmitted between core storage and any input-output device must pass through a data channel. The operation of a data channel is initiated by the execution of two instructions in the central processing unit. Once started, the channel operates independently of the main program being executed by the CPU. A data channel has the responsibility for controlling the quantity and destination of all data transmitted between core storage and the input output device. It also performs limited counting and testing operations concerned with the transmission of data.

The IBM 7909 Data Channel is also able to instruct and select an input-output device adapter, such as the IBM 7631 File Control or the IBM 7640 Hypertape Control.

Programs for a channel operation are stored in core storage just as are instructions executed by the CPU. To distinguish between CPU and data channel programs, data executed in the CPU are termed instructions, data executed by the data channel are termed commands, and data executed by the adapter are termed orders.

Although a channel, once started, operates asynchronously, the main program may exercise a large degree of supervisory control through instructions which test the status of a data channel. A single command may transmit a large block of words between core storage and an input-output device so that normally many instructions in the main program may be executed during the time taken to execute just one command in a data channel.

All transmission is in 36-bit word parallel fashion. Since the CPU and a data channel cannot take a storage reference cycle at the same time, the execution of an instruction in the main program may be delayed at least one computer cycle. Once such a delay occurs, all of the time

needs of all data channels will be satisfied before the main program execution is resumed. Such delays are imposed automatically and do not interfere with the internal registers or calculations in the CPU. If the instruction being executed is not using core storage when a channel requires a storage cycle, normally no delay is occasioned. When several data channels require storage cycles at one time, the sequence of transmission is handled automatically.

A data channel controls all input-output units attached to it in much the same way. Because of the importance of magnetic tape, the relationship between a data channel and tape is discussed here.

An input/output operation is started by execution of a select instruction, which is decoded by the processing unit. The output of the decoders is sent to the channel to select the input/output device specified by the select instruction. If the device is busy with other work or is not ready for operation, the select waits until the device is free.

Upon execution of the select, the input/output device called into use actually starts moving. When selection is successfully completed, the channel ends operation on the select and the processing unit gets the next instruction for execution. This instruction is normally a reset and load channel (RCH) instruction. The RCH specifies a storage location that contains the data channel command for this input/output operation. One RCH exists for each data channel.

Understanding the relationship between the select and the RCH instructions is very important. The speed of input/output devices is much slower than processing unit execution speeds; however, the program must have executed a RCH instruction by the time the particular device being used is ready to read or write. This is shown in Figure 45, using a read select instruction for a 729II magnetic Tape Unit. A maximum of 4 milliseconds may exist in the program between execution of the RDS to a 729 II tape unit and execution of the RCH. The RCH may, however, immediately follow the RDS or be placed anywhere within the time allowed. To further point out the time relationship; the 4 milliseconds needed to get the tape unit moving and actually reading data into the processing unit is enough time for execution of 1800 machine cycles on a 7090 system or about the time required to execute 900 average-time instructions.

Operation	Comments
RDS 729II Tape	Execution of the RDS starts physical motion in the selected tape unit.
RCH	Four (4) milliseconds after the RDS is executed, the RCH must be executed; otherwise, the tape unit is disconnected from the read operation.

Figure 45. Time Relationship Between Instructions

Execution of the reset and load channel instruction places the data channel command in the Channel Registers. The command (Figure 46) specifies the number of words to be moved (word count-positions 3-17) and the first storage address (start address-positions 21-35) to be used for the read or write operation. Read operations are defined as input; write operations are defined as output.

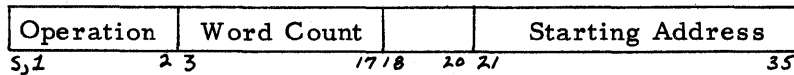


Figure 46. Data Channel Command Format

The word count portion of the command is tested for zero count and, when it is zero, the channel ends operation on the command and disconnects the input/output device. Disconnect means that the input/output device is no longer needed (has finished its operation) and may be released for further use.

Magnetic Tape and Data Channel (7607) Addresses

A maximum of ten tapes per channel may be used with the 7090 system. Like locations in core storage, each tape unit has an address. Each data channel also has an address. The combination of the two addresses will then specify a particular tape unit attached to a particular data channel.

To start a tape unit for reading or writing, a SELECT instruction must be executed in the main program. The instruction READ SELECT prepares the tape for a reading operation and the instruction WRITE SELECT prepares it for writing. The joint address of a tape unit and data channel occupies the address part (positions 21-35) of the select instruction. If the address field of this instruction is viewed as a five-digit octal number, then the three low-order digits specify the tape unit, and the fourth and fifth digits the data channel (Figure 47).

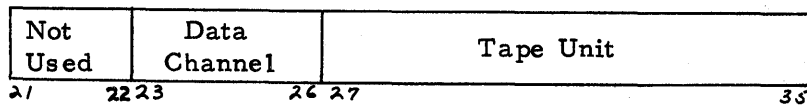


Figure 47. Select Instruction Address Field

The specific numerical addressing system used by the tapes is as follows:

1. Data channels A through H are identified by the octal numbers 1 through 10 and occupy positions 23-26 of the instruction.

2. In the BCD mode, tape units 1-10 are identified by the octal numbers 201-212, occupying positions 27-35 of the instruction.
3. In the binary mode, the tape units are identified by the octal numbers 221-232 which occupy positions 27-35 of the instruction.

Examples: If the instruction READ SELECT 1201 is executed, tape unit 1 attached to the data channel A will be selected and started for a reading operation in the BCD mode. If the instruction WRITE SELECT 3223 is given, tape unit 3 attached to data channel C will be selected and started for a write operation in the binary mode.

Data Channel Registers (7607)

Once the select instruction has been executed, the operation of the tape and transmission of data between core storage and tape are under control of the data channel. There are four registers in the channel which control its operation. These registers are similar in function to the control registers in the CPU.

The first command of a data channel program must be sent to the channel by a RESET AND LOAD CHANNEL instruction specifies the location in core storage containing the data channel command. When this instruction is executed, the contents of the location in core storage (specified by the RESET AND LOAD CHANNEL instruction) are sent to the data channel control registers.

There is a separate RESET AND LOAD CHANNEL instruction for each data channel. Where select instructions specify the appropriate data channel through usage of part of their address field, the address field of the reset and load channel is fully occupied by the storage address of the command. Thus, the distinction between data channels is made in the operation part of the instruction.

Word Count Register. The contents of positions 3-17 of the data channel command are loaded into this register (Figure 48). This register specifies the number of words to be transmitted between core storage and the input-output unit. As each word is entered or taken from core storage, the contents of the word register are reduced by one.

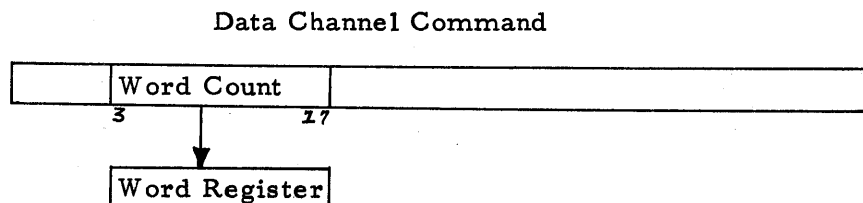


Figure 48. Data Channel Word Count Register

Channel Address Register. The contents of positions 21-35 of the data channel command are loaded into this register (Figure 49). The register specifies the location in core storage from which the data are to be taken during writing or to be entered into during reading. The contents of this register are increased by one after each word transmission to or from core storage. Thus, the address register directs the transmission of data into or from consecutive locations in core storage.

Data Channel Command

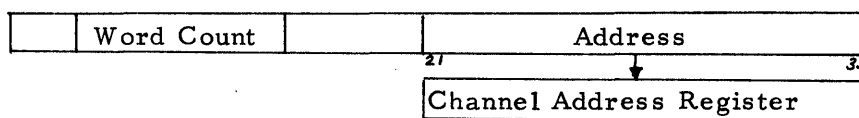


Figure 49. Data Channel Address Register

Location Register. This register is similar to the instruction counter in the CPU. The location register contains the location of the current data channel command, plus one. Thus, data channel commands are taken normally from sequential locations in core storage. Just as control or transfer instructions alter the contents of the CPU's instruction counter, transfer commands change the contents of the location register in a data channel.

The size of these registers (word register, channel address register, and location register) is set a 15-bit length. Each register has a capacity large enough to hold the address of the last location in core storage. When a 15-bit command field is loaded into a channel register, the leftmost bits which exceed the capacity of the register are ignored. When the contents of a channel register are stored in a 15-bit field in core storage, the leftmost bits corresponding to the absent bits of the register are set to zeros.

With reference to blocks of consecutively located commands or data words, the highest location in core storage and location zero are treated as consecutive locations.

Operation Register. This register is similar to the instruction register in the CPU. The contents of positions S, 1, 2, and 19 of a data channel command are loaded into this register (Figure 50). Bit positions S, 1, and 2 provide for eight possible data channel operations. Each of the eight commands may accomplish either reading or writing. Position 19 is used only for reading operation; it has no effect on writing operations. When this position contains a 1, all functions proceed normally except that no transmission of data to core storage occurs. Thus a command with position 19 containing a 1 may be used to skip over a number of words (determined by the word count) while reading from an input device. When position 19 contains a 1, the data channel involved is said to be operating in the non-transmitting mode.

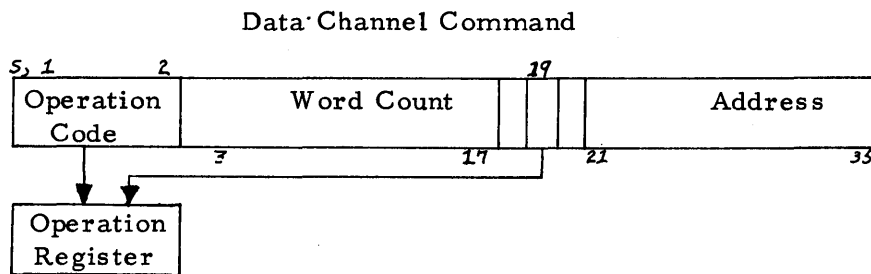


Figure 50. Data Channel Operation Register

Data Register. This 36-bit register serves as a buffer between core storage and an input-output device.

Calc Exit-Entry. To punch or print the 72 possible positions for a row of information, two separate words must be brought from core storage. These words are stored in the 72-position exit-entry register because all punch or print magnets must be energized at about the same time. At the proper time, all 72 positions are sent to the output device being used.

Tape Register. Tape words coming from the data register are transmitted in parallel. Likewise, words going to the data register from the tape register are sent in parallel. Words going to and from the tape unit must, however, go in six-bit groups. In writing a word on tape, the word is first sent from the data register to the tape register and is then sent to the tape unit, six bits at a time. After the first six-bit group has been sent to the tape unit, the contents of the tape register are shifted six positions to the left. This process is repeated six times until the entire word has been written on tape. In the case of reading, groups enter from the right of the tape register and are shifted left until a full word has been assembled.

Data Channel Entry Keys. These keys are similar in action to the console entry keys on the CPU. There is one set on each data channel console. Data flow for the 7607 is shown in Figure 52.

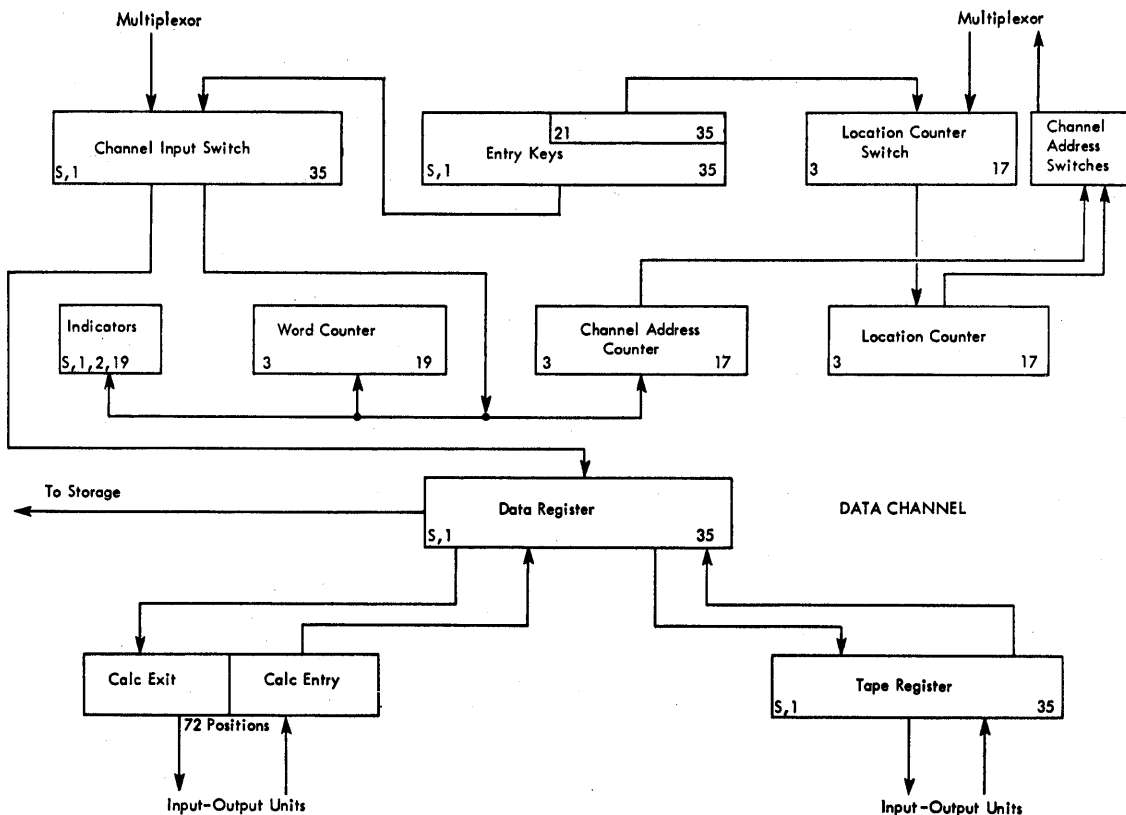


Figure 52. 7607 Data Flow

Data Channel Register Example. An example is shown in Figure 53. If core storage location 1546 contains the data channel command, represented here as an octal number, 000124002117, and the instruction reset and load channel A with an address of 1546 is executed, then zeros will be placed in the operation register, 00124 will be placed in the word register, 2117 will be placed in the channel address register, and the location register will contain 1547.

Once a tape unit has been started by a select instruction, the data channel must receive its first command within a definite time period. Thus the reset and load channel must be executed by the main program within this allotted time period following the select order. Specific tape timings are given in the magnetic tape section of "input-Output Components."

If the reset and load channel is not executed within the allotted time period, the tape unit is logically disconnected from the calculator and no word transmission will occur. If a reset and load channel is given at any time when an input-output device is not logically connected to the data

channel, the instruction is executed in the CPU but a special indicator, called the input-output check indicator, is turned on. The status of this indicator may be tested by the stored program.

Exact description, together with examples, of the data channel commands is found in the "Computer Instruction" section of this manual.

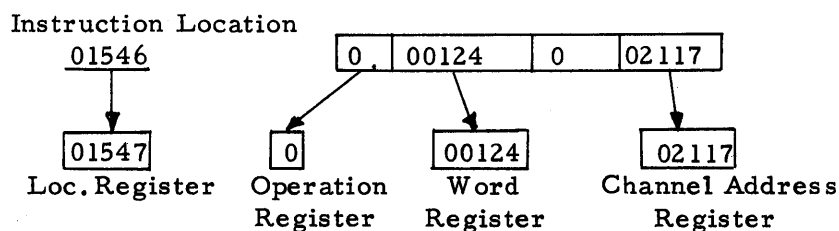


Figure 53. Data Channel Register Example

IBM 7607 DATA CHANNELS

A maximum of eight IBM 7607 Data Channels may be used with the 7090 system. As many as ten IBM 729 II or 729 IV Magnetic Tape Units, intermixed in any fashion, may be attached to each 7607 Data Channel (except 7607 Model 5) in a 7090 System.

Magnetic Tape Units

As many as ten IBM 729 II, IV, V, or VI Magnetic Units, intermixed in any fashion, may be attached to each 7607 Model 3 or 4 in the 7090 system.

Data may be read or written in the BCD or binary modes in one of two character Density rates. Effective character rates (both low and high density) and physical tape speeds for each tape unit are shown in Figure 54.

Tape Unit	Characters per Second			Tape Speed (inches per second)
	200 CPI	556 CPI	800 CPI	
729II	15,000	41,667		75
729V	15,000	41,667	60,000	75
729IV	22,500	62,500		112.5
729VI	22,500	62,500	90,000	

Figure 54. Tape Characteristics

Character Alteration in BCD Mode

As six-bit BCD characters are read from magnetic tape, the zone bits of

some of the characters are altered. This alteration is performed so that the digits 0-9 and the characters A-Z are represented in core storage by six-bit binary numbers of increasing magnitude. The alteration of these zone bits is:

<u>Characters</u>	<u>In Core Storage</u>	<u>On Tape</u>
	<u>BA</u>	<u>BA</u>
Numeric	00	00
A to I	01	11
J to R	10	10
S to Z	11	01

The digits 1 through 9 are represented by the six-bit binary numbers 000001 through 001001; that is, by their exact values as binary integers. Thus, the zone part of the digits is 00. The number zero is represented on magnetic tape by the bit configuration 001010. This representation is automatically altered to 000000 during reading in the BCD mode.

During writing in the BCD mode, the alteration procedure is reversed so that the storage BCD characters are transformed to the BCD tape format by the tape control. In writing, the tape control automatically performs the modifications described but does not check whether the six bits being transmitted form a legitimate BCD character. Thus, if binary numbers are written in the BCD mode, both a pure zero and the number 10 (001010) are recorded on the tape as the BCD zero character (001010). Also, the integer 15 (001111) is identical to the BCD tape mark, signifying an end-of-file condition. Therefore, random binary data should not be recorded in the BCD mode.

In addition to alphabetic and numeric characters, the BCD format provides for punctuation marks and other special characters. Included is the BCD character "blank" which suppresses printing or punching in any desired position during auxiliary operations.

Data Channel Select Registers

When a select instruction addressing a tape, card, or printer unit is interpreted in the CPU, it is sent to the specified data channel for execution.

If the select instruction is a data select (read select or write select) the instruction is placed in the channel's data select and unit registers. The operation to be performed and the type of unit involved is placed in the data select register. The unit number is placed in the unit register (Figure 55). Once the specified input-output unit has been selected, the unit register is free to accept new information sent from the main program. The data select register, however, is used throughout the entire input-output operation to control the data channel reading or writing activity. This register cannot accept another data select operation until the present operation has been terminated.

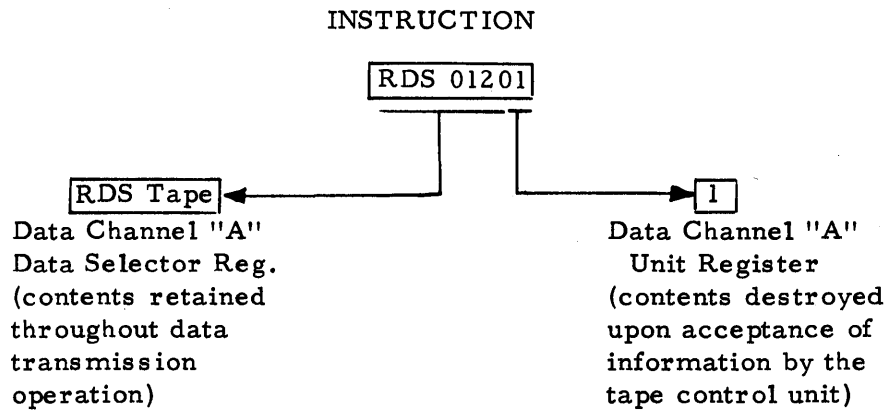


Figure 55. Data Channel Command Stacking

If the select instruction is a non-data select (backspace, rewind, or write-end-of-file instruction) the instruction is placed in the specified data channel's non-data and unit registers (Figure 56). The unit register has the same function as described for a data select. When the data channel executes a backspace record (BSR), backspace file (BSF), or a rewind (REW) instruction, the non-data and unit registers are used only until the tape units have been selected. This requires only a few microseconds. After this selection, the data channel is no longer required and is free to accept any select instruction sent from the CPU. In the case of a BSR or BSF, the execution is completed by the multiplexor. Any select instruction, addressing a card or printer unit and sent to the data channel before completion of a BSR or BSF, will be executed immediately.

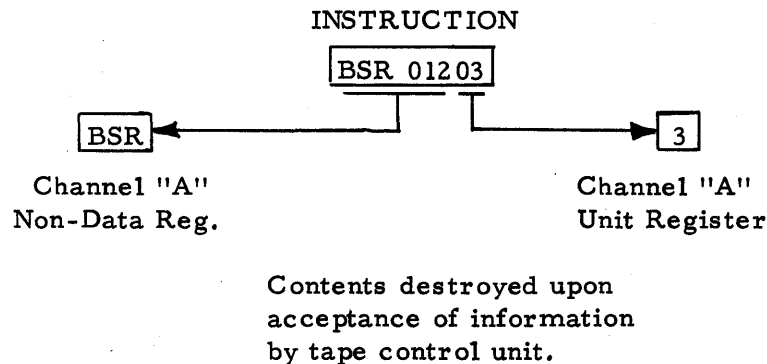


Figure 56. Data Channel Command Stacking

The REW operation differs from BSR and BSF in that it uses the tape control for a few milliseconds only. The operation is then controlled by the tape unit that is being rewound. Thus, once a REW has been started, any select instruction which does not address the tape unit being rewound

will be executed immediately.

The WEF instruction is similar to a data select in that the specified data channel's non-data register is in use throughout its entire execution.

The data select, non-data, and unit registers of a channel are not directly addressable by the main program. The registers are of importance, however, in terms of synchronous timing relations between the main program and the data channel operation. For example, the outcome of the test made by a transfer-in-operation or transfer-not-in-operation instruction depends solely on the status of the channel's select registers.

Card Reader

One IBM 711 Card Reader may be attached to any channel on the system. With a 711 reader attached, a 716 Printer must also be attached to the same channel because power is supplied to both the card punch and reader from the printer.

The 711 reads cards at a rate of 250 or 500 cards per minute. Cards to be read are placed in the feed hopper face down, 9-edge first. Information punched in the cards may be decimal, alphabetic, binary, or any other character code. The reading format is controlled by the stored program and a control panel located on the reader.

Card Punch

One IBM 721 Card Punch may be attached to any channel on the system. Channels having an attached punch must also have an attached printer.

Cards to be punched are placed in the punch feed hopper face down, 9-edge first and are punched at 100 cards per minute. Punched card output may be in decimal, alphabetic, binary, or any other character code. The punching format is controlled by the stored program and a control panel on the punch.

Basic operations are analogous to those of the reader, except that instead of a card image in being built up in core storage with information read from a card, the channel sends a card image from storage to the punch to be recorded on a card.

Printer

One IBM 716 Printer may be attached to any channel on the system. The printer is equipped with 120 rotary type wheels, each wheel has 48 characters including numerals, alphabetic symbols, and special characters. Use of the stored program enables the computer to print any desired information in any form convenient to the programmer. This information is printed 150 lines per minute.

Printing format is controlled by the information itself and a control panel located on the printer.

A line is printed in a time period which is called a print cycle. During this interval the type wheel is rotated until the specified character is centered in front of the platen and then printed. The amount of rotation, and consequently the character printed, depends on the time in the print cycle when the electrical impulse initiates motion. For example, if a print wheel receives an electrical impulse during that part of the cycle designated as 9 time, the number 9 is printed. Also if the print wheel receives an impulse at 1 time and an impulse at 12 time, the print unit positions the wheel to print a letter A.

Printing is similar to tape and card operations with respect to the role played by the channel to which the printer is attached. A printer record corresponds to a single printed line or to the information in core storage necessary to print one line. An end-of-record condition occurs at the end of each print cycle. Consequently, all eight channel commands are available for use in a print program.

INPUT-OUTPUT TRANSMISSION OPERATIONS

Instructions that either send commands to a data channel or store information from data channel registers are classified as input-output transmission operations. These instructions are described in groups of eight. All eight instructions within a group are identical in function and differ only in that each refers specifically to one of the eight possible data channels. The instructions will be described for data channel A.

Store Channel A -- SCHA Y, T

This instruction replaces the c(y) with the contents of the channel A address location, and operation registers. If channel A is not attached to the computer when the SCHA is given, the c(y) are cleared by the SCHA.

The c(y) 21-35 are replaced by the c(ar), and c(y) 3-17 are replaced by the c(lr), and the c(y) 8, 1, 2, 19 are replaced by the contents of the operation register. An SCHA instruction may be executed at any time, regardless of whether or not the specified channel is in operation. If the channel is in operation and the channel registers are in the process of being changed, the execution of the SCHA will be delayed until the change has been completed. Note that the c(ar) will be one greater than the storage location of the last word involved in data transmission and that the c(lr) are one greater than the storage location from which the current command was taken. A channel relinquishes priority between the time the last word is transmitted by an IOCP or IOSF command to the arrival of a subsequent channel command. Therefore, an SCH may store an address which is one greater than the address of the last word transmitted by the IOCP or IOSF commands (explained later in text).

<u>INSTRUCTION</u>	<u>CODE</u>	<u>NAME</u>
SCHB	-0640	Store Channel B
SCHC	0641	Store Channel C
SCHD	-0641	Store Channel D
SCHE	0642	Store Channel E
SCHF	-0642	Store Channel F
SCHG	0643	Store Channel G
SCHH	-0643	Store Channel H

Reset and Load Channel A-- RCHA Y, T

If channel A has been selected by either an RDS or WRS, the c(y) S, 1, 2, 19 replaces the channel operation register, c(y) 3-17 replace the c(wr) and the c(y) 21-35 replace the c(ar). In addition, the number Y plus one replaces the c(lr).

If channel A is not selected when the RCHA is given, the RCHA executes normally but the I-O indicator is turned on (If the command loaded by the RCHA specifies indirect addressing, it will not occur). For each RDS or WRS, the corresponding RCHA must be given if any transmission between storage and the selected I-O device is to take place. If a second RCHA is given at a later time, the order is executed immediately. No other select instruction should be inserted between the RCH and its associated WRS or RDS.

<u>INSTRUCTION</u>	<u>CODE</u>	<u>NAME</u>
RCHB	-0540	Reset and Load Channel B
RCHC	0541	Reset and Load Channel C
RCHD	-0541	Reset and Load Channel D
RCHE	0542	Reset and Load Channel E
RCHF	-0542	Reset and Load Channel F
RCHG	0543	Reset and Load Channel G
RCHH	-0543	Reset and Load Channel H

Load Channel A -- LCHA Y, T

If the data channel has been selected, the computer delays until an IOCT, IORT, or IOST command is processed for channel A or the channel leaves operation. After an IOCT, IORT, or IOST command has been executed by the channel A, the LCHA is executed as shown below.

The c(y)s, 1, 2, 19 replace the contents of channel A operation register. C(y) 3-17 replace the c(wr), the c(y) 21-35 replace the c(ar), and the number Y plus one replaces the c(lr). If an LCHA is issued and either (1) the channel is not selected, or (2) channel A is selected but an IOCT, IORT, or IOST command is not executed before the channel disconnects, the I-O check indicator is turned on and the LCHA is treated as a no-operation.

<u>INSTRUCTION</u>	<u>CODE</u>	<u>NAME</u>
LCHB	-0544	Load Channel B
LCHC	0545	Load Channel C
LCHD	-0545	Load Channel D
LCHE	0546	Load Channel E
LCHF	-0546	Load Channel F
LCHG	0547	Load Channel G
LCHH	-0547	Load Channel H

Read Select--RDS

Basically, the read select instruction simply gets the selected I/O unit started. The programming system mnemonics take care of the addressing of the unit. Mnemonics and associated units are:

RCDx	Read card reader, channel x
RPRx	Read printer, channel x
RTBx	Read tape binary, channel x
RTDx	Read tape decimal, channel x

Write Select -- WRS

As with the read select, the write select instruction gets the I/O unit started and the programming system mnemonics take care of the addressing. Mnemonics and associated units are:

WPBA	
WPBx	Write printer binary, channel x
WPDx	Write printer decimal, channel x
WPUx	Write punch, channel x
WTBx	Write tape binary, channel x
WTDx	Write tape decimal, channel x

DATA CHANNEL COMMANDS

The eight types of data channel commands are described in much the same manner as other computer instructions. The following steps list command conditions:

1. The letter Y in the address part (21-35) of a command is used to denote a core storage location.
2. The letter C in the decrement part (3-17) of a command is used to denote a word count amount.
3. The numerical operation code is shown by an octal digit in the prefix part (S, 1 and 2). The digit may be visually converted to

its binary equivalent for reference to the bit pattern actually used.

4. Indirect addressing of data channel commands is possible on the 7090 system. Position 18 of the command contains the flag bit. Thus, with a command having an address part (Y) and a one in position 18, the address part of location Y replaces the address part of the command before it is executed. With a zero word count, indirect addressing does not occur on IOCP or IOSP commands. Indirect addressing will occur only on read or write operations.
5. Separate commands have not been formulated to handle bit position 19. Instead, a fifth character (N--denoting non-transmit) is appended to the mnemonic codes used for position S, 1, and 2. This type of command is used for read operations only.
6. Seven of the eight commands deal with data transmission. The codes for these commands all contain the letters "I-O".
7. The eighth command is a transfer in channel command.
8. The word disconnect means that the units involved are separated logically rather than physically. When a disconnect is signalled, it may be delayed depending on the operation being performed.

In execution of the Input-Output commands, magnetic tape is assumed to be the I-O device used. However, mechanical motion on the tape (from record to record) may be compared to card equipment (from card to card or from line to line) on the printer. The descriptions may then be applied to input-output devices other than magnetic tape.

Input-Output under Count Control and Disconnect---IOCD Y

C words are transmitted between an input-output device and core storage beginning with location Y. The data transmission is under control of the count (C) field only.

Read Operation. If the word count has been reduced to zero before a record gap is reached, the channel leaves operation and tape motion will continue until a record gap is reached. No transmission will occur during this time. If a record gap is encountered when the word count is not zero, the gap is ignored and reading continues from the next record. An IOCD command with a 0 word count, loaded at the end of record will disconnect the channel.

Write Operation. When C words have been written on tape, a record gap is written. If this command is given at the beginning of a record and C is initially zero, approximately 3-3/4 inches of blank tape will be written before the record gap is written. The channel is then disconnected.

Input-Output under Count Control and Proceed -- IOCP Y

C words are transmitted between an I-O device and core storage location Y. The data transmission is under control of the count field only. When C is reduced to zero, or is initially zero, the next sequential command is brought into the data channel and executed.

Read Operation. C words are read from tape and stored in consecutive storage locations beginning with location Y. When the word count has been reduced to zero, the channel takes its next command in sequence and executes it.

If the word count is reduced to zero by the last word of a record and the next command is an IORP, IOSP, IORT, or IOST, the present end of record will be recognized. Unlike the IOSP, the IOCP command need not proceed within 11 cycles in order to recognize the present end of record.

Write Operation. C words from storage beginning with location Y are written; the channel proceeds to the next sequential command. An end of record is not written on tape when the word count is reduced to zero.

In printing or the punching of cards, if the owed count is reduced to zero on the 12-row right transmission point, and the next command is an IORP or IORT, the end of the present machine cycle (record) will be recognized.

Input-Output of a Record and Proceed -- IORP Y

C words are transmitted between tape and core storage location Y.

Read Operation. Words are transmitted from tape and stored in consecutive storage locations until either an end of record is encountered or the word count has been reduced to zero. If the word count is reduced to zero (or is initially zero) before an end of record is reached, the rest of the words in that record are skipped without transmission to storage. When the record gap is reached, the channel takes the next sequential command.

Write Operation. When C words have been written, a record gap is written and the channel proceeds to the next sequential command.

Input-Output under Count Control and Transfer -- IOCT Y

C words are transmitted between an I-O device and storage beginning with location Y. The data transmission is under control of the count field only. When C is reduced to zero, the next command is taken from a core location specified by the load channel instruction in the main program or disconnects (and traps if the channel is enabled) if no load channel instruction is waiting. If this command is loaded by an RCG or LCH instruction and C is initially zero, the channel is immediately disconnected unless restricted by the operation.

Read Operation. Execution of the command is under control of the count field only and end of record gaps are ignored. If the word count is reduced to zero by the last word of a record and the next command is an IORP, IOSP, IORT, or IOST, the present end of record is recognized. Unlike the IOST command, the LCH instruction need not load the channel within 11 cycles in order to recognize the present end of record.

Write Operation. An end of record will not be written after C words have been written if the LCH instruction results in bringing into the channel a new word count. In printing or punching, if the word count is reduced to zero on the 12-row right transmission point and the next command is an IORP or IORT, the end of the present machine cycle (record) is recognized.

Input-Output of a Record and Transfer --- IORT Y

Read Operation. Words are transmitted from tape and stored in consecutive storage locations, beginning with location Y, until either an end of record is encountered or the word count has been reduced to zero. If the word count is reduced to zero (or is initially zero) before an end of record is reached, the rest of the words in the record are skipped without transmission to storage. When the record gap is reached, the new command is taken from a core location specified by the LCH instruction in the main program or disconnects (and traps if the channel is enabled) if no LCH is waiting.

Write Operation. When C words have been written, a record gap is written and the channel takes its next command from the core location specified by the LCH instruction in the main program or disconnects (and traps if the channel is enabled) if no LCH is waiting.

Input-Output until Signal, then Proceed --- IOSP Y

Read Operation. Words are read into consecutive storage locations beginning with location Y until either the contents of the word counter are reduced to zero or an end of record is reached. When either event occurs, the channel takes the next sequential command and executes it.

Write Operation. C words are written on tape beginning with storage location Y. When the specified number of words have been written, the channel proceeds to the next sequential command. An end of record is not written on tape when the word count reaches zero. In printing or punching cards, if the word count is reduced to zero on the 12-row right transmission point and the next command is an IORP or IORT, the end of the present machine cycle (record) is recognized.

Input-Output until Signal, then Transfer --- IOST Y

Read Operation. Words are read into consecutive core locations beginning with location Y until either the word counter is reduced to zero or an end of record is reached. When either event occurs, the channel takes its next command from a core location specified by the LCH instruction in the

main program or disconnects (and traps if the channel is enabled) if no LCH is waiting.

Write Operation. C words are written on tape from storage beginning with location Y. When C words have been written, the channel takes its next command from a core location specified by the LCH instruction in the main program or disconnects (and traps if the channel is enabled) if no LCH is waiting to be executed. In printing or punching cards, if the word count is reduced to zero on the 12-row right transmission point and the next command is an IORP or IORT, the end of the present machine cycle (record) is recognized.

Transfer in Channel --- TCH Y

This command is the transfer command for all data channels. When a TCH command is executed, the channel executes the next command specified by location Y of the TCH.

Program Example

The following example illustrates the use of commands to read and skip tape records. (Figure 57).

<u>Location</u>	<u>Instruction/Commands</u>	<u>Comments</u>
500	RDS 01201	Selects tape 1, channel A
501	RCHA 01000	Loads first command
...
01000	IOCP 00006 0 03000	Read first six words
01001	IOCPN 00005 2 00000	Skip next five words
01002	IORP 77777 0 03006	Read rest of the record
01003	IOCD 00000 0 00000	Disconnect.

Figure 57. Program Example

The two instructions (RDS and RCHA) select tape unit 1 on channel A and load the command located at 1000 into the channel's registers. The IOCP reads the first six words of the first record from tape and places them into core locations 3000-3005. The IOCPN skips the next five words in the tape record. The IORP reads the remaining words in the record into locations starting with 3006-300n. When the record gap is sensed, the IOCD command disconnects the channel and the tape unit and thereby ends the operation.

Channel Testing Instructions

Transfer on Channel x in Operation--TCOx Y, T

If channel x is in operation, the computer takes its next instruction from Y. If not in operation, the next sequential instruction is taken. There is a TCO instruction for each channel:

TCOA	+0060	TCOC	+0062	TCOE	+0064	TCOG	+0066
TCOB	+0061	TCOD	+0063	TCOF	+0065	TCOH	+0067

Transfer on Channel x End Of File--TEFx Y, T

If the end of file indicator for channel x is on, it is turned off and the computer takes its next instruction from Y. If the indicator is off, the next sequential instruction is taken. There is a TEF instruction for each channel:

TEFA	+0030	TEFC	+0031	TEFE	+0032	TEFG	+0033
TEFB	-0030	TEFD	-0031	TEFF	-0032	TEFH	-0033

Transfer on Channel x Redundancy Check--TRCx Y, T

If the channel x tape check indicator is on, it is turned off and the computer takes its next instruction from Y. If the indicator is off, the next sequential instruction is taken. There is a TRC instruction for each channel:

TRCA	+0022	TRCC	+0024	TRCE	+0026	TRCG	+0027
TRCB	-0022	TRCD	-0024	TRCF	-0026	TRCH	-0027

Unit Addresses

All unit addresses for channel A and B (for both the BCD and binary operating modes) are shown in octal and decimal coding in Figure 58.

Device	Channel	BCD Mode		Binary Mode	
		Octal	Decimal	Octal	Decimal
Magnetic Tapes	A	1201-1212	0641-0650	1221-1232	0657-0666
	B	2201-2212	1153-1162	2221-2232	1169-1178
Card Reader	A	1321	0721		
	B	2321	1233		
Card Punch	A	1341	0737		
	B	2341	1249		
Printer	A	1361	0753	1362	0754
	B	2361	1265	2362	1266

Figure 58. Unit Addresses

The Table on the next page shows the relationship between Word Count (WC), End of record gaps (EOR), channel disconnect time, and what command is taken next.

DATA CHANNEL COMMANDS

Bits S, 1, 2	Command	Channel Disc. when WC=0?	WC=0; EOR gap not reached (reading)	WC≠0; EOR gap is reached (reading)	EOR written on tape?	Takes next command in sequence
0	IOCD	Yes	Go to EOR, no transmission	Read from next record.	Yes, when WC = 0	No
4	IOCP*	No	Next command reads from same record	Same as IOCD	No	Yes, when WC = 0
2	IORP	No	Same as IOCD	Does not read from next record	Yes, when WC = 0	Yes, when EOR is read or written
5	IOCT*	No	Same as IOCP	Same as IOCD	No	No ***
3	IORT	No	Same as IOCD	Same as IORP	Yes, when WC = 0	No ***
6	IOSP**	No	Same as IOCP	Same as IORP	No	Same as IORP
7	IOST**	No	Same as IOCP	Same as IORP	No	No ***

* If the word count (WC) is reduced to zero by the last word of a record and the next command is IORP, IOSP, IORT, or IOST, the present end of record gap (EOR) is recognized.

** If the word count is reduced to zero by the last word of a record and the next command is IORP, IOSP, IORT, or IOST, this next command normally would transmit no data and would be effectively skipped. If this next command enters the channel eleven (11) machine cycles after the previous command, the EOR is not recognized and the following record will be processed.

*** The next command is taken from the location specified by the LCH (load channel) instruction, or disconnects (and traps if the channel is enabled) if no LCH instruction is waiting. If the LCH is executed after the channel is disconnected, the LCH is treated as a no-operation and the I/O check indicator is turned on.

DATA CHANNEL TRAPPING

This feature allows the data channel to signal or interrupt processing by trapping the computer program. The trap may be initiated by: (1) the completion of a channel command, (2) an end of file, or (3) a tape check. These conditions are called channel signals.

Two instructions, enable (ENB) and restore channel traps (RCT), are used with the feature and are described in the Instruction Section.

A trap indicator on the operator's console indicates when a trap occurs. The execution of an enable or restore channel traps instruction will turn the indicator on. Execution of any trap turns the indicator off. With the indicator off, traps are said to be inhibited.

A data channel is enabled by use of the ENB instruction. This instruction conditions the channel so that a channel signal may be combined with it, when and if it occurs. When a channel is enabled for a particular channel signal (for example, tape check), the other signals are not used for trapping. Thus, the channel is said to be disabled with regard to these other channel signals. A logic flow chart of circuits involved with one data channel is shown in Figure 59. Note that the trap indicator must be on for a trap to occur.

Data channel may be individually or collectively prevented from causing traps until the program is able to handle them. In this event the channel signal is saved until the program allows it to become effective.

When a trap occurs, the contents of the instruction counter (IC) are stored and the next instruction is taken from a fixed location as follows:

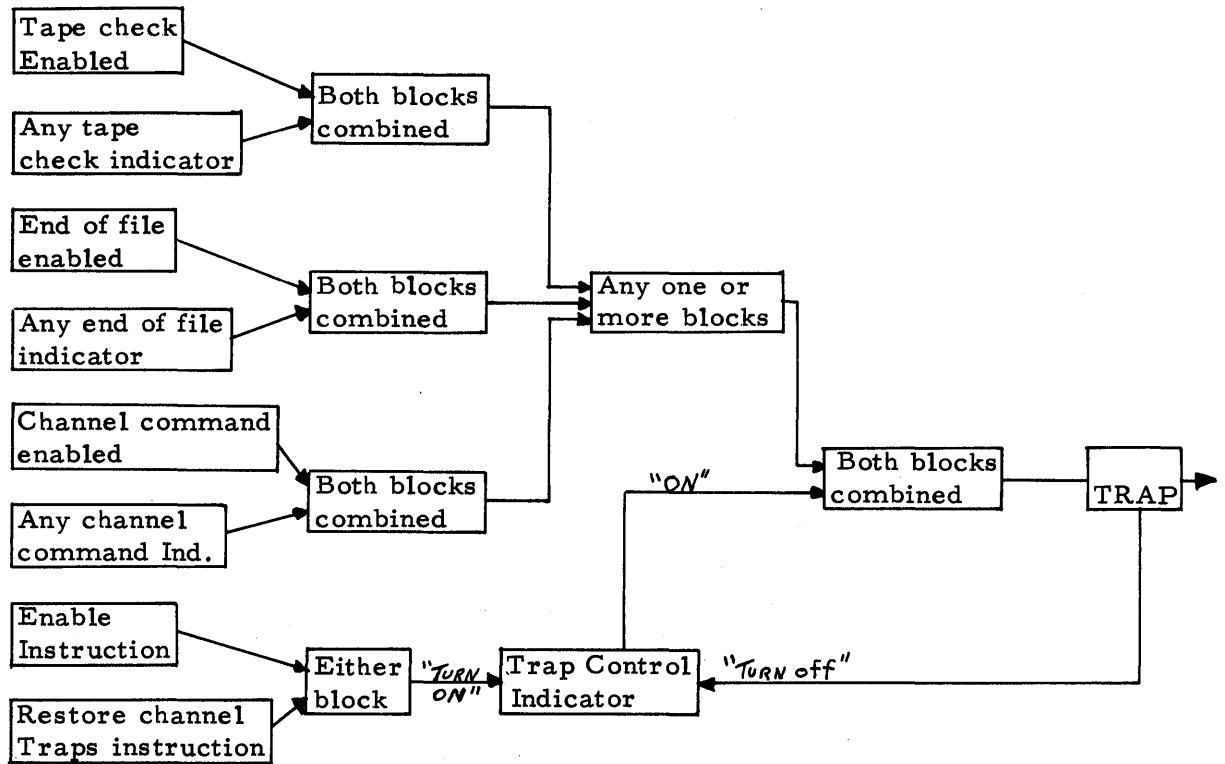


Figure 59. Logic Flow of Data Channel Trapping

<u>Channel</u>	<u>Store the IC at</u>	<u>Next Instruction From</u>
A	00012	00013
B	00014	00015
C	00016	00017
D	00020	00021
E	00022	00023
F	00024	00025
G	00026	00027
H	00030	00031

The first instruction after execution of a trap should be an unconditional transfer. If these instructions, located at the odd locations 00013-00031, are not provided by the programmer and do not alter the instruction counter, the program resumes from the point at which the trap occurred (after executing the instruction at the odd location).

If a trapping signal is generated while a channel is disabled or inhibited, the trap request is remembered until the channel is enabled or restored. The instruction following an ENB, RCT, or execute instruction will always be executed before another trap is processed. Furthermore, traps are prevented from occurring between a read or write select and the following

RCH or LCH instruction.

Floating Point Trap

During the execution of floating point instructions the resultant characteristic in the AC and MQ registers may exceed eight bit positions (result is too large for storage). The capacity is exceeded if the exponent goes beyond 177 or below -200. Beyond 177 is termed overflow while below -200 is termed underflow. Overflow and underflow may occur in either the AC or the MQ registers.

To aid the programmer in checking for these conditions, a check called floating point trap is used. The computer will, upon sensing an underflow or overflow, put the address plus one of the instruction that caused the condition into the address portion of location 00000.

An identifying code, telling whether an underflow or an overflow occurred and whether the most significant result is in the AC or MQ, is placed in the decrement portion of location 00000. The computer then executes the instruction at location 00010 and proceeds from there. These underflows and overflows are termed spills. The decrement positions and meaning of a 1-bit in these positions is:

<u>Decrement Position</u>	<u>Meaning</u>
14	Divide only (MQ register is not an extension of the AC).
15	Overflow in either AC or MQ, (or both), registers.
16	AC factor exceeded.
17	MQ fraction is excessive.

SUBROUTINES

To solve even the simplest of problems with a computer, it is necessary to write many instructions. These instructions are then strung together and form a program for the solving of the particular problem. This program is referred to as the "main program" and may contain a few or many hundred computer instructions.

It is very often desirable to be able to repeat the same group of instructions many times during the execution of a main program. Since numbers are usually recorded in the decimal format and must be converted to the form that the computer can use, a series of instructions that would perform a decimal-to-binary conversion would have much use. If this conversion were to be made many times within a program, it would not be desirable to write out the necessary instructions each time the conversion is needed.

Instead, the needed instructions could be written only once and the main program could then be arranged to transfer to this block of conversion instructions each time they are needed. Such a block of instructions is called a "subroutine;" that is, "a sequence of machine instructions which carry out a well defined function."

These subroutines normally perform such basic functions that they may be used in the solution of many types of problems. For instance, a subroutine which computes square roots could be used in a wide variety of scientific problems. Another example of a desirable subroutine would be one which computes the logical check sum for a block of words in storage. This check sum (number) could then be used to insure the validity of the block of words when they are moved about within the computer system.

Open Subroutine

Subroutines may be used in two ways with respect to the main program. One method is to insert the subroutine into the main program at the point where it is to be used. Subroutines designed for this type of usage are called "open-subroutines." For example, assume a square root subroutine (called SQRT in symbolic language) is to be used three times in a given main program. Figure 60 shows the main program after insertion of the subroutine.

Main Program

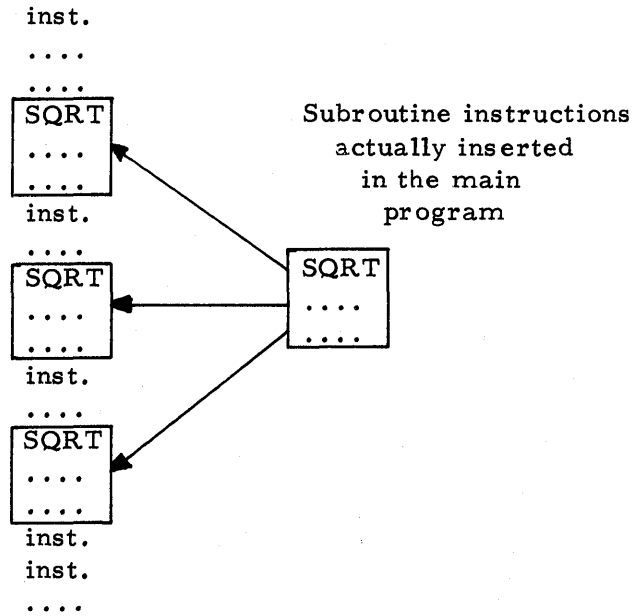


Figure 60. Open Subroutine Example

The open subroutine is thus "sandwiched" into a program as though it were a part of the original coding of the program. This type of subroutine is normally restricted to cases where the main program uses the subroutine only one time. This is so because of the waste of storage space in holding the square root instructions in the program each time they are needed.

Closed Subroutine

When the main program uses the same subroutine several times -- which is the common situation -- it is apparent that the open subroutine is not desirable. The subroutine type used in these situations is called a "closed subroutine".

The need for the instructions in the closed-subroutine may occur many times within one main program, but the set of instructions comprising the subroutine need appear in storage only one time. A transfer from the main program occurs whenever the subroutine is needed and, at the end of the subroutine, a transfer must be made back to the proper place in the main program. This process is shown in Figure 61. The places in the main program which call for the SQRT subroutine are marked "SQRT." Thus, when the SQRT subroutine instructions are needed, the program transfers out to these instructions, processes them and then transfers back to the proper place in the main program.

Main Program

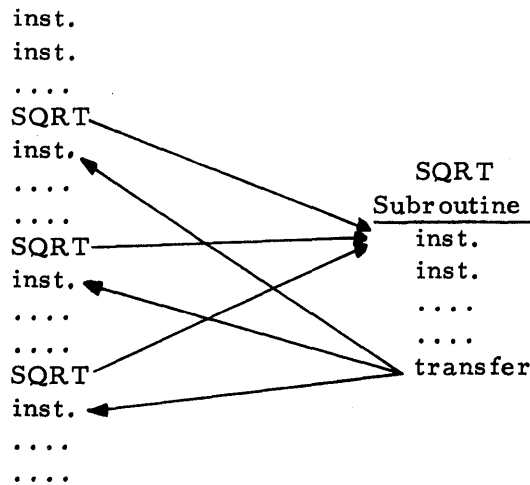


Figure 61. Closed - Subroutine Example

Problems of communication between the main program and the subroutine exist when using the closed-subroutine that do not occur with the open-subroutine. With a closed-subroutine, a transfer from the main program to the subroutine location must occur every time the subroutine is used. Additionally, the last instruction of the subroutine must transfer back to the main program at the instruction location immediately following the last linkage instruction.

The transfer of program control from the main program to the subroutine takes place with a set of instructions (or one instruction) known as the calling sequence or basic linkage. The calling sequence transfers control to the subroutine, tells the subroutine what instruction to return to when finished, and gives the subroutine any other information required.

Calling Sequence

The complete transfer of control between the main program and the subroutine is based on the use of the transfer and set index instruction. Upon execution of this instruction, the two's complement of the location of the link (calling sequence) instruction is placed in index register 4. From the standpoint of algebraic operation, the two's complement of a number is equivalent to the negative of the number. For example, 1 minus the two's complement of the link location is equivalent to 1 plus the location of link.

The calling sequence and a subroutine which computes logical check sums is shown in Figure 62. The return to the main program is also shown. The calling sequence (linking instruction) would, of course, be physically located in the main program along with the location of the 1st word in the

1	2	6	7	8
LINK	TSX	BLKSM, 4		Linking Instruction
	TSX	FIRST		Location of First Word in Block
	TSX	N		Number of Words in Block
	TSX	CKSUM		Location for Storing Check Sum
	TSX			Location to which control will be returned
				<u>SUBROUTINE</u>
BLKSM	CLA	2, 4		Get number of words in block
	PAX	0, 1		Place N in index register 1
	ADD	1, 4		Add location FIRST to form FIRST + N
	STA	ADDER		Initialize logical add instruction
	CLA	3, 4		Get location to store check sum
	STA	STSUM		Place address in store instruction
	CLM			Clear accumulator register to zeros
ADDER	ACL	0, 1		Two instruction loop to compute logical
	TIX	ADDER, 1, 1		check sum for block of N words
STSUM	SLW	0		Store check sum in location CKSUM
	TRA	4, 4		Return control to main program

Figure 62. Calling Sequence and Check Sum Subroutine

data block (for which the check sum is being computed), the number of words in the block, and the location in which the computed check sum is to be placed. The check sum routine itself, is physically separate from the main program.

In the subroutine the instructions which make use of the two's complement operation are:

<u>Instruction</u>	<u>Effective Execution</u>	<u>Equivalent Execution</u>
CLA 2,4	CLA 2-(2's comp link)	CLA LINK + 2
ADD 1,4	ADD 1-(2's comp link)	ADD LINK + 1
CLA 3,4	CLA 3-(2's comp link)	CLA LINK + 3
TRA 4,4	TRA 4-(2's comp link)	TRA LINK + 4

From the above table it can be seen that the subroutine will be able to make use of the information found in the parameter locations of the calling sequence without knowledge of their exact location in storage.

Since LINK is a symbol representing any location in storage, the subroutine can thus communicate with the main program at any location in the main program. By means of the TRA 4,4 instruction, the subroutine has the ability to transfer control back to the proper location in the main program.

One of the main responsibilities of a subroutine is to insure that when control is transferred back to the main program the status of all registers is the same as when control was transferred to the subroutine. This does not apply, of course, to a subroutine designed to change a machine condition. For example, in the check sum routine the contents of index register 1 are destroyed. Thus, when control is transferred back to the main program, index register 1 may not be the same as when control was transferred to the subroutine.

The contents of index register 1 may be preserved by adding the instruction SXA SAVE, 1 immediately after the CLA 2,4 instruction. Thus, the contents of index register 1 will be stored in the address part of location SAVE. Now, if location SAVE is inserted just before the TRA 4,4 instruction and contains the AXT 0,1 instruction, the original contents of index register 1 will be replaced just before control is transferred back to the main program. The improved subroutine is shown in Figure 63.

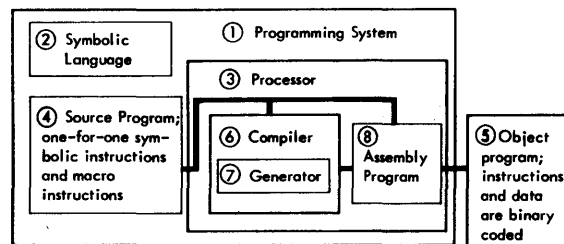
Location		Operation	Address, Tag, Decrement/Count	Comments
1	2	6 7 8		
BLKSM		CLA	2, 4	
		SXA	SAVE, 1	Save the contents of index register 1
		PAX	0, 1	by
		ADD	1, 4	storing
		STA	ADDER	them
		CLA	3, 4	in
		STA	STSUM	the
		CLM		address
ADDER		ACL	0, 1	part
		TIX	ADDER, 1, 1	of location
STSUM		SLW	0	SAVE
SAVE		AXT	0, 1	Replace the original contents of index register 1
		TRA	4, 4	

Figure 63. Complete Check Sum Routine

IBM PROGRAMMING SYSTEMS

INTRODUCTION

Terms like programming system, symbolic language, processor, compiler, generator, etc., have become commonplace enough so that their meaning differs dependent on previous experience. The intended meaning of these terms (in this text) and their inter-relationship is shown in Figure 64. The circled numbers with each term refer to the comments below in the Figure.



- 1. Programming System:** Any method of programming problems, other than machine language, that consists of a language and its associated processor(s).
- 2. Symbolic Language:** Any collection of symbols used in programming to represent operation codes, functions, and/or addresses, with rules of usage.
- 3. Processor:** A machine language program that performs the functions necessary to convert a source program into the desired object program.
- 4. Source Program:** A program coded in other than machine language that must be translated into machine language before being used.
- 5. Object Program:** A program coded in machine language for use by the computer.
- 6. Compiler:** A translation program that translates macro instructions of a symbolic program into one-for-one symbolic instructions, and then passes the entire set of instructions to an assembly program for final translation.
- 7. Generator:** A machine language program used during compiling to produce symbolically coded (one-for-one) instructions that will perform the operation called for by the symbolic coding of the source program.
- 8. Assembly Program:** A translation program that substitutes binary coding for symbolic instructions, may assign storage locations, and performs other activity necessary to produce a loadable object program. This object program may be self-loading or, in some systems, a load program is needed.

Figure 64. Programming System Segment Relationship

Operating Systems

In the beginning -- 10 years ago-- the programmer who had access to a working, reasonable efficient, symbolic assembly program could consider himself lucky. With the advent of FORTRAN, COBOL, and macro languages, and the sophistication of input-output concepts and devices, it rapidly became apparent that some overall control or monitor program was necessary.

This type of program is called an Operating System or IBSYS. It is an integrated set of systems, coordinated by the IBSYS Basic Monitor and

uses an Input/Output Control System. The basic monitor provides continuous operation during a sequence of jobs, each of which might involve a different programming system. The systems that work with the Basic Monitor include (Figure 65) :

- IBEDT: A systems library editor program that maintains the system library.
- IOCS: A control system for efficient scheduling of input and output.
- IBSRT: A generalized sorting program to sort and merge data.
- IBJOB: A processor program containing the following components:
- IBJOB Monitor: Supervises the execution of the compilers, assembly program, and loader.
- IBLDR Loader: Processes and combines programs produced by IBCBC to form one binary object program.
- IBLIB Subroutine Library: Contains routines that will be loaded if required for object program generation.
- IBMAP Macro Assembly Program: Processes programs written in the MAP language and the internal languages programs produced by the COBOL and FORTRAN compilers. IBCBC produces for each compilation a binary program card deck that retains enough symbolic content to enable communication with previously compiled program decks.
- IBFTC FORTRAN Compiler: Processes programs written in the FORTRAN IV language and produces input to IBCBC.
- IBCBC COBOL Compiler: Processes programs written in the COBOL language and produces input to IBCBC.

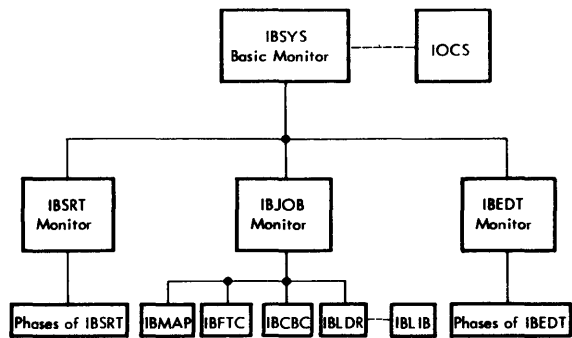


Figure 65. IBSYS Operating System Components

The operation of the IBJOB Processor on source language programs of different types is shown in Figure 66.

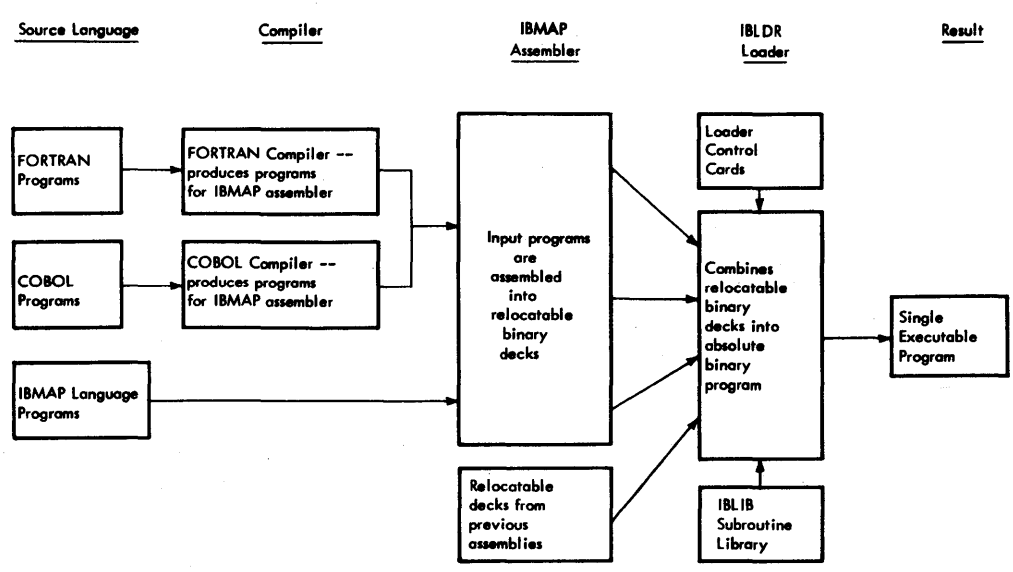


Figure 66. IBJOB Processor Flow Chart, Source Programs

Operation of the IBM Operating System is automatic; once an input reel is mounted it should not be necessary for the computer to be idle until the output reel is dismounted, provided enough input-output devices are available. It should not be necessary for the operator to take any action other than dismounting unloaded reels and replacing them with reels to be used later in the job, or on succeeding jobs.

The operation of each phase of IBSYS is directed by control cards, but the programmer has to use only the parts of the operating system that apply to his current job. Basic monitor control cards are ordinarily not used by the programmer; they are the concern of the machine operator. Thus, the programmer concerns himself with a comparatively small number of control cards to run any one job.

Over-All Operation

The Basic monitor (IBSYS) acts as an intersystem monitor that calls the appropriate submonitor according to control card specifications. A portion of IBSYS remains in main storage at all times and permits return of program control to the basic monitor, reference to the input-output control system, and loading of the programming systems into main storage. This portion of IBSYS contains a set of common system routines and subroutines and a communications region where common information shared by the programming systems is maintained.

A supervisory portion of the basic monitor is called in when required to transfer control between system monitors, to initialize the first portion of IBSYS, to change machine environment, and to assign external storage devices to logical input-output functions.

Figure 67 shows a typical input to the IBSYS Operating System. The input consists of a data card (handled by the basic monitor), two jobs to be processed by IBJOB, followed by the control cards for an IBSRT machine run. The first job to be processed by IBJOB is a program consisting of two FORTRAN source decks, two MAP source decks, a binary deck (from a previous compilation), and data cards. The second job is unspecified.

When the basic monitor (which is already in main storage) encounters the \$IBJOB control card, it transfers program control to the IBJOB monitor. The \$ symbol specifies that this particular card is a control card. The \$IBJOB card is recognized by the IBJOB monitor, as well as by the basic monitor, and indicates the beginning of a new job. Thus, the IBJOB monitor will supervise job-to-job transition until it encounters a \$IBSYS control card, which indicates that the next operation is not within IBJOB's operating scope. When this occurs, program control is returned to the basic monitor.

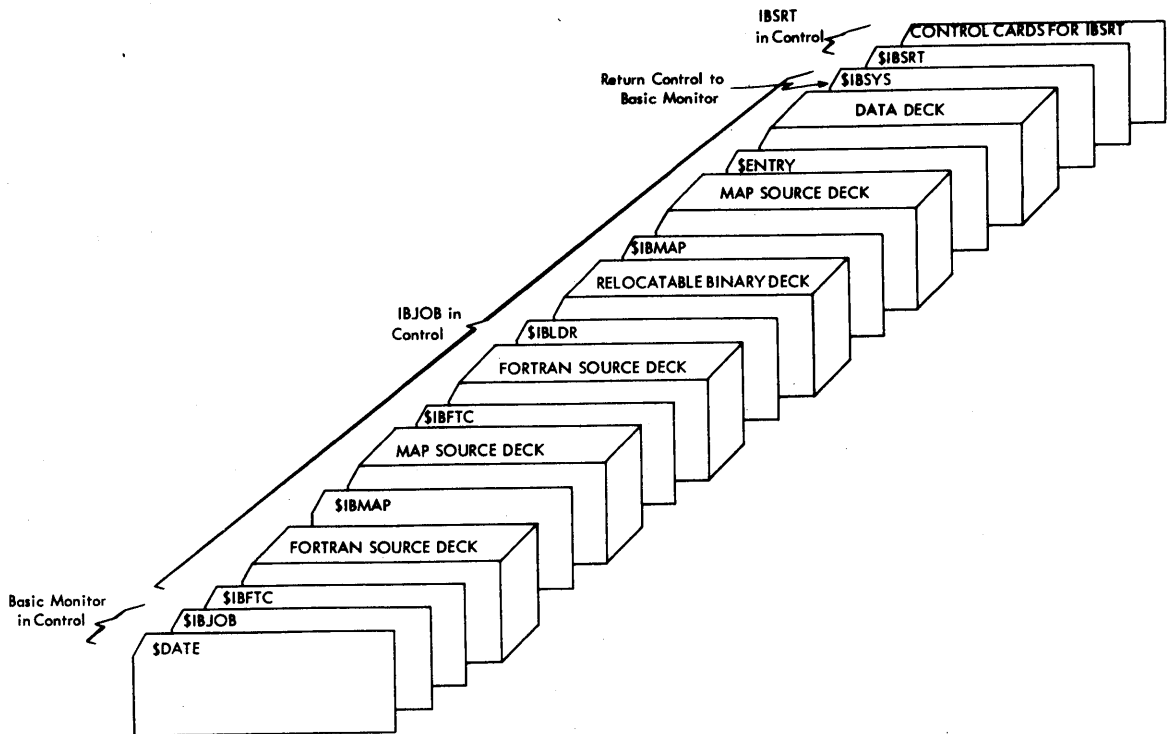


Figure 67. Typical Input for the IBSYS Operating System

When the basic monitor encounters a \$IBSRT control card, it calls in the sort monitor and transfers control to it. The sort monitor controls execution of the sort, according to specifications on the control card. Successive sorting and merging jobs can be handled by the sort monitor without relinquishing control to the basic monitor. Control is returned to the basic monitor when the next \$IBSYS control card is encountered.

FORTRAN ASSEMBLY PROGRAM

An assembly program is similar to a compiler program (such as FORTRAN II) in that it produces machine language programs. However, the symbolic language used with an assembler is closely related to the machine language of the computer, while the source language used with a compiler resembles a language in which problems are commonly stated, such as mathematical notation.

Compilers have several advantages over assemblers: the language used with a compiler is easier to learn and to use (because of prior knowledge);

the programmer using a compiler usually does not need an intimate knowledge of the internal operations of the computer; programming is faster; and the time required to produce a finished, working program is reduced, since there is less chance of programming errors, and since most errors which are made are detected by the compiler.

The assembler compensates for these disadvantages by offering the programmer a degree of flexibility not available with any present-day compiler. The FORTRAN Assembly Program (FAP) is such an assembler.

Using FAP and FORTRAN operating under either the IBM FORTRAN monitor system, or the Basic Monitor system (IBSYS), a programmer may write the major parts of his program in FORTRAN, using FAP subroutines where necessary to accomplish those parts of the job for which FORTRAN is not suited. Likewise, he may write the major part of the program in FAP using FORTRAN subroutines for certain computational and input-output operations. For those jobs which are coded entirely in symbolic language, FAP may be used to produce an "absolute" program which can be either loaded in the computer by the Monitor program or used independently of the Monitor.

Elements of the FAP Language

Symbolic instructions are punched one per card in the following format. The location field, which may be blank, occupies card columns 1-6. Card column 7 is always blank. The operation field begins in column 8 and is from three to seven characters in length. A blank column or a comma separates the operation field and the variable field. The variable field (operand) may begin immediately following the operation field, but must not begin after column 16. The variable field may not extend beyond column 72 and must be followed by a blank column to separate it from the comments field.

The comments field follows the variable field and extends through column 72. In the absence of a variable field, the comments field may not begin before column 17.

Symbols

A symbol-location symbol and symbolic address - is a string of from one to six non-blank alphabetic and numeric characters, at least one of which is non-numeric. In addition, a symbol may include the characters decimal point, left parentheses, and right parentheses. A symbol may not include the special characters: +, -, *, /, \$, =, ', or , (comma).

A symbol may thus be used as name for a storage location, tape unit address, or any other program parameter. A symbol is defined in one of three ways:

1. By its appearance in the location field of an instruction.
2. By its use as the name of a subprogram.
3. By its appearance in a system symbol table.

Each symbol used in the program must be defined only once.

Symbolic Expressions

In writing symbolic instructions, the programmer is concerned with building expressions to represent the address, tag, and decrement portions of the machine instruction.

The smallest component of an expression is an element, which is either a single symbol or a single integer. When elements are combined with operators (symbols representing machine operations) a term is formed. A term may consist of a single element, two elements separated by an operator such as * or / (the * character means multiply and the / character means divide), three elements separated by two operators, and so on. A term must begin with an element and end with an element. Two operators or two elements in succession are not allowed.

In addition to being an operator, the asterisk is also an element. In this use, the asterisk stands for the location of the instruction in which it appears. Thus, the element * will have different values in different instructions. There is no ambiguity between this use of the * and its use to denote multiplication, because the position of the asterisk always makes clear what is meant. For example, the asterisk in the following instruction means

CLA *-1

that the location to be cleared and added to the accumulator is the location immediately in front of (lower than) the location of the CLA. That is if the CLA *-1 instruction is located at core location 105, the core location 104 will be brought to the accumulator when the instruction is executed. For the other asterisk use, the instruction

CLA A*B

means that the quotient of A times B is to be brought into the accumulator.

An expression is made up of terms separated by the + or - operators, (+ means add and - means subtract). An expression may consist of a single term, two terms separated by + or -, three terms separated by two operators, and so on. The programmer may not write two operators in succession or two terms in succession, but an expression may begin with the + or - symbol.

An expression is terminated by a comma symbol (,) or, for the last expression of a statement, by a blank column. A negative expression is represented in 2's complement notation. A null expression is an expression that is indicated as being present but has no value. It can occur:

1. When an assembly scan encounters a comma rather than the first element of an expression. The comma shows that a null expression is indicated. Two consecutive commas indicate a null expression, or a comma as the first character of the variable field indicates that the first expression is null.
2. When a scan encounters a blank character following a comma. This character combination indicates that the last expression of the statement is a null expression.

Literals

Often a programmer wishes to refer to a word containing a constant. For example, if he wishes to add the number 1 to the contents of the accumulator, he must have somewhere in storage a word containing the number 1. Pseudo-operations are provided to allow introduction of data words and constants into the program, but often this introduction is more easily accomplished by the use of a literal.

The appearance of a literal directs the assembler to prepare a constant equivalent in value to the contents of the literal subfield, store this constant in a location at the end of the program, and replace the address field of the instruction containing the literal with the address of the constant thus generated. Three types of literals are permitted: decimal, octal, and alphameric.

Decimal Literals. A decimal literal consists of the = symbol followed by a decimal data item. For example, the instruction

MPY = -3

means "multiply the contents of the MQ register by the decimal number -3." (that is, multiply the contents of the MQ by the contents of a storage location that contains -3). Three types of decimal data items are recognized.

Decimal Integer. A decimal integer is composed of one or more digits, and may be preceded by a plus or minus sign. A decimal integer is distinguished from other types of decimal data items by the absence of the letter B, the letter E, and the decimal point itself.

Floating Point Number. A floating point number has two components: (1) the principal part is a decimal number written with a decimal point. The decimal point may appear at the beginning, end, or within

the principal part, or it may be omitted if the exponent part is present. If omitted, the decimal point is assumed to be at the right end of the principal part; and (2) the exponent part consists of the letter E followed by a signed or unsigned decimal integer. The exponent part must follow the principal part; it may be omitted if the principal part contains a decimal point.

A floating point number is distinguished from a decimal integer by the presence of either a decimal point or the letter E (or both). It is distinguished from a fixed point number by the absence of the letter B.

Fixed Point Number. A fixed point number has three components: (1) the principal part is a decimal number written with or without a decimal point. The decimal point may appear at the beginning, end, or within the principal part, or it may be omitted. If omitted, the decimal point is assumed to be at the right end of the principal part; (2) the exponent part consists of the letter E followed by a signed or unsigned decimal integer. The exponent part may be absent; if present, it must follow the principal part, and may precede or follow the binary place part; and (3) the binary place part consists of the letter B followed by a signed or unsigned decimal integer. The binary place part must be present in a fixed point number and must follow the principal part. If the number has an exponent part, the binary place part may precede or follow the exponent part. A fixed point number is distinguished from other types of decimal data items by the presence of the letter B.

Literals are considered to be single precision numbers unless two E's (EE) appear in the exponent of a floating point number. Double precision numbers are stored in consecutive storage locations with the high-order parts first.

Octal Literals. An octal literal consists of the character =, followed by the letter O, followed by a signed or unsigned octal integer. For example, the instruction

ADD = 037

means "add to the contents of the accumulator register the contents of a core location that contains 000000000378."

Alphameric Literals. An alphameric literal consists of the character =, followed by the letter H, followed by a signed or unsigned octal integer. For example, the instruction

LDQ = H12AB

would load into the MQ register the contents of a core location that contains 0102212260608. When fewer than six BCD characters are

specified, the unspecified characters are assumed to be BCD zero characters.

Data Generating Operations

These pseudo-operations (OCT, DEC, BCI, and BCD) may be used to introduce words of data into a program during the assembly run. Numbers introduced in this way are often referred to as constants.

OCT--Octal Data. The OCT operation is used to create binary data expressed in octal form. It consists of a symbol or blanks in the location field, the operation code OCT in the operation field, and one or more subfields, each containing a signed or unsigned octal integer, in the variable field. The symbol in the location field is the address of the pseudo-operation.

The subfields of the variable field are separated by commas; any number of subfields are permissible, but the last subfield must be ended by a blank.

The effect of this operation is to convert each subfield to a binary word; these words are assigned successively higher storage locations as the variable field is processed from left to right. If there is a symbol in the location field, it refers to the first word of data generated. An example of the data generating function of the OCT pseudo-operation is:

OCT -2345677777,63,47,5,

which generates data in core locations as follows:

first location	data
n	-2345677777
n 1	0000000063
n 2	0001000047

BCI -- Binary Coded Information. The BCI pseudo-operation is used to create binary coded character data. Each data word generated consists of six 6-bit characters in standard BCD character code. It consists of a symbol or blank in the location field, the operation code BCI in the operation field, two subfields in the variable field (count and data subfields).

The number in the count subfield specifies the number of six character words to be generated; the number of characters in the data subfield is the number in the count subfield multiplied by six. Since the count subfield determines the total length of the variable field, the comments field is assumed to begin immediately following the end of the data subfield, and no blank character is needed to separate the comments field from the variable field.

Thus, the BCI operation introduces data words into the consecutive locations, the number of words generated being equal to the number in the count subfield. If there is a symbol in the location field, it refers to the first word of data generated. An example of the data generating function of the BCI is:

i BCI 2, BCD

which means that location i and i plus 1 contain

i 222324604425
i 1 626221272560

BCD Binary Coded Decimal. The BCD pseudo-operation has been supplanted by the BCI in 7090 operation. BCD operation is similar to BCI but is used with the 704 Symbolic Assembly Program. Differences between the two are:

location	data
n	-23456777777
n + 1	+00000000063
n + 2	+00000000047
n + 3	+00000000005
n + 4	+00000000000

DEC -- Decimal Data. The DEC pseudo-operation is used to create words of data expressed as decimal numbers. DEC is identical to OCT, except that the subfields of the variable field are taken to be in decimal data notation. It consists of a symbol or blanks in the location field, the operation code DEC in the operation field, and one or more subfields, each containing a decimal data item, in the variable field.

The subfields of the variable field are separated by commas; any number of subfields is permissible, but the last subfield must be terminated by a blank.

The effect of this operation is to convert each subfield to a binary word; these words are stored in successively higher storage locations as the variable field is processed from left to right. If there is a symbol in the location field, it refers to the first word of data generated. An example is:

LAST DEC 15, -24, 9, 107

which generates data in the following locations:

location	data
LAST	+000000000017
LAST + 1	-000000000030
LAST + 2	+000000000011
LAST + 3	+000000000153

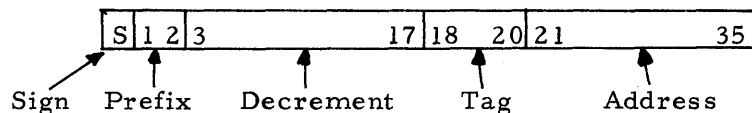
1. The operation code BCD appears in the operation field.
2. The count digit must appear in column 12.
3. No comma separates the count digit from the data subfield; the data subfield always begins in column 13.
4. A blank or zero in column 12 is used to indicate ten data words.

If column 12 does not contain a blank or the digits 0-9, one word of blanks is generated and the error flag E is given.

Extended Operation Codes

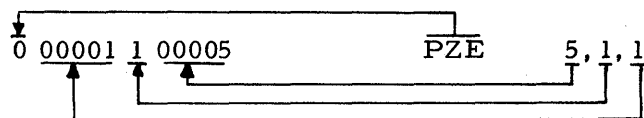
Certain operations are provided by the assembly program that do not fall into any of the previous categories although they are similar to the data generating operations. These operations are called extended or prefix operations.

The instruction format, as seen by these operations, is:



The sign and prefix portion make up an operation field, positions 3-17 (15 positions) are the decrement, 18-20 are the index register tag, and positions 21-35 form the address field.

The extended operations may be used to specify parameters in each of the four sections of a binary word -- prefix, decrement, tag, and address. The decrement, tag, and address are specified in the variable field of the operation. Of course, they must be specified in the normal order; address, tag, and decrement. For example, assume the extended operation PZE (plus zero) is used in a program in the form shown on the left below. The octal configuration on the right would be generated at the location of the PZE.



Other extended operations are provided to specify the value of the prefix bits in all usable configurations. Data generated by these operations is the same as with the PZE except in the prefix portion of the location. All extended operations, their meaning, and octal prefix generated are shown in the table below.

<u>Operation Code</u>	<u>Meaning</u>	<u>Octal Prefix</u>
PZE	Plus Zero	0
PON	Plus One	1
PTW	Plus Two	2
PTH	Plus Three	3
MZE	Minus Zero	4
MON	Minus One	5
MTW	Minus Two	6
MTH	Minus Three	7

Note that certain operations may generate the same prefix in an instruction word. For example, the extended operation PTH, the input-output command IORD, and the control instruction TXH all generate the prefix of 3.

Storage Allocation Operations

The following operations are used to allocate core storage space:

BSS--Block Started with Symbol. The operation consists of a symbol or blanks in the location field, the operation code BSS in the operation field, and any expression in the variable field.

The effect of this operation is to reserve a specified amount of storage. This is achieved by increasing the value of the current location counter by the assigned value of the variable field expression. If there is a symbol in the location field, its definition is taken to be the value of the location counter before the increase.

BES -- Block Ending with Symbol. This operation functions exactly like BSS, except that a symbol in the location field is defined after the location counter is increased.

BSS and BES Examples. An example of reserved storage locations generated with the BSS operation is:

TRA	START
BSS	4
DEC	97

with the TRA instruction located at location 1000, the BSS reserves four locations, and the DEC is then located at location 1005. In a similar fashion, the BES appearing in the instruction string

TRA	START
BES	4
DEC	97

gives the following result:

1000	TRA	START
1005	BES	4
1005	DEC	97

Symbol Defining Operations

The following operations are declarative in nature. They are used to define symbols.

EQU--Equal. The operation consists of a symbol in the location field, the operation EQU in the operation field, and any expression in the variable field.

The effect of the operation is to give the location field symbol the same definition as the variable field expression. As an example, consider the use of EQU in the following instruction string:

	CLA	TMPI
FSTL	EQU	**
	ADD	TMP2

If the CLA instruction is assigned to location 102, the symbol FSTL would be defined as a symbol, which can be relocated in the program, whose value is 103; the ADD instruction would then be assigned to location 103. Note that the occurrence of the EQU between two instructions does not alter the sequence of locations assigned by the assembler.

UPDATING SYMBOLIC TAPES

The FAP assembler has two uses: the translating of symbolic language to machine language, assigning storage locations, and performing other operations necessary to produce the final object program (this is known as assembly); the second use is to update a symbolic tape by changing, deleting, or adding instructions. The update may occur with the assembly of the updated symbolic language program.

Updating involves up to three tapes: one tape (the systems tape) is used by all assemblies, compilations, and executions; and two tapes (the update input and update output tapes), are used only for updating.

The systems input tape contains all jobs to be processed by a monitored system. A job may have many elements: compilations, assemblies, and executions. In the case of an update, the corrections to be made to a

specified symbolic tape constitute a job element and are contained in a job on the systems input tape.

The update input tape is a blocked or unblocked symbolic tape which requires updating. All additions, changes, and deletions are made on the basis of the serialization in card columns 73-80. Thus, instructions on the update input tape must be properly serialized. An end of file on the update input tape is ignored.

The update output tape is an updated, blocked or unblocked, symbolic tape which includes all the requested corrections. This tape may be assembled at a later date; if it includes the necessary control cards, it may become a systems input tape.

The monitored system requires two tapes: the system listing tape contains the listing output for all successful elements of all jobs on the system input tape; a system punch tape contains row or column binary card images for the processed jobs.

UPDATE Pseudo-Operation

This operation is used to initiate the updating mode, to assign update input and update output tapes, and to specify whether or not blocking and assembly are required. The UPDATE consists of:

1. Blanks in the location field.
2. The code UPDATE in the operation field.
3. Two symbolic expressions and two strings of characters, all separated by commas, in the variable field.
4. Serialization, if wanted, in card columns 73-80.

The first subfield is an expression designating the logical systems tape number of the update input tape. This subfield may be void or zero, implying that there is no update input tape, or it may be the logical number of a tape containing the symbolic input to be updated. If the FORTRAN Monitor system is used, tapes 1 through 8 may not be used as update tapes.

The second subfield is an expression designating the logical tape number of the update output tape. This subfield may be void or zero, implying that there is no update output tape, or may be the logical tape number that is to contain the updated symbolic data.

The third subfield is a string of characters which indicates whether or not the update output tape is to be blocked. If this subfield is void or zero, the update output tape will be blocked; otherwise, the update output tape will not be blocked.

The fourth subfield is another string of characters which indicates whether or not assembly is required. If this subfield is void or zero, assembly is required; otherwise, assembly is not required. The UPDATE containing this subfield must appear in the first card group. Having once appeared, neither this subfield nor its preceding comma may appear in another UPDATE card.

If assembly is not required, FAP is reduced to an updating or blocking routine, or both, since no table entries are made and Pass 2 is omitted. The only instructions that are recognized are update pseudo-operation, END, and those instructions with an asterisk (*) is card column 1.

Serialization in card columns 73-80 of the UPDATE may be useful for multi-reel input and output.

Example of UPDATE

The coding form shown in Figure 68 shows one possible use of UPDATE. The PZE pseudo-operation is to be added to an assembly program at locations OPTB0471 (serialization number) and OPTB0472.

REWIND Pseudo-Operation

The REWIND pseudo-operation causes the addressed update tape to be rewound. The REWIND consists of:

1. Blanks in the location field.
2. The code REWIND in the operation field.
3. A FAP expression, or logical tape unit number in the variable field.
4. Serialization, if wanted, in card columns 73-80.

If the logical tape number in the variable field is that of the current update output tape assigned by an UPDATE, the last partial block of instructions waiting to be output will be written on the update output tape before it is rewound.

If the variable field is blank, the update output tape is rewound. If the update input or update output tape is already rewound, it is logically disconnected. No update operation referring to it will be executed unless it is reassigned by a subsequent UPDATE pseudo-operation.

DELETE Pseudo-Operation

This operation causes the deletions of one or more instructions on the update input tape. Deleted cards will appear on the pre-processor assembly listing, labeled as deleted. The DELETE consists of:

Share 709 Symbolic Coding Form

Example of UPDATE Control Card Use			
Balance	Date	Page	of
	1/64	1	1
Location	Operation	Address, Tag, Decrement, Count	Comments
6 7 8	14	16	60 61 68 69 72 73
●	Row binary card	to call in the system off of tape	
*	DATE	1/64 *FAPUP	
*		I.D. Card	
*	●	XEQ	
*	●	FAP	
*		ADD PZE Pseudo-operation (comment Card)	
●	UPDATE	12	
	SKIPTO		OPTB0010
	BCI	1,000PZE	OPTB0471
	OCT	000000000000	
	END		OPTB0790
●		Column Binary Cards	
●		for the Main Program	MAIN0000
●		END OF FILE CARD	MAIN0001
*	●	ENDTAPE	EOF
			ENDTAPE

529

Figure 68. UPDATE Example

1. Blanks in the location field.
2. The code DELETE in the operation field.
3. THRU, when required, in the variable field.
4. A serialization number of eight characters in card columns 73-80.

If the variable field is blank, the instruction with matching serialization will be deleted; the variable field contains THRU, all instructions starting at the current position of the update input tape, up to and including the instruction with matching serialization, will be deleted. In the latter case, if matching serialization does not exist, deletions are made up to, but not including, the next instruction of higher serialization.

LBL (Label) Pseudo-Operation

The LBL causes binary cards to be serialized in card columns 73-80. It consists of:

1. Blanks in the location field.
2. The code LBL in the operation field.
3. Two subfields, separated by commas, in the variable field: the first subfield contains up to eight alphabetic and numeric characters; the second subfield may contain any non-zero, non-blank character, or the number 1.

Serialization begins with the characters appearing in the first subfield of the variable field. The characters are left justified and filled with zeros. Serialization is incremented until the right most non-numeric character is reached, at which time the non-numeric portion recycles to zero. This subfield may not contain any special characters.

If the variable field of the LBL is blank, serialization is discontinued or never occurs. In order to serialize from zero, an explicit zero must appear in the variable field.

A non-blank, non-zero second subfield, if it exists, causes serialization to be listed. If this second subfield is the number one, only the first use of the LBL will be listed.

COUNT Pseudo-Operation

Some of the speed of FAP assembly is due to the fact that it does not keep the computer waiting while a tape unit rewinds. The intermediate information produced and used during assembly is written on two tapes with half of the information on each, so that one of these tapes is in use while the other is rewinding.

The assembler is given an estimate of the number of cards in the symbolic deck by the COUNT card which must be the first card of the deck. The COUNT card consists of:

1. Blanks in the location field.
2. The code COUNT in the operation field.
3. A single decimal integer, which is the estimate of the number of cards in the symbolic deck, in the variable field.

If the COUNT card is missing or contains anything but a decimal integer in the variable field, the assembler assumes the card count is 2000 and prints out "CARD COUNT ESTIMATE MISSING" in the pre-processor assembly listing.

PCC (print control card) Pseudo-Operation

This operation causes the DETAIL, EJECT, INDEX, LBL, LIST, REF, SPACE, TITLE, and TTL pseudo-operations to be listed. The PCC consists of:

1. Blanks in the location field.
2. The code PCC in the operation field.
3. Blanks, ON, or OFF in the variable field.

If any of the pseudo-operations, whose listing is controlled by PCC, is in error and is flagged, the pseudo-operation will always be listed. PCC itself will always be listed. Alternate appearances of PCC turn this feature ON and OFF. The initial mode, prior to the appearance of the PCC, is OFF.

The use of ON or OFF in the variable field gives absolute control of the PCC itself.

Program Linking Pseudo-Operation

The ENTRY pseudo-operation is used within a program to provide a communication link between the main program and other programs. The character \$ may also be used for this purpose.

ENTRY is used to define an entry point to a relocatable subprogram. A main program is distinguished by the fact that it contains no ENTRY instructions or that it contains an ENTRY with an explicit zero in the variable field. The ENTRY consists of: blanks in the location field, the code ENTRY in the operation field, and a single symbol in the variable field.

The variable field symbol must be defined subsequently as a relocatable symbol and is the name of a subprogram. The first character of the subprogram must not be a numeric character.

ENTRY performs two functions: (1) the symbol in the variable field is placed in the program card. An explicit zero in the variable field will cause the program card to indicate that the entry point to the main program is the first instruction following the transfer vector and the linkage director; and (2) the value of the symbol, which is defined in the program, is placed in the program card following the symbol itself.

Thus, the ENTRY establishes the symbol as a name of the program and identifies the associated entry point with it. There may be more than one ENTRY in a subprogram. For example, a subroutine to computer sines and cosines could be:

				00001	ENTRY	SIN	
				00000	ENTRY	COS	
00000	0300	00	0	00072	COS	FAD	1.57079632679
00001	0634	00	4	00071	STO	SXA	ARG

The ENTRY may also be used to provide secondary entries for subroutines. If the symbol in the variable field is preceded by a minus sign, the word on the program card which contains the address of this entry point will have a 1 in the sign position. This causes the loader to ignore the subroutine unless one of its primary entries has also been called. This feature of FAP is useful when assembling subroutines for inclusion in the Library tape.

FORTRAN II MONITOR PROGRAMMING SYSTEM

The 7090/7094 FORTRAN II Monitor System is used to compile FORTRAN coded source program, assemble FAP coded source programs, and to execute the resultant object programs. Input to the Monitor System consists not only of the source program, but may also include FAP symbolic cards, object program cards, data cards, and Monitor control cards. The order of the different types of input does not matter, provided that each separate deck, whether source program, object program, etc., is preceded by appropriate control cards.

The FORTRAN II Compiler is considered a subsection of the Monitor. Under FORTRAN control a single source program may be compiled. Nothing further, including execution, may be done. If multiple compilation of a series of FORTRAN source programs is desired, Monitor control is needed.

There is a complete set of control cards for the Monitor System. These cards are distinguished by an asterisk (*) in column 1. In general, they are of two types; one type governs the job as a whole (telling what it consists of), and the other governs output options. In addition, DUMP,

START, and RESTART cards are available for use. These three are self-loading binary cards and perform the dump, start, and restart operations necessary for system use.

The Monitor System uses eight tapes on two channels. These are the A1 through A4 and B1 through B4. A2 is the input tape and A3 is the output tape. The Monitor System tape is mounted on tape unit A1.

Monitor Control Cards

All Monitor control cards must have as '*' in card column 1. With the exception of the I D card, the specific control instruction of the card is punched in columns 7-72. Punching may be done according to normal FORTRAN rules. Only the control cards that are used in this program are described in this test.

I D Card. This card must be present for every job, and if there is no date card, it must be the first card of the job. If there is a date card, the date card is first followed by the I D card.

XEQ Card. The execute card must follow the I D card of a job that is to be executed after it is assembled.

DATE Card. This card permits the programmer to obtain the present data as an additional part of the heading of each printed page (he must supply the date in the DATE card). The DATE card may be placed in either or both of the following places.

1. Preceding the I D card for a job. The DATE card is the only card which may precede the I D card.
2. Following the Monitor START card. The date specified in this manner is used throughout the Monitor run. However, a DATE card appearing with a job as the first card takes precedence over the DATE card following the START card for this job only.

Following are examples of the date field which is placed after the word DATE in the control card: 2/5/64; 3/1/64; 11/12/63. There must be two slash characters in the date field in addition to two characters for the year.

CARDS COLUMN Card. This card causes the Monitor to punch on-line column binary relocatable cards (without a loader).

LIST or LIST 8. Each of these cards causes the Monitor to write the object program in FAP-type language following the storage map printout. Both appear on the output tape. The LIST card produces listings in three columns without octal instruction representation; the LIST 8 card produces listings in two columns with octal representation of each instruction and its relocation bits. If both cards are used, the LIST 8 card takes precedence.

LABEL CARD. Labeling and serialization may be obtained on the off-line output cards of a FORTRAN compilation. The label is designated by the programmer in the following manner:

1. The contents of columns 2-7 of a card are taken as the label if (a) it is the first card of the program which does not have an asterisk in column 1, and (b) the card has a 'C' punched in column 1, and (c) at least one of the 2-7 columns does not contain a blank.

This label, with blanks treated as zeros, is then placed in columns 73-78 of the output cards, with columns 79 and 80 used for serialization. Serialization begins with the number 00 and recycles when the number 99 is reached. If, however, the label does not use all of the columns 73-78, serialization begins with zero and increases to 99. . 9, filling all remaining columns before it recycles.

If conditions a, b, and c (above) do not all hold, the labeling is applied as follows. For a subprogram, the name of the subprogram is used; for a main program, 000000 is used in columns 73-78. The LABEL card option corresponds to END card setting 7.

Labeling may be obtained on the off-line output cards of a FAP assembly. The information in card columns 2-7 of the FAP page title card will be used as the label. Serialization occurs as described above.

PRINT Card. This card causes the Monitor to print on-line all information being written on the off-line listing tape.

FAP Card. This card is placed immediately before the FAP program cards (source deck) that are input to the Monitor. It specifies that those cards are to be assembled by FAP. The FAP card follows any of the CARDS COLUMN, LIST, LABEL, or PRINT control cards that may be used.

END TAPE Card. The END TAPE card designates the end of the last monitor job. It must be a separate file on the input tape.

Other cards which are not strictly control cards and may be input to the Monitor include:

1. End Of File. This card is used only when input to the Monitor is on-line. It writes an end of the file mark which separates the jobs on the input tape. An end of file card is specified by a 7 and 8 punch in card column 1.
2. Cards with an asterisk '*' in column 1 may be included with the control cards but their information field will be tested and treated as if they were comments. When read, they will be printed on-line and written on tape for off-line printing.

IBSYS - THE BASIC MONITOR SYSTEM

It has been established that a computer can execute any job presented to it in the proper language. From this it can be concluded that within a given period of time, or computer run, it is possible to execute one or a series of individual jobs. The program utilized by a job within the series need not be identical to that used by the job previously performed.

During an operational run, a computer requires a controlling element which specifies the programs to be used and the sequence of their utilization. It was initially the duty of the system operator to intervene between each job. At that time he would wrap up the job previously executed and initialize the computer system for the processing of the next job. The preparation consisted of loading the program, input and output files, along with required console procedures. Control would then be passed to the program and the job execution begun.

Operator intervention, strictly a manual effort, is costly due to loss of computation time. This left open an obvious area for improvement. The solution to this problem centered about the transfer of responsibility of the operator over to the computer. Thus the concept of the programmed monitor originated. The function of a programmed monitor can be described as the control of the job execution on a computer during a given run. Its major objective is to increase the efficiency of computer operation by minimizing the time loss due to operator intervention.

The monitor in control of IBSYS is called the Basic Monitor and has control over all programming systems which are participants of IBSYS (Figure 69). In addition to those shown in the figure, all future systems released for the 7090/7094 will be incorporated. Provision for expansion of the system to the utilization of four reels of tape, for library purposes, has been incorporated.

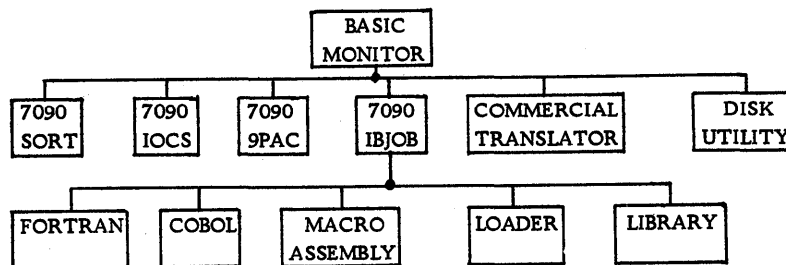


Figure 69 - 7090 IBSYS

Obviously, it is impossible for the Basic Monitor and all participating programming systems to be in core storage simultaneously. In view of this only a section of the monitor, called the Nucleus, remains in core at all times. Its remaining sections, as well as programs of participating systems are brought into storage as required.

FUNCTIONS OF THE BASIC MONITOR

The first function of the Basic Monitor can now be summarized as follows: to call programming systems of IBSYS in a predetermined sequence.

When a program is written to operate under Basic Monitor control, physical I/O device designations cannot be used. The programmer coding the package cannot foresee the I/O device utilization at the time his package is used within a run. Some devices may contain data from a previous job which are not to be destroyed; other units may not be operational at that time. These are only two of the many conditions that can exist. How then are I/O device needs within a program to be satisfied? This is accomplished in the following manner:

1. All input-output device specifications are made symbolically at the time the program is coded.
2. An account of the current status of each I/O device on a system is maintained by the Basic Monitor.
3. Routines are incorporated within the program which make input-output unit assignments at the time of execution by reference to this data made available by Basic Monitor.

Due to this procedure, only the devices available at the point of execution of the program within a system run will be utilized. It is therefore the second function of Basic Monitor to make available the current status of each I/O device on the system.

The Basic Monitor contains routines which should be utilized by any participating programming system to perform its input-output operations. These routines offer many advantages to their user in that:

1. The routines are coded and debugged and thus available for immediate usage.
2. Standard tape error recovery procedures are employed automatically when necessary.
3. All tape unit positions are constantly maintained and made available to the user. In view of this, the third function of the Basic Monitor can be considered the making available, to the user, of a complete set of debugged input-output routines.

IBSYS consists of a series of programming systems, which together with a monitor, are contained on a library tape or tapes dependent upon its overall size. A monitor system such as IBSYS would not be complete without providing a simplified method of updating its library. This is the fourth and last function of the Basic Monitor - to provide a means of modifying and updating the IBSYS library.

BASIC MONITOR SECTIONS

In keeping with these four functions, the Basic Monitor is divided into four distinct sections (Figure 70). The first is called the nucleus, symbolically represented as IBNUC. This section remains in core storage throughout a given run. It contains tables and key cells which provide data representing current conditions of the run. Among these are included the status of each I/O device as well as the programming system in control, date, etc. In addition, IBNUC has the responsibility of loading the second section of the monitor.

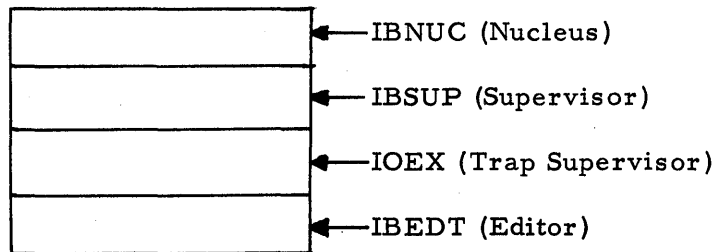


Figure 70 - Basic Monitor

The second section is called the supervisor, symbolically represented as IBSUP. This section is loaded when the initial programming system is to be called at the start of a run or a subsequent system is to be called during a run. It has the responsibility of processing all Basic Monitor control cards other than those used for edit purposes. Interpretation of a particular control card (\$EXECUTE) leads to the execution by IBSUP of another of its functions. Namely, to locate the desired programming system in the library and to initiate loading of the system's monitor by IBNUC which then transfers control to that system.

Modifying and updating of the IBSYS library is the responsibility of the third section of the Basic Monitor, called the editor which is symbolically represented as IBEDT. Editing and other maintenance procedures will not be covered in this manual.

Although considered to be contained within the nucleus, the trap supervisor or IOEX is functionally separate and therefore treated as a fourth section of the monitor. It is comprised of the routines available to the user for input-output operations. Similar to IBNUC, IOEX remains in core

storage constantly during a computer run.

SPECIALIZING THE BASIC MONITOR

The Nucleus section of the Basic Monitor has the responsibility of keeping an accurate account of the current status of each I/O device on the system. To accomplish this objective IBNUC contains a control area, called a Unit Control Block, for each device. The greater the number of devices accounted for, the greater the area of core consumed by Unit Control Blocks and the smaller the area available for use by the participating programs. In view of this it would not be feasible to include one of these areas for every possible I/O device that can be present on the largest 7090/7094 system. The ideal situation would be the representation of only those devices on the user's system. This would require a Basic Monitor tailored to suit the unit configuration at each installation.

THE NUCLEUS OF BASIC MONITOR

The Basic Monitor is divided into four sections. The Nucleus (IBNUC) and Trap Supervisor (IOEX) remain in core storage throughout a system run. The Supervisor (IBSUP) is loaded between job executions and the Editor (IBEDT) is called only when the library tape/s are to be modified in some manner.

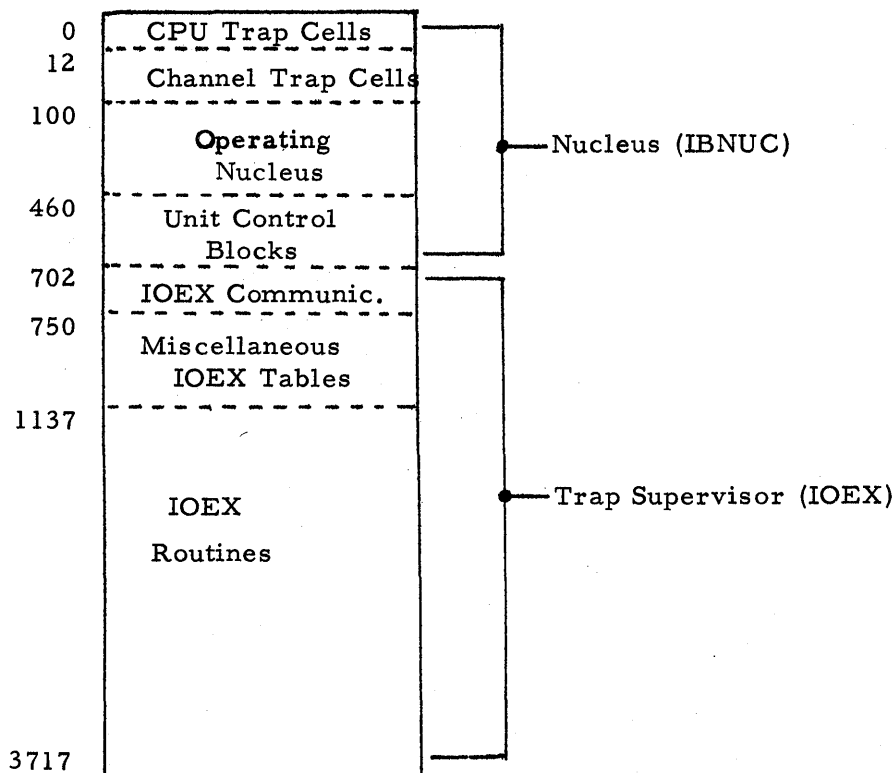


Figure 71 - IBNUC + IOEX

Figure 71 illustrates the placement of the Nucleus and Trap Supervisor in core storage. Further examination of the figure reveals a subdivision of the nucleus into four sections:

1. The trap cells which occupy storage locations 0 through 11; these cells are primarily utilized by C. P. U. initiated traps. (Example: STR)
2. The trap cells which occupy storage locations 12 through 61; these cells are utilized by channel initiated traps.
3. Cells 62 through 457 contain what is referred to as the Operating Nucleus; This is the area to be given prime consideration in this section of the text.
4. The Unit Control Blocks, one per I/O device on the system, follow in cells 460 through 701.

The core locations stated above apply to the basic monitor as assembled for a Standard 7090/7094 system. Upon modification of the package to suit another system configuration, the origin of the unit control block area will change.

Now that the major sub-divisions of the Nucleus have been established, consideration is to be given to that portion called the Operating Nucleus. A detailed examination of this section is shown in Figure 72 and described below.

1. The area called the One Word Entries Table consists of a group of cells which remain at the same core location despite 7090/7094 system configuration. By reference to specific cells within this table, the user can communicate with an area subject to a change in its location. Cells in the table area not utilized for communication purposes are used to store constants (Ex: DATE) which are of value to all participating programming systems.
2. The System Units Table will be considered in detail in the material to follow. Briefly, it is a table area used to keep account of certain I/O device assignments.
3. The Disk Limits Table is a supplementary area to the System Units Table required for disk usage. As disk will not be considered in this text, further discussion of this area is not required.
4. The UCB Locators contain the location of the first of the consecutive Unit Control Blocks for each channel, the total number of I/O devices and the number of card units on that channel. The UCB Locator area will therefore consist of one cell per channel.
5. Available Channel Locators sometimes referred to as the Unit Availability Table also consists of one cell per channel. Their

function lies in the accounting of the tape units which can be assigned by programming systems as required.

6. The System Loader area consists of the coding available to the user as well as the Supervisor for purposes of loading records of a programming system.
7. Interrupt Routine is a routine used to interrupt the monitor operation for the purposes of manual intervention.
8. Dump Calling Routine is a routine used to call a Dump Program which is made available by Basic Monitor.
9. IBSUP Restoration Routine is used to load and initiate execution of the supervisor when utilization of a programming system is no longer required and control is returned to the Basic Monitor.

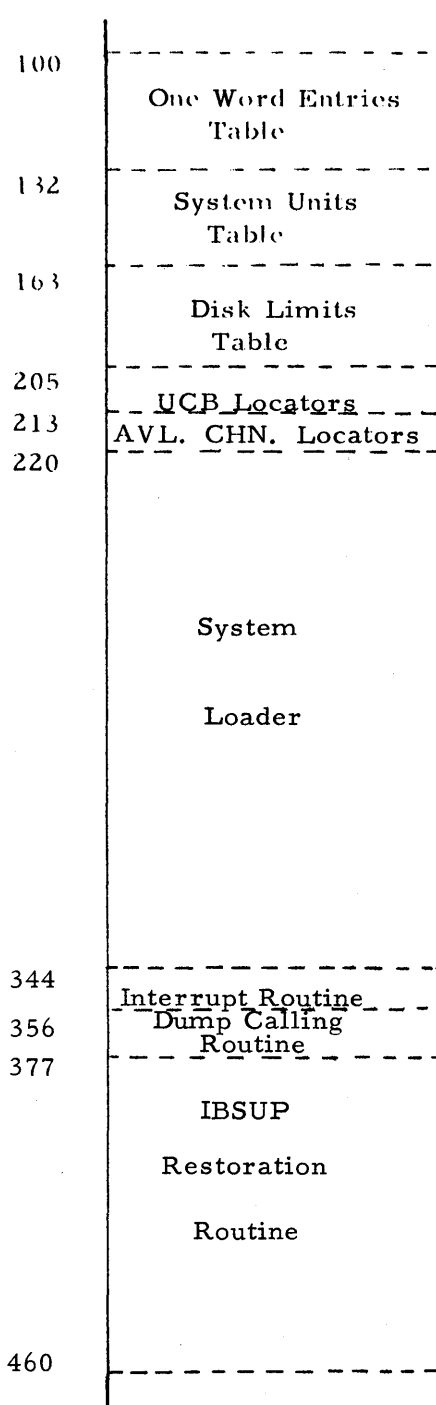


Figure 72. Operating Nucleus

Communication Region and One-Word Entry Table

The one word entry table serves two functions: First, that of a means of communication between the user and various areas of the nucleus; and second, to provide a fixed set of locations to store constants of value to all participating programming systems.

The table area uses 26 consecutive core storage cells as shown in Figure 73.

ADR	SYMBOL	FUNCTION
100	SYSTRA	Transfer instruction to current system
101	SYSDAT	Date Word
102	SYSCUR	Name of current system
103	SYSRET	Location to which each system returns
104	SYSKEY	Contents of entry keys at cold start
105	SYSSWS	Contents of sense switches at cold start
106	SYSPOS	Initial position and index of current system
107	SYSUNI	Location and length of SYSUNI and Disk Limit Table
110	SYSUBC	Location and length of table of UCW locators by channel.
111	SYSUAV	Location and length of SYSUAV chains by channel.
112	SYSUCW	Location and length of Unit Control Blocks (UCB)
113	SYSRPT	Transfer to job interrupt routine
114	SYSCEM	Transfer to CE diagnostic routine
115	SYSDMP	Transfer to bootstrap for SQUEZY core storage dump
116	SYSIOX	Location and length of IOEX reference entry table
117	SYSIDR	Transfer to installation accounting routine.
120	SYSCOR	Lower limit of usable core storage in decrement, upper limit in address
121	SYSLDR	Transfer to system scatter-load routine
122	SYSACC	Location and length of block of accounting information
123	SYSPID	Flag word for installation accounting routine
124	SYSCYD	Channel commands for system to copy and disconnect
125	SYSCYD+1	
126	SYSSLD	Self-load Sequence
127	SYSTCH	Self-load Sequence
130	SYSTCH+1	
131	SYSTWT	System trap, wait and transfer point

Figure 73. One Word Entry Table

Names and functions of these cells are:

SYSTRA. Gives program control to the monitor of a participating programming system. A transfer instruction to the starting point of execution is loaded into this cell as part of the programming system monitor. On completion of the load of the monitor, IBSUP executes cell SYSTRA, which then initiates programming system operation.

SYSDAT. The current date is entered into this cell by a control card.

SYSCUR. The name of the current programming system is entered into this cell by a control card.

SYSRET. When the utilization of a programming system is completed and control is to be turned back to the basic monitor, a transfer is executed to this cell. As a result the supervisor will be called.

SYSKEY. The condition of the entry keys at the start of a run is saved in this cell.

SYSSWS. The condition of the sense switches at the start of a run is saved in this cell.

SYSPOS. When the supervisor (IBSUP) is about to locate a desired programming system on the library tape it stores the following information into cell SYSPOS.

- 1- address, the number of the library tape (1 through 4) the programming system is located on.
- 2- decrement, the number of files to be skipped from load point to reach the system.

SYSUNI. Contains the location of the system unit table in its address and the number of cells used by the table, in its decrement. As mentioned earlier this table is used to keep account of certain I/O device assignments and will be covered in detail in the material to follow.

SYSUBC. Contains the location of the UCB locators in its address. As mentioned earlier, these locator cells, one per channel, locate the first of the consecutive unit control blocks for each channel.

SYSUAV. The address of this cell is used to locate the available channel locators and its decrement to find the number of core cells occupied by the locators. As mentioned earlier, these locators are used for the accounting of available tape units on each channel that can be assigned by programming systems as required.

SYSUCW. Contains, in its address, the location of the first unit control block and in its decrement, the number of cells occupied by all unit control blocks (representing all channels).

SYSRPT. Participating programming systems of IBSYS normally allow an interrupt of monitor operation at various points within the flow of a program for purposes of manual intervention. An example of the need for this feature is the interruption of monitor control by an operator, between two jobs utilizing the same programming system, for purposes of altering some I/O assignment.

Cell SYSRPT contains a transfer to the interrupt routine discussed earlier in the section on the operating nucleus. Due to the availability of cell SYSRPT, the programmer coding a system need merely insert a TSX SYSRPT, 4 wherever an interrupt point is desired. At execution time, the interrupt routine will test sense switch 1 to determine whether or not an interrupt is to be executed at that point. If so, control is temporarily returned to the basic monitor which will perform the desired action.

SYSCEM. May contain a transfer instruction to a Customer Engineering routine used as an aid in diagnosing system failures. Normally this cell contains a TRA 1,4 which will result in an immediate return to the main program.

SYSDMP. Contains a transfer to the dump calling routine discussed earlier. When executed this routine will call a dump program from the IBSYS library and perform a full mnemonic dump of core storage. On completion, control will be returned to the programming system that initiated the dump.

SYSIOX. Communication between the user and the trap supervisor (IOEX) is established through the communication region of IOEX. This region occupies an area within IOEX starting at the location specified in the address of SYSIOX and extends over the number of cells given in the decrement.

SYSIDR. Installation accounting is used by a customer to keep account of job processing on a computer system for departmental billing purposes. The procedure involves the printing of sign-on and sign-off messages identifying each job, along with the associated programmer, department, running time, etc. Cell SYSIDR normally contains a TTR 2,4; however, if installation accounting procedures are to be used, its content will be changed by the user to a transfer to the accounting routine he supplies.

SYSCOR. Contains the limits of the core area available to programming systems. These limits take into consideration the cells utilized by the ever present areas such as IBNUC. The address of SYSCOR specifies the highest available location of core or System End (SYSEND), the decrement specifies the lowest available location or System Origin (SYSORG).

SYSLDR. Contains a transfer to the system loader area mentioned in the discussion of the operating nucleus.

SYSACC. Contains in its address the location of a block of data used for system accounting. Its decrement specifies the length of the block. The data consists of descriptive information, such as the date, time, program name, etc., required by the accounting routines.

SYSPID. To indicate that system accounting is to be performed this cell, which serves as a flag, is set by means of a basic monitor control card (\$ID).

SYSCYD. This cell is the first of two in succession which contain channel commands used for loading from disk.

SYSSLD. Contains a channel command (IOCP *+1,,1) made available for loading purposes.

SYSTCH. This is the first of two consecutive cells. The first is used for loading from disk. The second contains a TCH SYSSLD, made available for loading from tape.

SYSTWT. This cell is used in conjunction with the 7909 channel.

UNIT CONTROL BLOCK

Each I/O device on a 7090/7094 system is represented in the nucleus by a unit control block. The unit control blocks (UCB's) for all devices are clustered together into a common area. This area can be located by referring to cell SYSUCW of the one word entry table which not only contains the location of the area but also its length. UCB's associated with a particular channel can be located by reference to the appropriate cell of the UCB locators. Furthermore, the UCB locators can be located by reference to cell SYSUBC of the one word entry table. Since the function of a unit control block is to keep an account of the current status of each I/O device, four cells are allocated to each UCB. An examination of the content of each cell is desirable at this time. These cells are shown in Figure 74.

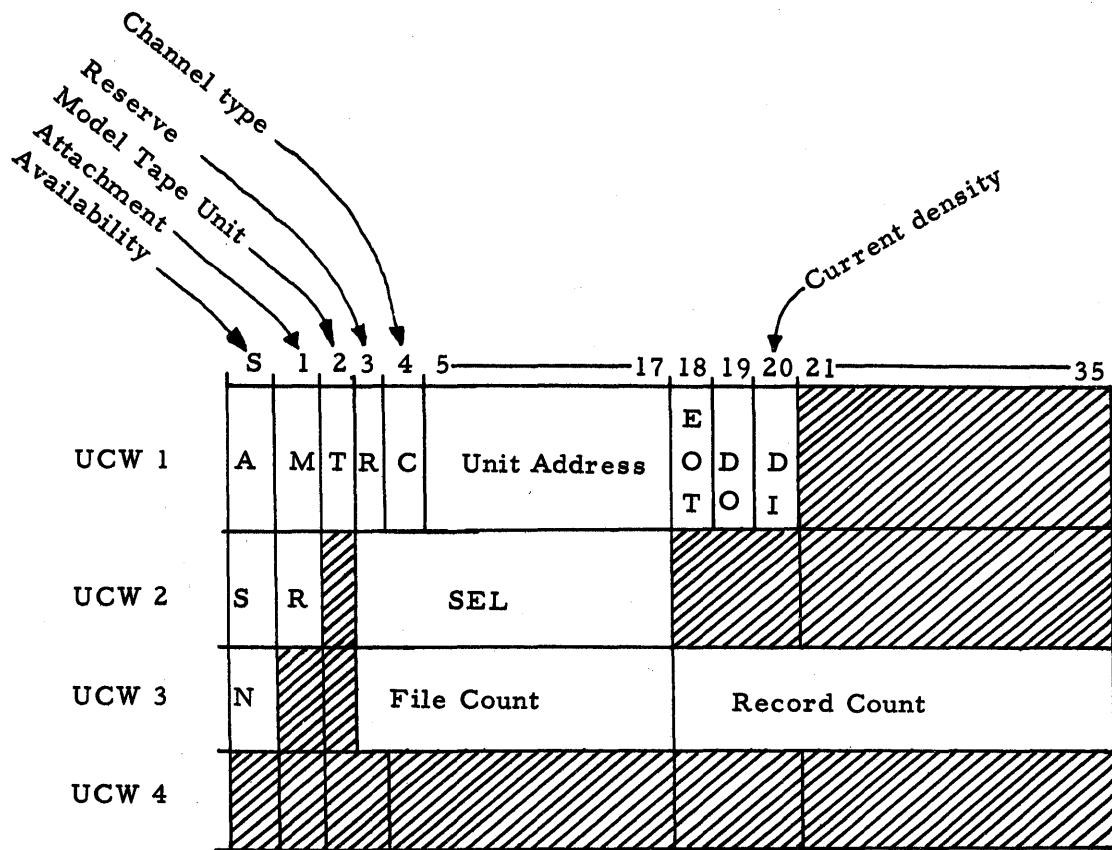


Figure 74. Unit Control Block

Unit Control Block Word 1 (UCW 1)

(A). (Position 5) - This position is called the availability flag. If it is set to 1 the unit is available for assignment by a programming system.

(M) (Position 1) - This position can indicate one of two conditions dependent upon the availability status of the unit:

1. If the unit is available for assignment ($A = 1$), this position becomes an attachment flag. A unit is said to be attached ($M = 0$) if it is physically connected to the computer system. When off-line for maintenance or any other reason it is considered detached ($M = 1$).
2. If the unit is not available for assignment it must be in use. Under these conditions, position 1 becomes a restart repositioning flag. The term restart applies to the concept of continuing a job which was interrupted at some earlier time. At that time a checkpoint would have been taken which saved the entire content of core storage as well as the condition of all major registers and keys. During the restart, all conditions that existed at the point of interrupt are restored from the data saved by the checkpoint. The conditions to be restored consists of core storage content, condition of all major registers and keys and position of all tape units. The latter is under control of the restart repositioning flag in the unit control block of each I/O device. By this means tape units which are not to be repositioned on restart for some reason can be deferred from doing so.

(T) (Position 2) - This position of UCW 1 specifies the model of the tape unit as either II or IV. Distinction cannot be made between Models II and V or IV and VI.

(R) (Position 3) - This position is called the reserve status flag and indicates whether or not the unit is reserved. Reserved units will be covered in detail later in this section of the text.

(C) (Position 4) - This position describes the channel type the associated unit is attached to as either a 7607 or 7909.

(Unit Address) (Positions 5 - 17) These thirteen bit positions contain the physical I/O address of the associated device. Tape unit addresses are always those which specify BCD mode of operation.

(EOT) (Position 18) - This position acts as an end of tape flag which is set when an end of tape has occurred while writing. It remains set until the associated tape unit is rewound.

(DO) (Position 19) - This position specifies the tape density at load point.

(DI) (Position 20) - This position specifies the tape density at the current position.

Positions 21 through 35 - These positions of UCW 1 are normally available to the user for his utilization; however, an exception to this rule exists for reserve units. Reserve units will be discussed in detail later in this section of the text.

Unit Control Block Word 2 (UCW 2)

Word 2 of a unit control block has a dual function: First, that of a switch which is tested by basic monitor (IOEX) to determine if the associated I/O device is to be activated. If the decrement is equal to zero the unit is to remain dormant, if it is non-zero an I/O operation is to be initiated on the unit; and second, if an I/O operation is to be initiated, word 2 supplies information required by IOEX to perform the operation. A breakdown of the information supplied is:

(S) (Position S) - This position is set by the user to indicate the type of I/O operation (read or write) to be performed on the associated device.

(R) (Position 1) - This position called permanent redundancy message control is set by the user to instruct basic monitor to either print or omit printing a message if a permanent read redundancy occurs on the associated tape unit.

(SEL) (Positions 3-17) - This area of the second word is specialized by the user to inform basic monitor (IOEX) of the location within the programming system of the select routine to be used to activate this I/O device. Input/Output operations utilizing the IOEX section of basic monitor involve complicated interconnection between IOEX routines and the programming system.

Unit Control Block Word 3 (UCW 3)

(N) (Position S) - This position of the third word serves as a noise record flag. Any tape record shorter than three words in length is considered a noise record. If this type of record is encountered during a read operation, the flag is set by basic monitor (IOEX) as an indication of its occurrence.

(FILE COUNT) (Positions 3 - 17) - A count of the file marks written or read from the associated device is maintained in the decrement by IOEX.

(RECORD COUNT) (Positions 18-35) - A count of the records read from or written onto the associated I/O device within the current file (as specified by the file count) is maintained in these positions by IOEX.

The details of the content of a unit control block given above pertain to one associated with a tape unit. UCB's associated with card equipment will be similar in all areas except those which obviously do not apply to these devices (Ex. : Model). Disk UCB's vary extensively and require the utilization of a fourth word. Disk applications will not be considered in this text; therefore, investigation of disk UCB's has not been included.

AVAILABLE UNITS

During the discussion of the communication region and one word entry table, cell SYSUAV was said to contain the location and length of the area in the nucleus called the available channel locators or unit availability table. The function of this area was described as the accounting of all tape units (available units) which can be assigned by programming systems as required. This table consists of one cell per channel (Figure 75); the first for Channel A, the second for Channel B, etc. By reference to the appropriate cell of the table, a participating programming system must be able to locate any number of available units on the desired channel. To accomplish this goal the concept of chaining is employed. Chaining is best described as a method of tying together areas of core storage associated with a common function. As in actual physical chains there must be a starting link. In the case of the unit availability chain, as the one being considered at this point is called, the first link can be found in the unit availability table cell. The address of this cell contains the location of the unit control block of the first available unit. The address portion of the first word of this UCB forms the second link of the chain by containing the location of the UCB of the next available unit on the channel. This process is continued until all available units on the channel, associated with the cell of the unit availability table under consideration are linked together. The UCB of the last available unit contains zeroes in the address portion of its first word to identify the end of the chain. A similar chain is in existence for each channel on the system.

These chains are initially set up by basic monitor. When a programming system requires tape units, it is its responsibility to interrogate the appropriate chains, remove the unit reference from them and then update them. On completion of a job, it is the responsibility of the programming system to return all units that are no longer required to the chains. There are times when the basic monitor is in control that it will alter the status of the availability chains, due to the interpretation of certain control cards. These instances will be covered during the discussion of control cards in the following chapter.

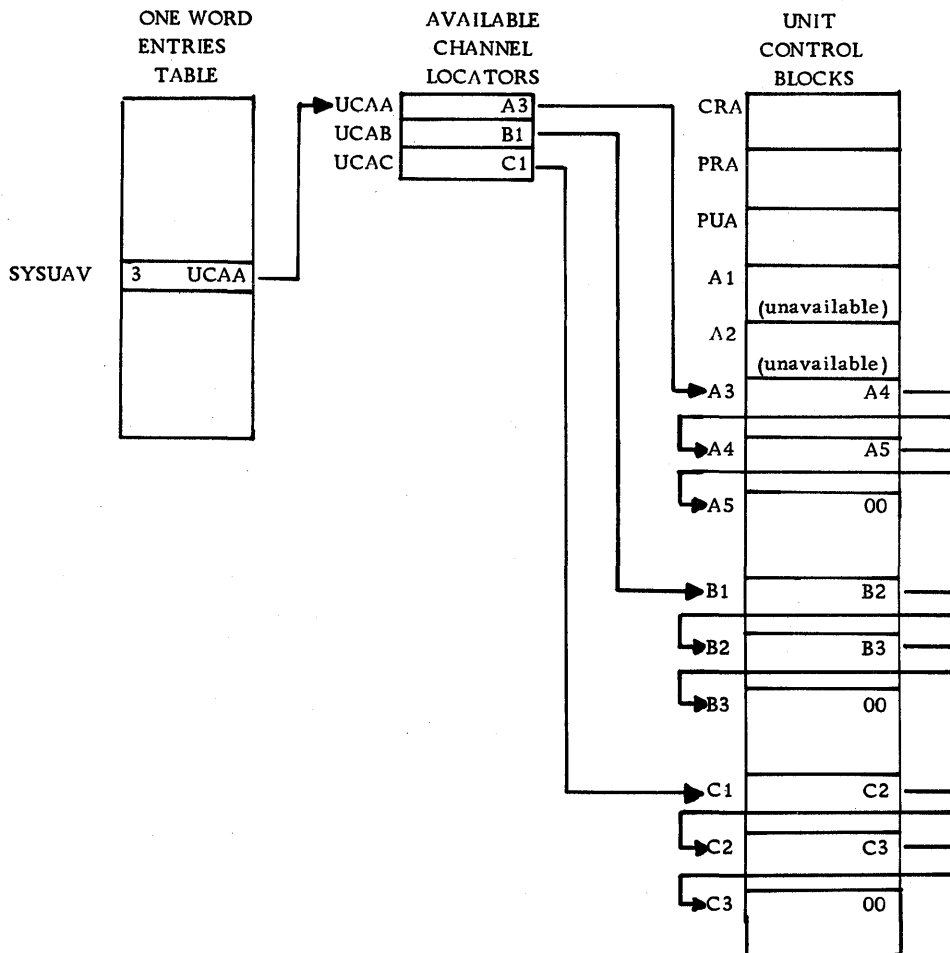


Figure 75. Unit Availability Chains

UNITS THAT ARE NOT AVAILABLE

All units available for use by programming systems are tied together via the unit availability chains. Consideration is now to be given to those units which are not considered available. These units can be grouped into three categories as follows:

Detached Units

Each I/O device on a 7090/7094 system is represented in the nucleus of basic monitor by a unit control block. Reference is made to the UCB's whenever I/O device assignments are to be made. Even though a desired device is represented by a control block, it could not be used if temporarily disconnected from the 7090/7094 system. Perhaps the particular I/O device was taken off-line for service or preventive maintenance or if it is a tape unit, it might have been loaned to another system in the installation. In either case, some means had to be incorporated within basic monitor to flag a unit in this detached status. In the earlier discussion of the U.C.B., word 1-bit position 1 was found to act as an attachment flag. This flag normally indicates that the associated I/O device is attached to the system; however, by means of a basic monitor control card it can be altered to indicate a detached status. When returned to the 7090/7094 system the device can again be flagged attached by a basic monitor control card.

Function Units

The devices attached to a system fall into two distinct categories: equipment (card devices) and tape units. Tape units which are available for assignment by a programming system are tied together into one of a number of unit availability chains. What of the card devices and remaining tape units? Consider them one at a time. First, each of the remaining tape units will fall into one of two groups, function or reserve. Second, the card devices will also fall into the function group.

During any system run there are a number of quantities of information common to all jobs that should be gathered together on the same I/O device. For example, the IBSYS library where all programming systems are located; the input control data to the basic monitor and programming systems; the output of all jobs; or the output to be processed on off-line tape to punch equipment. For this purpose basic monitor has incorporated within its nucleus a means of associating a particular I/O device to each of nineteen possible functions. This means consists of a nineteen word table area called the system units table or system units function table. This table is located within the nucleus and can be found by reference to cell SYSUNI of the one word entry table.

The system unit table is set up as shown in Figure 76. Each cell is associated with a particular function and is given a symbolic reference accordingly. The first four cells are associated with the library function of IBSYS. Note that these cells, represented symbolically as SYSLB1(System

Library 1) through SYSLB4, allow four tape assignments for the expansion of IBSYS to its maximum size.

The next three cells are associated with the card equipment. These are followed by two cells each for system output (SYSOU), system input (SYSIN), peripheral punch (SYSP) and checkpoint (SYSCK) function. Two units are allotted to each of these functions for reel switching purposes. As soon as one reel is completely processed a switch is made by the user to the alternate device. This eliminates the loss of time that would be due to rewinding and tape handling. The first unit is called the primary unit, and the alternate the secondary unit.

The last four cells are used for system utility (SYSUT) assignments. Each cell has in its address the location of the unit control block of the device assigned to that specific function. The sign of each word specifies the density of the associated data. Assignments need not be made to all functions at all times, in which case the associated table cells will be set to zero. Assembly parameters are responsible for the initial assignment made as shown in Figure 76 . These may be altered and additional assignments made via Basic Monitor control cards whenever the monitor has control.

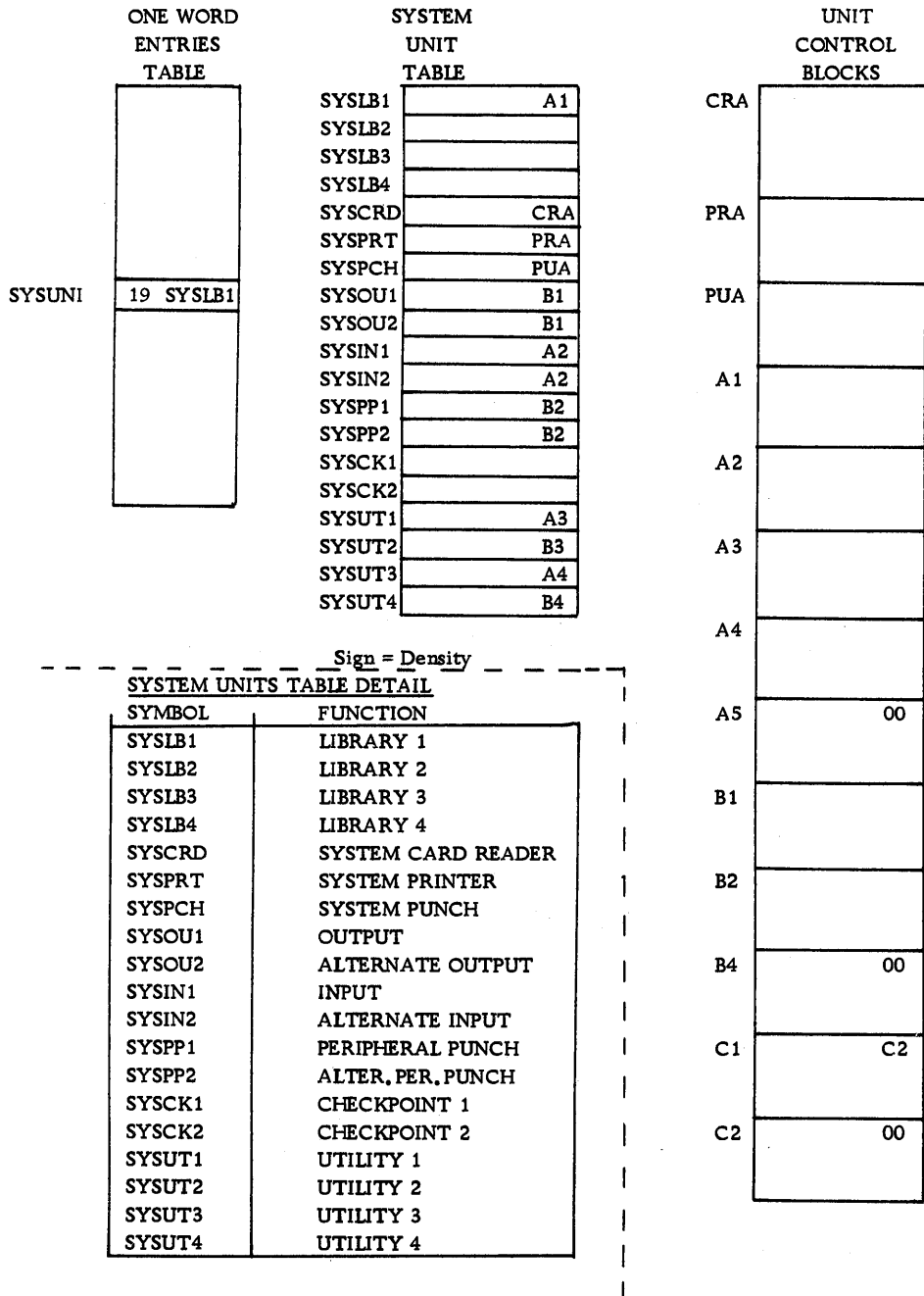


Figure 76. System Units Table

RESERVE UNITS (INTERSYSTEM)

Assume that a run consisting of a IOCS job followed by a SORT job is to be executed. In addition, assume that the output of the IOCS job is of an intermediate nature and requires processing by SORT to be finalized. The output of the first job (IOCS) must therefore serve as input to the second (SORT). Being of an intermediate nature, the output of the IOCS operation will not be placed on a function unit. In view of this, a tape unit for this application must be selected by the IOCS program from the unit availability chains.

A reserve unit can be defined as a tape unit that is withheld from the availability status or reserved for purposes of data transfer between jobs utilizing different programming systems. Thus, it is often referred to as an intersystem reserve unit. This type of unit is considered symbolically to be located on any one of channels J through Q. The symbolic designation of the tape unit to be used for the output of the IOCS job mentioned above will be given as J1 or Q5 or K4, to list a few of the possibilities at random. The input tape unit symbolic designation for the SORT job will have to agree to that given for the output unit of the earlier job. It too will be specified as J1 or Q5 or K4, etc.

When the IOCS package interprets the reserve specification for its output unit it acquires an available tape device from one of the unit availability chains. It will then set the reserve status flag in the associated unit control block (UCW 1 - position 3). This device is now reserved for the specific output data which will not be disturbed by any other operation. The BCD representation of the designation is stored into the address portion of word 1 of the unit control block to relate this unit to the symbolic designation given. The first job (IOCS) is now performed and the desired output written on the selected tape unit.

The execution of the second job (SORT) follows and is begun with the processing of unit assignments. Interpretation of an input unit specification of a reserve unit (ex. - J1) initiates a search of all U.C. B.'s. When a reserve unit is found with the desired BCD representation in the address portion of word one (UCW1), it is assigned to the input function.

The unit can remain in a reserved status as J1 indefinitely, and therefore can be referenced by any number of jobs that follow. When it is no longer required as such, it can be released to its availability chain by appearing as an input unit specification of J1R. On interpretation of the R (release) the programming system will not only utilize its content but in addition on completion of the job will return the unit to the appropriate availability chain. The release specification must be used in association with the last job utilizing the reserved unit as an input device.

Note that the basic monitor merely supplies the means of reserve unit handling to the participating programming systems. This means consists of

the reserve status flag (UCW1 - position 3) and an area to place the BCD representation of the symbolic designation given (UCW1 - address). The actual processing of the designation along with the specialization of the first word of the unit control block is the responsibility of routines within a programming system.

SYMBOLIC UNIT DESIGNATION

It has been established that it is the responsibility of a programming system of IBSYS to interpret symbolic I/O device designations⁽¹⁾ and utilizing information made available by Basic Monitor make I/O unit assignments. This procedure must be followed by all participating programming systems.

This brings up an interesting point. Since each system processes its own designations, the symbolic representation of a specific device by one programming system obviously need not be identical to that used by another to specify the same physical unit. If this haphazard arrangement were encouraged, the result would be discouraging to say the least. As an aid toward standardization of these symbolic designations, suggested symbolics for each device classification are given.

First of all, consideration is to be given to the designation of tape units of all classifications discussed, except function. The symbolic representation of these units is of the type that is readily associated with tape units as a channel is specified followed by a unit number (Ex. A1). First consider the channel specification.

There are three groups of channels a tape unit designation can fall into. One group is associated with real channels, the other two with logical channels. The real channel group consists of the normal channel specifications of A through H. For example, a unit specification of A would be interpreted by a programming system as a request for any available tape unit on physical channel A. Designations involving any of the other real channels would be handled accordingly. The utilization of the first group of logical channels was seen earlier during the discussion of reserve units. There it was found that logical channels J through Q were utilized for reserve unit specifications. The second group of logical channels are specified as S through Z. These are associated with a rather unique scheme of assuring that tape units, common to particular operations, utilize the same channel. Grouping of I/O operations in this manner can result in increased computer efficiency due to maximum I/O overlap between channels. When assignments are made, all designations utilizing one of these channels (S-Z) are accumulated. A channel is then chosen which has a sufficient number of available tape units. In this manner the association of all similar logical channel designations (S - Z) with the same physical channel is assured.

(1) These designations are normally supplied to the programming system by a control card.

The channel letters of any of the above groups can be followed by a unit number (Ex. A 2). This number is merely used to differentiate between units associated with a particular channel designation.

Furthermore, the unit number can be followed by the unit model (Ex. - A2 II). The unit model can be specified as II for models II or V, and IV for models IV or VI. Along the same lines, do not forget that the channel and unit specification of a reserve unit to be used for input purposes can be followed by an R (Ex. - K3R). This indicates that the unit is to be released to its availability chain.

In summary, an I/O designation of C 3II will result in the selection from the availability chain of channel C a model II tape unit. All references to unit C3 will be associated with the same physical unit. In addition, it should be pointed out at this time that if any of the above designations cannot be satisfied, a substitution is made if a tape unit is available. For example, if a model II drive were not available for the designation of C3II above, a model IV would be used. If no units were available on channel C, one would be selected from another channel.

System function units are specified by using the first two characters of the function; OU for system output, IN for system input; etc. In addition, a specification of OU will result in the assignment of SYSOU1 as the primary output unit and SYSOU2 as the secondary unit. This would also hold true for PP, IN and CK designations.

Finally, if a blank designation is given, any tape unit is considered satisfactory for that usage. An * (asterisk), a secondary unit designation only, will result in an assignment of the secondary unit identical to that of the primary.

PROGRAM EXECUTION UNDER BASIC MONITOR CONTROL

When any participating programming system of IBSYS is to be utilized for a job, a minimum of two inputs is required as shown in Figure 77.

The first is the IBSYS library which contains the program routines of the basic monitor as well as all participating programming systems.

The second is the job input which contains control cards that provide instructions to the basic monitor and programming systems used. The job input may in addition contain this information as determined by the needs of the particular programs being executed. Though the job input is shown as tape in Figure 77, it may also be in punched cards.

Let us now consider each of these two inputs in detail.

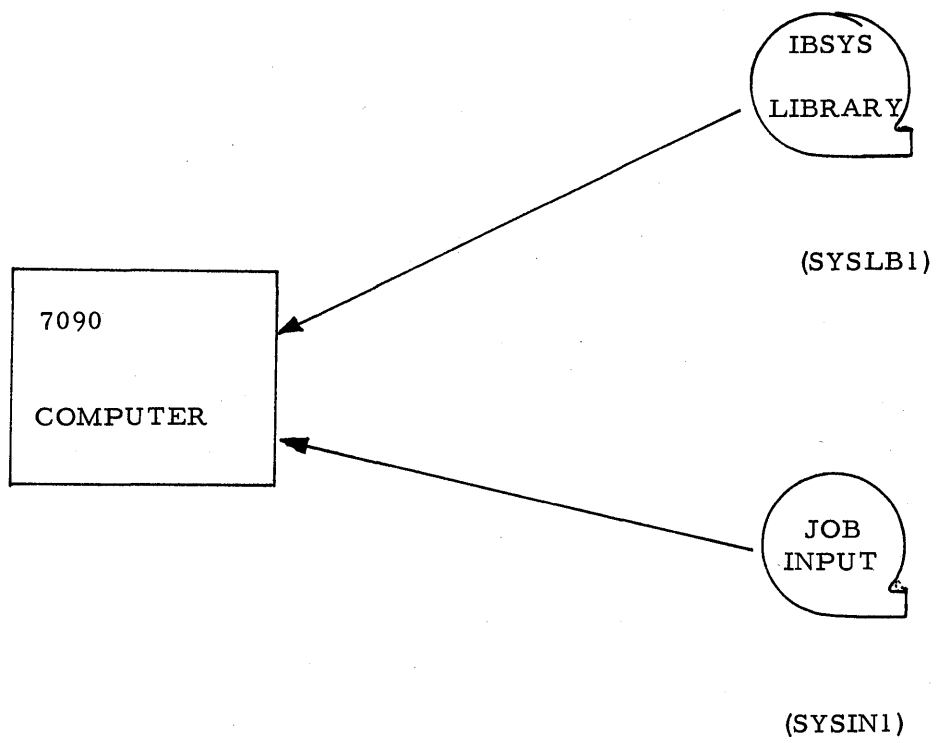


Figure 77.

The IBSYS Library

The IBSYS library contains routines of basic monitor as well as those of the programming systems. Though currently contained within one reel of tape, provision has been incorporated within IBSYS for expansion of its library to four reels. Function unit assignments SYSLB1 through SYSLB4 have been allotted for this purpose.

Visualize a hypothetical IBSYS library containing only one programming system, 9IOCS. The format of the library tape for this limited IBSYS package is shown in Figure 78. Note that the tape layout shown contains only those routines comprising the basic monitor and 9IOCS. Each record of the library is identified on a tape record format listing created by the basic monitor editor at the time the tape was generated. This was obtained through the use of a MAP option specified in the *EDIT control card.

Starting at load point, the first record encountered consists of 7 words and is a bootstrap type loader which initiates basic monitor loading when the 7090/7094 load tape key is depressed. Each record of the IBSYS is identified by the BCD configuration of its second word. This identification is transferred to the MAP printout at edit time and scatter loaded into cell SYSCUR of the one word entries table at execution time. As a result of this procedure the loader is identified as record 5U002 in the MAP which in this case is not a true BCD identification but rather an ENB 00002 instruction. A bootstrap could not of course surrender its second word for true identification purposes. The loader is not shown in any of the current basic monitor listings; therefore, a tape dump of it is included as Figure 79.

Record 2 of the first file is called IBSYS and is composed of the basic monitor supervisor (IBSUP), nucleus (IBNUC) and trap supervisor (IOEX). This record will be loaded at three distinct times. First, at the initial depression of the load key (cold start); second, by the IBSUP restoration routine of IBNUC when control is to be returned to basic monitor between two jobs utilizing different programming systems or when an interrupt of monitor operation is desired; third, when IBNUC is to be restored as specified by a basic monitor control card.

The IBSUP and IBNUC portions of this record are loaded into storage cells beginning at location SYSORG. The IOEX portion is scatter loaded directly into its fixed locations. During a cold start or restore operation the nucleus (IBNUC) is relocated from its temporary location above SYSORG to its final location. In addition, the unit control blocks are created. When control is returned to the basic monitor between jobs or when an interrupt is specified. IBNUC is not relocated and the unit control blocks are not generated.

In this manner, communication between jobs can be maintained.

In addition to these operations, IBSUP performs the processing of the basic monitor control cards; finds the location of the desired programming

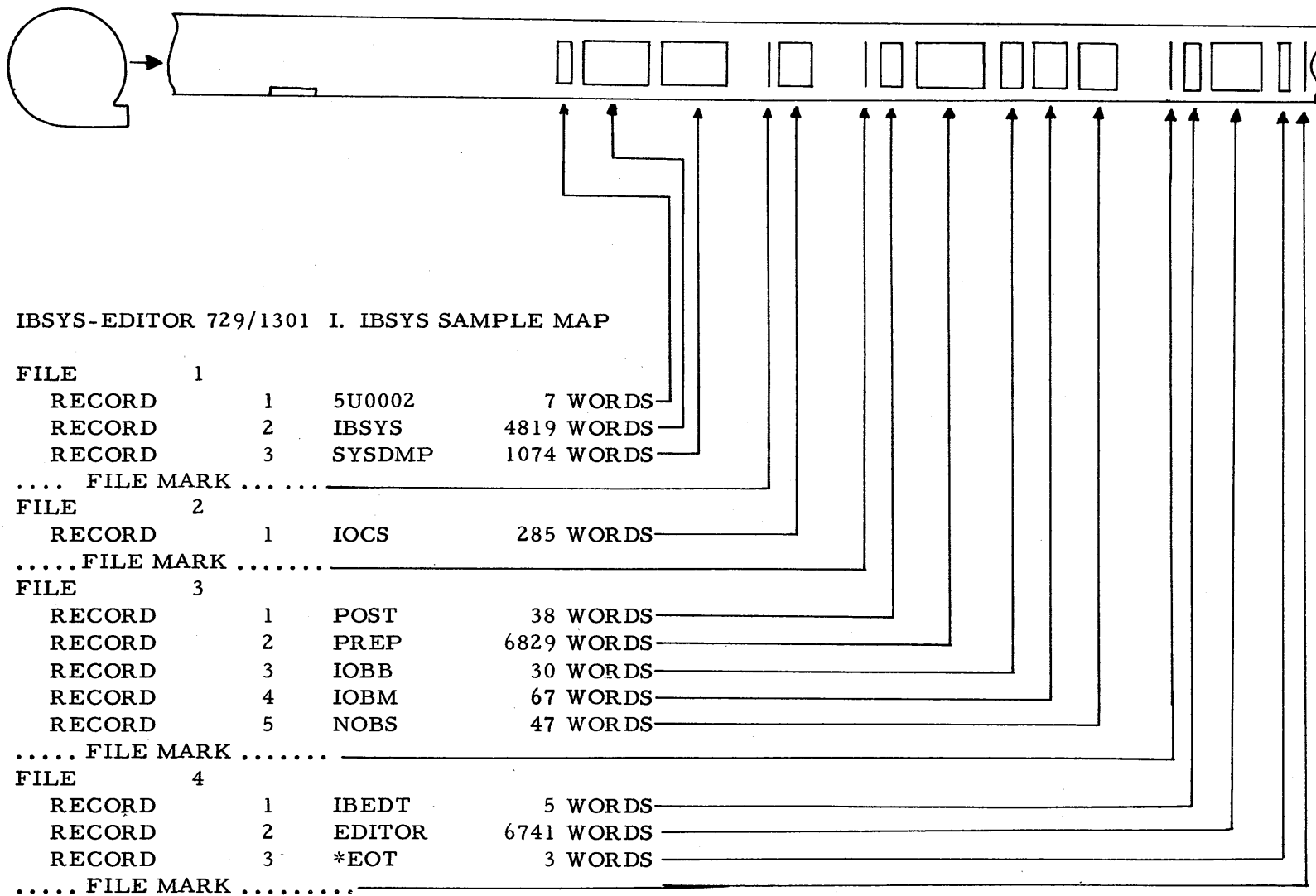


Figure 78

BINARY TAPE 01221 REWIND RDS (00000) 8 FILES RDS (00000) 8 RECORDS
RECORD 00001 (00007) 8 (00007) 10 WORDS
00000 -200002000001 056400000002 042000000377 -000001000000 100000000000 -000001000001 100000000000
RECORD IN BINARY

560

Figure 79

system and finally initiates its load and execution via the System Loader Routine within IBNUC. Once the load of the desired program has been initiated, IBSUP's presence in storage is no longer required and it can be overlaid by the programming system called.

Record 3 of the first file is the dump routine which is loaded via the dump calling routine in IBNUC when call SYSDMP of the one word entries table is executed.

File 2 contains the first record of 9IOCS which can be considered its individual monitor.

File 3 contains the remaining records of 9IOCS.

File 4, records 1 and 2 are called when the edit function of the basic monitor is to be utilized. These records are called by the supervisor (IBSUP) when it interprets a control card specifying an edit of the IBSYS library.

Record 3 of file 4 marks the end of the IBSYS library.

An IBSYS library containing additional programming systems will have the same format. However, the additional systems would be inserted along with 9IOCS as a series of files between the first and last file on the tape.

The Job Input

Now that the IBSYS library tape has been considered, let us investigate the job input.

It was pointed out earlier that the job input can be entered by means of card or tape. Though normally on tape, due to time saving considerations, card input is considered here for purposes of illustration. Figure 80 illustrates a typical input card deck for a run consisting of two jobs. It should be recalled at this point that the job input normally contains control information to the basic monitor as well as to the programming systems. In addition, other information, as required by the particular programming systems being executed, is included.

If the run shown in Figure 80 were to be executed, the IBSYS library tape would first be mounted and loaded and the job input deck readied in the card reader. On depression of the load tape key (cold start), the basic monitor would be loaded from the IBSYS library. IBSUP, after cold start initialization procedures, would load and process each basic monitor control card of the job input. These cards are identified by a \$ in column 1.

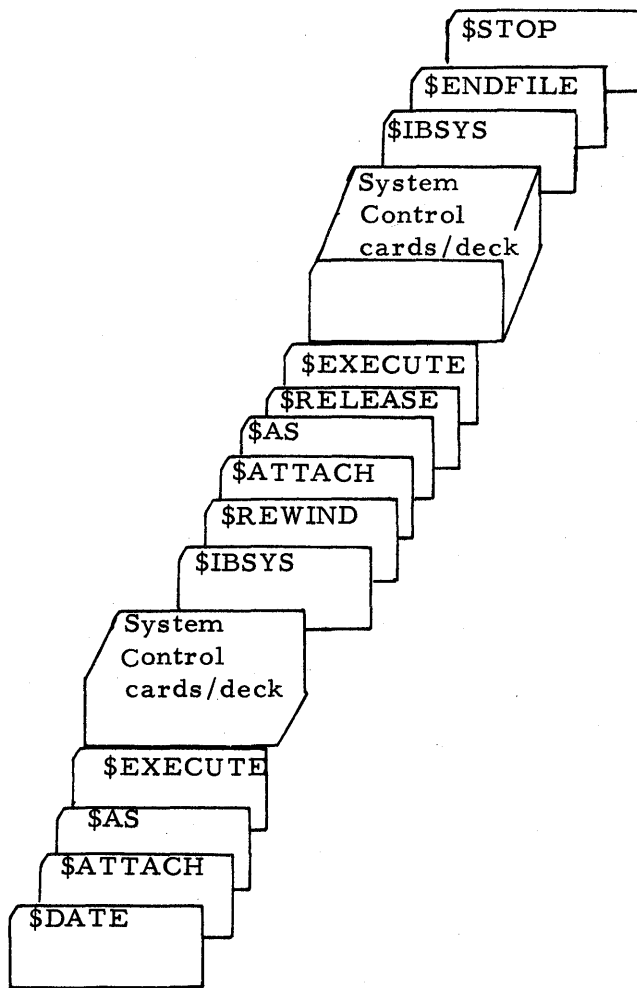


Figure 80.

In Figure 80, the first four cards are basic monitor control cards. Note that the last of the four is a \$EXECUTE card. This type of card specifies the name of the programming system to be loaded from the IBSYS library tape. On its interpretation IBSUP would initiate loading of the programming system and then transfer control to it. During the course of execution, the programming system would process its individual control cards as well as any other data included within the area identified as system control cards/deck in Figure 80.

Though the card following this area is a \$IBSYS card, it would actually be interpreted by the programming system. It is one of the few cards prefixed by a \$ that fall into this category. On interpretation of this card, the programming system would execute cell SYSRET of the one word entries table causing the supervisor (IBSUP) to be loaded and control returned to it.

The following five cards would supply control information to the basic monitor. Note that the last of this group is again a \$EXECUTE card which would result in the loading and execution of the programming system to be utilized for the next job.

The card group associated with this job would be processed during the course of its execution. The \$IBSYS card following the group would again return control to IBSUP of the basic monitor on completion of the job.

The remaining two cards would then be interpreted by the supervisor. The last of these (\$STOP) is indicative of the end of the run at which time basic monitor would inform the operator, End of Jobs- Cannot Proceed.

The job input for any run would be similar in overall appearance and concept except for the possible insertion of any number of individual job card groups. If the same programming system is to be utilized for two consecutive jobs, return to basic monitor between them is not necessary.

If the job input is to be on tape, the deck arrangement discussed above would merely be placed on tape by means of an off-line operation.

Running Under Basic Monitor Control

The two tapes discussed to this point are the minimum inputs required to initiate any basic monitor run. In addition, data input and output, scratch, punch output, print output, etc. would be required as determined by the needs of the jobs being executed during a given run.

A basic monitor run has been considered earlier from the job input processing point of view. Let's now view it from the operator aspect.

To initiate system operation and perform a cold start, the system operation would ready the IBSYS library tape and job input tape or cards as the case may be. He would then set sense switch 1 accordingly. This switch is placed down if the basic monitor control cards are in the card

reader and up if on tape. Depression of the load tape key would then result in the loading of basic monitor.

During the course of execution of the various programming systems required to perform the specified jobs, the operator would be instructed by on-line message printouts. These would specify the location and description of the tapes to be mounted and unmounted; report on the progression of the run; point out error conditions and the procedures to be followed when an error occurs, etc.

From this it can be seen that once the operation has been initiated, the operator need merely follow instructions presented to him on the on-line printer. This eliminates loss of computer time due to the familiar searching for job set up sheets; the making of errors due to lack of communication; failing to inform the operator of a change of setup procedures and so forth.

Control Cards

The job input source contains control cards of both basic monitor and the programming systems. Those of a programming systems are related to a particular system and therefore would have to be discussed in a coverage of the related programming system. In view of that, this section will be devoted entirely to basic monitor control cards.

Basic Monitor Control Card Format

The following item illustrates the format of any basic monitor control card.

GENERAL FORMAT OF IBSYS CONTROL CARDS

<u>Column</u>	<u>Contents</u>
1	\$
2 - 15	Control card name, usually a verb, left- justified
16 -72	Variable field information (arg1, arg 2, , argn)

Note that column 1 is always a \$ associating the card with basic monitor. Columns 2 through 15 contain the name of the card which in turn governs its function. The remainder of each card through column 72 supplies additional information as dictated by the type of card.

Control Card Categories

Basic monitor control cards can be grouped into four categories:

1. Operation cards are those that perform functions governing the overall operations of IBSYS while the supervisor (IBSUP) of basic monitor is in control. An example of this type is the \$EXECUTE card which specifies the programming system to be executed.
2. Unit assignment cards are used to attach or detach specific units as well as assigning them or releasing them from a specific function.
3. Tape manipulation cards allow the manipulation of any of the system function units; rewinding, writing end of file, etc.
4. Miscellaneous cards perform a variety of functions such as supplying the current date for systems use.

A discussion of each of the control cards by category follows:

Operational Control Cards

There are seven operational control cards as described below:

1. The \$EXECUTE card is quite familiar at this point. Recognition of this card causes IBSUP to locate the desired programming system on the library, load its individual monitor and relinquish control to it.
2. The \$IBSYS card though considered a basic monitor control card is one of the two which is interpreted by the programming systems. When recognized, a system must return control to the basic monitor via cell SYSRET of the one word entries table.
3. The \$PAUSE card causes a machine halt to allow operator intervention, as this card is interpreted by basic monitor, the halt would occur while the monitor is in control. In addition to the halt a message is printed on-line as follows:

OPERATOR ACTION PAUSE - PRESS START TO CONTINUE

Instructions to the operator can be inserted on the card and will be printed along with this message.

4. The \$CARDS card informs basic monitor that all of its following control cards will be read from SYSCRD (system card reader) until it encounters a \$TAPE card (Refer to 4.4.03 #5). Note that the programming system control cards will still be read from whatever device is assigned to the SYSINI function. A possibility arises here which has not been pointed out before. This possibility is that of having the basic monitor control cards in the card reader and the programming system control cards on tape and vice versa.

5. The \$TAPE card informs basic monitor that all of its following control cards will be read from the device assigned to the SYSINI function until it encounters a \$CARDS card (Refer to 4.4.03-#4).
6. The \$RESTORE card performs a majority of the functions that occur during a cold start. The supervisor is loaded; the nucleus is relocated from its load location except for cell SYSDAT which is not altered; and the unit control blocks are generated with the exception of word three of each, which reflects current tape positions.

In addition, the effect of all previous control cards is cancelled except for that of the \$CARDS or \$TAPE card.

7. The \$STOP card is normally the last card to appear in a control card deck. It is an indication of the end of an IBSYS Processor run.

Upon interpretation of this card a message is printed on line as follows:

END OF JOBS --- CANNOT PROCEED

After the processing of \$STOP card, a cold start must be executed to resume operations.

Unit Assignment Control Cards

There are five unit assignment control cards as described below:

1. The \$ATTACH card places the physical unit specified on the card into the availability chain for the associated channel and marks the unit attached by setting the attachment flag in the associated unit control block. The model of the unit attached can also be specified in the card and the model flag of the UCB will be set accordingly.
2. The \$DETACH card removes the physical unit specified from the appropriate availability chain and marks it detached by resetting the attachment flag of the associated unit control block. If the unit detached is a function unit its reference will also be removed from the system units table.
3. The \$AS card must follow a \$ATTACH card. When interpreted by IBSUP, the unit specified on the preceding \$ATTACH card is assigned to the system unit function specified on the \$AS card. As a result the appropriate entry will be made into the system units table. The density specification for the function unit will also be set as specified in the \$AS control card.

4. The \$RELEASE card will cause the reference of the unit associated with the system unit function specified in the card to be removed from the system units table. Since the unit is then available, unless assigned to another function, it is entered into the appropriate unit availability chain and flagged attached in its UCB.
5. The \$SWITCH card interchanges the unit assignment of the two system unit functions specified in the card. As a result of the interpretation of this card, the system units table references of the two units will be transposed.

Tape Manipulation Control Cards

The three tape manipulation control cards permit the machine operator to perform certain non-data operations on the function units. If no unit is assigned to the specified system unit function, the operation is ignored.

1. The \$ENDFILE card will cause an end of file mark to be written on the specified system function unit. No test is made for legality of operation on the unit, such as an EOF being written on SYSLB1.
2. The \$REWIND card causes the unit assigned to the specified SYSUNI function to be rewound.
3. The \$REMOVE card causes the unit assigned to the specified SYSUNI function to be rewound and unloaded.

Miscellaneous Control Cards

There are seven control cards in this group. The following list will serve to explain their operation.

1. The \$DATE card causes the six characters beginning in column 16 to be stored in cell SYSDAT of the one word entry table. The form should be:

Column	1	16
	\$DATE	MMDDYY
MM - month	01	12
DD - day	01	31
YY - year	63	99
2. The \$* card serves as a true comment card causing the text beginning in column 3 to be printed on line.
3. The \$UNITS card causes all SYSUNI names, units assigned, and assigned densities to be printed on line. In addition it lists all attached units not assigned or reserved and last it lists all inter-system reserve units.

4. The \$UNLIST card causes the printing of all IBSYS control cards except \$PAUSE and \$STOP to be suspended.
5. The \$LIST card causes printing of IBSYS control cards to be resumed after having been suspended by a \$UNLIST card.
6. The \$IBEDT card causes the supervisor to call the system editor from the library tape. Control is then relinquished to the editor which will then perform the edit functions as specified by the editor control cards.

Return to the supervisor of basic monitor will be accomplished when the system editor recognizes a \$IBSYS card.

7. The \$ID card causes a transfer to SYSIDR in the one word entry table. If an installation accounting routine is present, SYSIDR will contain a transfer to it in place of a TTR 2, 4.

THE INPUT-OUTPUT EXECUTOR (IOEX)

Functions of IOEX

The functions can be separated into three groups. Each group will have a number of sub-routines to accomplish its functions. The groups are broken down as follows:

The I/O Operation Scheduler - These subroutines will initiate or give priority to the movement of data. They will also perform all necessary non-data select operations.

The Channel Trap Supervisor - These routines will process all data channel traps and maintain activity on any channel as long as the user desires. Redundancy recovery procedures are also handled by this group.

Miscellaneous. - A group of routines are available to convert data from one form to another, print or punch data on-line, and provide temporary or permanent program halts.

Advantages Offered To The User

Some of the advantages to be found in using IOEX are:

1. All I/O activity is handled through a central point despite great differences in buffering techniques of the users.
2. The diagnosis of I/O failures is facilitated since all unit usage is controlled through one package.
3. The debugged routines within IOEX help to minimize the amount of I/O programming needed by the user.

4. Information about the current positions of all I/O devices is maintained by IOEX and is readily available to the user.
5. Standard and automatic redundancy recovery is performed when necessary.

The Use of IOEX By a Program

To help explain IOEX, we're going to use a hypothetical program which will contain all the necessary instructions to perform an I/O operation in conjunction with IOEX. Keep in mind that the term user can mean one of the participating programming systems within the IBSYS processor as well as a program being written in FAP or MAP language for execution under monitor control.

The Request Queue

The first step in beginning an I/O operation is the establishment of the request queue. To accomplish this the user will make use of the unit control block belonging to the device that he wishes to perform the I/O operation. The prefix of UCB word 2 must be set with the select type (read or write) and redundancy message control flags. The decrement is set to the address of the user's select routine. The use of the address portion of UCB word 2 is optional as IOEX does not interrogate it. It may be used for communication between the different subroutines of a user's program.

Initiating the I/O Operation

The main routines involved in starting an operation are illustrated in Figure 81. The routines from our hypothetical program are marked HYPO while all actual IOEX routines are labeled as such. Let's examine the blocks one at a time.

Main Program. The need for an I/O operation is discovered here. Decisions about the I/O device, make and type of select, and even the I/O sequences are generally determined here. This information and program control is then passed via a TSX and an associated calling sequence to a general read/write routine.

IOXR W. The general read/write routine prepares the request queue (UCB word 2) and sets up the proper calling sequence to enter IOEX. Decisions about channel activity and the establishment of holding loops (waiting for data) may also be made here.

ACTIV. Channel activity is checked and if the channel is dormant ACTIV will set up a few cells, establish a proper calling sequence, and transfer to the user's select routine (IOXSEL). If the channel is active, one of two courses will be followed; depending on the type of sequence used to enter ACTIV. The operation will be considered scheduled and a return made to IOXRW in one case. In the other, a wait loop will be established within

ACTIV and the unit will be given top priority on the channel.

IOXSEL(+). The actual read or write select is performed here followed by the reset and load channel. The I/O command sequence initiated here must be designed to result in a command word trap. A return is made from here to ACTIV, and back up the line to the main program which now proceeds.

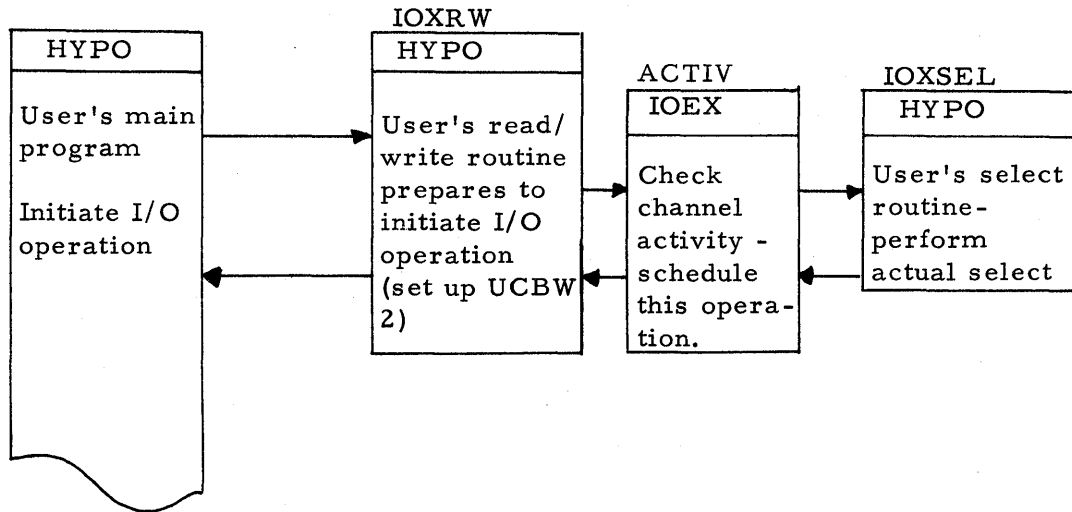


Figure 81

Completion of the I/O Operation

At the end of the I/O operation a command word trap results. Figure 28 assumes that we are processing in the main program when the CWT occurs. Let's examine the blocks again, in sequence, from the time the trap takes place.

XTRAP. The channel trap cells are initialized by basic monitor to result in transfer of control to the XTRAP routine. The normal saving procedures and redundancy checking takes place followed by a transfer to the user's select routine.

IOXSEL(-). The select routine must be designated to distinguish between an entry from ACTIV and one from XTRAP. It is able to accomplish this by testing the accumulator sign position. The sign is plus on entry from ACTIV and minus on entry from XTRAP. The minus entry (sometimes referred to

as posting entry) will perform some wrap-up operations and determine whether another I/O operation is desired on this device. To post the operation as complete the routine has only to clear UCB Word 2. To obtain another I/O operation on the same device IOXSEL - has only to leave UCB Word 2 untouched. In either case, the return is to XTRAP.

BGI. This routine is entered directly from XTRAP after a return from IOXSEL -. Its function is to scan the UCB of the channel that the trap has occurred on and determine if there is another request queue waiting. If a request is waiting, it will initiate the I/O operation by entering the IOXSEL+ routine. If no request is present, it will restore all the registers and return to the main program at the point where the CWT occurred.

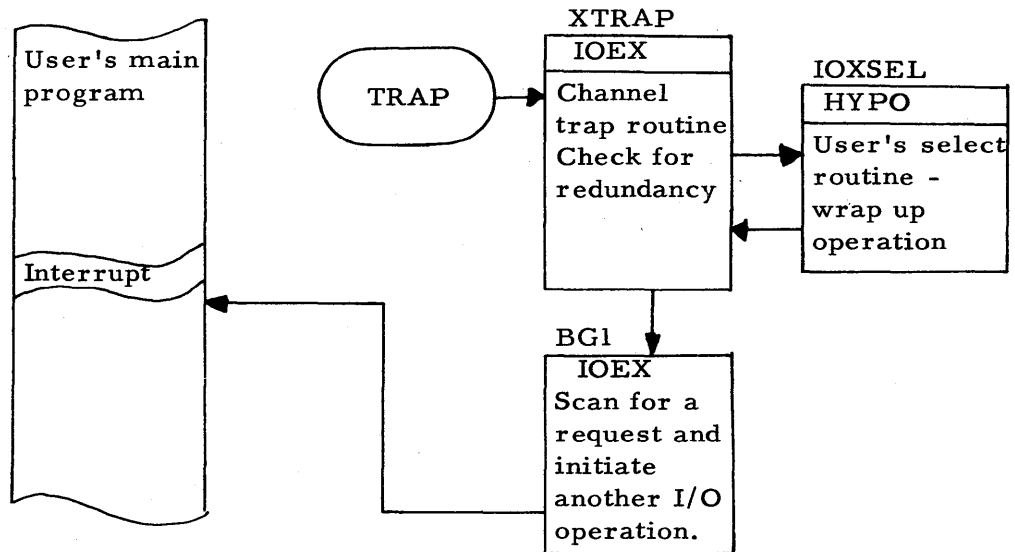


Figure 82.

A Detailed Investigation of IOEX Routines

Now that we've looked at the over-all logic of a data select operation involving IOEX, let's make a more thorough investigation of what these routines do. In addition, we'll determine what types of calling sequences are necessary for entry to them and what they provide on exit.

Information Provided By IOEX

Upon either entry to the user's select routine, IOEX disables channel trapping and provides :

- C(XR1) - The 2's complement of the channel index where 0 = A, 1 = B, 2 = C, etc. As an example, we would expect XR1 to contain

7777 octal (-1) on entry to a select routine for channel B.

S(AC) - Sign of the accumulator is set plus (SEL+) for initiating I/O operation and minus (SEL-) for the posting entry.

A(AC) - Location of the UCB.

During the posting entry (SEL-) IOEX provides in addition to the above:

Sense Indicators - - - 7607 Channel

Bit S -	Noise record flag
1 -	The record was <u>not</u> an apparent noise record
0 -	The record was a noise record.
Bit 1 -	EOF (Read) or EOT (Write)
0 -	<u>No</u> EOF or EOT
1 -	<u>EOF</u> or EOT
Bit 2 -	Permanent Redundancy (Read Only)
0 -	No permanent redundancy
1 -	Permanent redundancy

Bit S is on in all cases except when a noise record is detected.

The users select routine must not destroy XR1. XR 2 and the sense indicators need not be saved.

The calling sequence from IOEX to the user select routine is of the following form.

TSX	IOXSEL, 4
Return 1	Normal Return from Sel + or Sel-
Return 2	Return used at Sel-because of error
Return 3	Return used at Sel-because of error

The error return conditions will be discussed later.

Now that we've got an idea of what IOEX will provide us with, let's see what the routines do.

ACTIV

To enter the ACTIV subroutine, the user must provide the following:

TSX	ACTIV, 4
P	A, T
RETURN	

A, T gives the location of a cell which contains in its address, the location of the UCB upon which I/O activity is desired. Let's take an example and see how it works. Assume that ACTIV wishes to obtain the

address of the UCB so that it can specialize some instructions and that the I/O device involved is SYSUT1. We can use the SYSUNI table and the sequence:

```
TSX      ACTIV, 4
P        SYSUT1
```

If ACTIV uses:

```
CLA*    1, 4
```

you can see that the address of the UCB for SYSUT1 will be placed in the accumulator.

There are two interpretations for P. The first is P = PZE. If the channel is dormant upon entry, the proper channel activity cell (CHXAC - one or each channel) is set with the location of the UCB and the users select routine (IOXSEL+) is entered. Upon return from the select routine, the index registers and sense indicators are restored, traps are enabled, and a return is made to the user's program. If the channel is active upon entry, an immediate return is made to the user's program. This is all that is necessary as the unit will eventually receive attention through the functioning of the BGI routine.

If P = MZE this indicates that the user wishes this unit to obtain priority on the channel (i. e. the next select). If the channel is dormant, there is no difference between the operation of PZE or MZE. The channel activity cell (CHXAC) is set and entry is made to the select routine. However, when entry occurs while the channel is active, the address of the UCB is put in the channel priority cell (CHXSP - one for each channel) and a wait loop is established within ACTIV. The wait loop will hold until a trap occurs and the BGI routine interrogates the channel priority cell (CHXSP). BGI will cause the unit specified in CHXSP to obtain the next select by entering the SELECT + routine of the user. BGI will set the CHXAC equal to the CHXSP (Loc of UCB) and return to ACTIV after the select has been given. At this point ACTIV will find that the unit it had put in CHXSP is now the same as CHXAC and it will be free to make a return to the user's program.

Save - XTRAP

Once the trap occurs, the SAVE portion of the trap supervisor saves all machine registers and machine status conditions so that they may be restored upon exit. XTRAP will look for I/O checks, EOF or EOT conditions and will use a group of subroutines labeled REDXX (the XX designation are numbers in the program listing) to perform the redundancy checking and recovery operations. The redundancy recovery procedures are as follows:

Redundancy Recovery on Read. Tape check trap is never used for read as a full count of words is desired to determine if a noise record has been read.

Any record of less than three words (1 or 2) is considered a noise record. In the event of a noise record, no redundancy recovery action is taken and the noise flag will be set at entry to SEL - (Sense Indicator S=0).

The total number of recovery tries on a read redundancy is controlled by an assembly parameter (RDUNRT - Released as 100). Every tenth time during redundancy recovery a tape cleaner action is taken. This involves backspacing an extra two records and skipping forward two before actually re-reading. This will normally allow the bad area to pass over the tape cleaner blade. The user's SEL+ routine is used for the actual retry. If a redundancy becomes permanent (Number of re-tries = RDUNRT), the permanent redundancy flag (Sense Indicator Bit 2 = 1) will be set on entry to SEL- and a message will be printed providing it has not been suppressed through the use of UCB Word 2, Bit 1.

At this point, let's see what the returns from SEL- will do in the event of errors.

The three exits are:

- RETURN 1 - The record is accepted
- RETURN 2 - The user considers the record noise and it is discarded. An operator message is printed and the noise record flag is set on in UCB word 3. SEL+ is re-entered to read the next record.
- RETURN 3 - Redundancy recovery action is re-started and IOEX will go through the entire procedure again.

Redundancy Recovery on Write. Tape check trap is used for redundancy checking on write. If a TCT occurs, XTRAP will use REDXX to backspace and erase. If a redundancy occurs during the erase, a message will be printed on line. The record will be rewritten by entering SEL+. This procedure will repeat until the record is either written correctly or EOT condition is reached. After each group of 25 erase areas, an operator message is written.

After each successful write, IOEX checks the apparent record length and if it finds that less than 3 words have been written, it will enter SEL- with the noise record indication in the sense indicators. The returns from SEL-after a write are as follows:

- RETURN 1 - The Record is accepted.
- RETURN 2 - The Record is accepted and an operator message is printed indicating that short record has been written.

The last function performed by XTRAP prior to entering SEL- is the updating of the record count in UCB Word 3.

BG1

Upon normal return from SEL-, XTRAP will transfer to BG1. This routine will check for another request queue on the channel in a priority sequence. First the channel priority cell is tested. If it contains a UCB location and the UCB Word 2 has a select address, this device will receive the next select. If the channel priority cell is not set, then a test is made to determine if another I/O operation should be started on the same unit that just completed the operation. If none is required on this unit, each UCB following this one is checked to determine if a request is waiting for that unit. At the end of the block of UCB's a search is re-initialized from the beginning of the block. In this manner any UCB's prior to the one that has just completed will also be checked. This also insures that the units will all be serviced. If no UCB's have a request queue waiting, the channel is allowed to become dormant and will require an entry to ACTIV to start its operation again. The first UCB found with an address of a select routine in UCB Word 2 will have the SEL+ entry made for it before leaving the trap supervisor.

Before returning to the point interrupted by the data channel trap, the machine registers and status conditions previously saved are restored.

The Sample Program

Now that we've looked at some of the details of the routines, let's flow chart a sample program that uses IOEX.

The make-up of the main program is unimportant at this time, so we'll pick up the operation at the point where we enter the general read/write routine. Just preceding the program listing there is an explanation of the types of calling sequences needed and so that everyone will be thinking along the same line let's assume that we're entering with the following:

```
TSX      IOXRW, 4
PZE      LOCFIL, 1, RCHSEQ
PZE      EOF, , ERR
```

Assume that the symbols used above are defined in the main program. The only remaining symbol that should need explanation is (RCHXI used in the IOXSEL routine. (RCHXI, defined in the SST, is a cell within IOEX of the form:

```
RCHXI PZE      RCHX, 1
```

RCHX is a table of reset and load channel instructions of this form:

```
RCHX  RCHA      **
      RCHB      **
      RCHC      **
      RCHD      **
      RCHE      **
      RCHF      **
      Etc.
```

If you recall that XR1 will contain the 2's complement of the channel index, (0=A, 1=B, etc.) the use of this table should be apparent. There are several tables within IOEX that are constructed in the same manner.

Here are a few questions that you should find the answers to as you flow chart the sample routine.

1. How is the address of the I/O command sequence passed to the SELECT routine?
2. Where does the program wait while the data is being read in?
3. How is the wait loop changed to allow the program to proceed?

* Sample Routine Perform Read or Write From Tape Using IOEX

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

*

Calling Sequence

TSX IOXRW, 4

P LOCFIL, M, RCHSEQ

PZE EOF, , ERR

Where

P If Minus, Write If Plus, Read

Bit 1 If 1, No Message If 0, Message

LOCFILE Location of a Word with UCB in Address (Indir.Ref.)

M If BCD, 1 If Binary 0

RCHSEQ Location of I/O Commands (Ending in Trap)

EOF End of File or End of Tape Exit

ERR Error Exit

Indicators and ACC are destroyed. IRS are saved.

IOXRW

SXA IOXS4, 4

Save IR4

CLA 2, 4

STA IOXEF

Set EOF Exit

ARS 18

STA IOXER

Set Permanent Redundancy Exit

CAL 1, 4

Get First Word of Call Sequence

STT IOXSLL

Save Mode for Select

STP IOXSLL

Save Prefix for Select Type

STA IOXND

Put Loc In Active Calling Sequence

STA *+1

Set To Pick Up UCB

LAC **, 4

-L (UCB)

ARS 18

I/O Commands Loc to Address

STA IOXSLL

Put in Select Word

CLA IOXSLL

Get Select Word

ZET 1, 4

Test for Other Use of This Unit

TRA *-1

Wait Till Unit Free

STO 1, 4

Signed Control Word to UCB Word 2

STZ IOXIN

Set In Operation Word On.

	TSX	ACTIV, 4	Go To Activate
IOXND	PZE	**	Unit
	NZT	IOXIN	Test For Request Complete
	TRA	*-1	Not Done, Wait
	LDI	IOXIN	Pick Up Completion Bits Set By IOXSEL-
IOXS4	AXT	**, 4	Set For Exit
	LFT	200000	Test EOF, EOT
IOXEF	TRA	**	EOF Exit
	LFT	100000	Test for Perm. Redundancy
IOXER	TRA	**	Yes, Error Exit
	TRA	3, 4	Normal Return
*			
IOXSLL	PZE	0, , IOXSEL	Location of Select Routine
IOXIN	PZE	**	In Operation Cell
IOXMD	PZE	**	Mode Switch
*	IOXSEL	-- Routine Entered Twice By IOEX for Each I/O Operation	
*			
*			
IOXSEL	SXA	IOXSS4, 4	Save IR 4
	PAC	0, 4	-L(UCB)
	TMI	IOXPST	Select Minus or Posting Entry
	CLA	0, 4	UCB Word 1
	PDX	0, 2	Unit to IR 2
	CLA	1, 4	UCB Word 2
	STA*	RCHXI	Store Loc to RCHX
	STT	IOXMD	Save Mode Flag
	NZT	IOXMD	Test Mode
	TXI	*+1, 2, 16	Set Binary Mode for Unit
	TMI	IOXWR	Write
	SXA	*+1, 2	Place Read Select Address
	RDS	**	Read Select
	XEC*	RCHXI	Issue Reset Load Channel
IOXSS4	AXT	**, 4	Restore IR 4
	TRA	1, 4	And Exit IOXSEL
IOXWR	SXA	*+1, 2	Place Select Address for Write
	WSR	**	Write Select
	TRA	IOXSS4-1	Go To Issue Channel Commands
IOXPST	STI	IOXIN	Posting Entry Save Error Flags
	STL	IOXIN	Set In-Operation Word Off
	STZ	1, 4	Set UCB Word 2 Zero
	TRA	IOXSS4	Go To Exit

APPENDIX

INSTRUCTION LIST BY TYPE

INDEX TRANSMISSION*****		
AXI	+0774	ADDRESS TO INDEX TRUE 40
LAC	+0535	LOAD COMP.OF ADDRESS IN XR 39
LXA	+0534	LOAD INDEX FROM ADDRESS 39
PAC	+0737	PLACE COMP.OF ADDRESS IN XR 40
PAX	+0734	PLACE ADDRESS IN INDEX 40
PXA	+0754	PLACE INDEX IN ADDRESS 40
SXA	+0634	STORE INDEX IN ADDRESS 40
UNCONDITIONAL TRANSFER*****		
HTR	+0000	HALT AND TRANSFER 49
NOP	+0761	NO OPERATION 49
TRA	+0020	TRANSFER 49
ARITHMETIC RESULT TESTING*****		
LBT	+0760	LOW ORDER BIT TEST 49A
PBT	-0760	P BIT TEST 49A
TMI	-0120	TRANSFER ON MINUS 49A
TNZ	-0100	TRANSFER ON NO AC ZERO 49A
TPL	+0120	TRANSFER ON PLUS 49
TZE	+0100	TRANSFER ON ZERO 49
STORAGE TESTING*****		
CAS	+0340	COMPAKE AC WITH STORAGE 50
LAS	-0340	LOGICAL COMPARE AC/STORAGE 50
INDEX TESTING*****		
TIX	+2000	TRANSFER ON INDEX REGISTER 51
TSX	+0074	TRANSFER AND SET INDEX 50
TXI	+1000	TRANSFER WITH XR INCREMENTED 51
FIXED POINT ARITHMETIC*****		
ACL	+0361	ADD AND CARRY LOGICAL WORD 27
ADD	+0400	ADD 25
CHS	+0760	CHANGE ACCUMULATOR SIGN 26
DVP	+0221	DIVIDE OR PROCEED 26
MPY	+0200	MULTIPLY 26
SUB	+0402	SUBTRACT 26
REGISTER SHIFTING*****		
ALS	+0767	ACCUMULATOR LEFT SHIFT 29
ARS	+0771	ACCUMULATOR RIGHT SHIFT 29
LGL	-0763	LOGICAL LEFT SHIFT 29
LGR	-0765	LOGICAL RIGHT SHIFT 30
RQL	-0773	ROTATE MQ LEFT 30
WORD TRANSMISSION*****		
CAL	-0500	CLEAR AND ADD LOGICAL WORD 35
CLA	+0500	CLEAR AND ADD 35
LDQ	+0560	LOAD MQ REGISTER 34
SLW	+0602	STORE LOGICAL WORD 35
STA	+0621	STORE ADDRESS 35
STO	+0601	STORE ACCUMULATOR 35
STQ	-0600	STORE MQ REGISTER 34
STZ	+0600	STORE ZEROS 35
XCA	+0131	EXCHANGE AC AND MQ REGISTERS 35A
XCL	-0130	EXCHANGE LOGICAL AC AND MQ 35A
ZAC		ZEROS TO ACCUMULATOR 35A
LOGICAL OPERATIONS*****		
ANA	-0320	AND TO ACCUMULATOR 31
ANS	+0320	AND TO STORAGE 31
ERA	+0322	EXCLUSIVE OR TO ACCUMULATOR 32
ORA	-0501	OR TO ACCUMULATOR 32
ORS	-0602	OR TO STORAGE 32

Op. Code	Flow Notation
ACL +0361	(Y) + (AC) P, 1-35 → (AC) P, 1-35
ADD +0400	(Y) + (AC) S, 1-35 → (AC) S, 1-35
ALS +0767	(AC) Q, P, 1-35 shifted left n places
ANA -0320	(Y) ANDed with (AC) P, 1-35 → (AC) P, 1-35
ANS +0320	(Y) ANDed with (AC) P, 1-35 → (Y)
ARS +0771	(AC) Q, P, 1-35 shifted right n places
AXT +0774	(AXT) 21-35 → index register
CAL -0500	(Y) → (AC) P, 1-35. Zeros → (AC) S, Q
CAS +0340	If (AC) > (Y); IC+1 → IC. If (AC) = (Y); IC+2 → IC. If (AC) < (Y); IC+3 → IC.
CHS +0760	Invert (AC) S
CLA +0500	(Y) → (AC) S, 1-35. Zeros → (AC) Q, P
DVP +0221	(AC) and (MQ) ÷ (Y). Quotient → (MQ), remainder → (AC)
ERA +0322	(Y) S, 1-35 ORed with (AC) P, 1-35. If position not =, 1 → (AC). If =, 0 → (AC)
HTR +0000	Stops computer. (HPR) 21-35 → IC with start key depression
LAC +0535	2's complement (Y) 21-35 → index register
LAS -0340	If (AC) > (Y); IC+1 → IC. If =, IC+2 → IC. If (AC) < (Y), IC+3 → IC.
LBT +0760	If (AC) 35 = 1, IC+1 → IC. If (AC) 35 = 0, IC → IC
LDQ +0560	(Y) → (MQ)
LGL -0763	(AC) Q, P, 1-35 and (MQ) shifted left n places
LGR -0765	(AC) Q, P, 1-35 and (MQ) shifted right n places
LXA +0534	(Y) 21-35 → index register
MPY +0200	(Y) x (MQ). Product → (AC) and (MQ)
NOP +0761	Dummy instruction.
ORA -0501	(Y) ORed with (AC). If position is a 1, 1 → (AC). If both positions 0, 0 → (AC)
ORS -0602	(Y) ORed with (AC). If position is a 1, 1 → (Y). If both positions 0, 0 → (Y)
PAC +0737	2's complement (AC) 21-35 → index register
PAX +0734	(AC) 21-35 → index register
PBT -0760	If (AC) P = 1, IC + 1 → IC. If (AC) P = 0, IC → IC
PXA +0754	Index register → (AC) 21-35. (AC) S, Q, P, 1-20 set to zeros
RQL -0773	(MQ) shifted left n places. (MQ) S → (MQ) 35
SLW +0602	(AC) P, 1-35 → (Y)
STA +0621	(AC) 21-35 → (Y) 21-35
STO +0601	(AC) S, 1-35 → (Y)
STQ -0600	(MQ) → (Y)
STZ +0600	Zeros → (Y)
SUB +0402	(AC) - (Y). Result to (AC)
SXA +0634	Index register → (Y) 21-35
TIX +2000	If (XR) > (V); (XR) - (V) → (XR) and (TIX) 21-35 → IC
TMI -0120	If (AC) S = 1, (TMI) 21-35 → IC
TNZ -0100	If (AC) Q, P, 1-35 = 0, (TNZ) 21-35 → IC
TPL +0120	If (AC) S = 0, (TPL) 21-35 → IC
TRA +0020	(TRA) 21-35 → IC
TSX +0074	2's complement of TSX location → index register. (TSX) 21-35 → IC
TXI +1000	(V) added to (XR) → (XR). (TXI) 21-35 → IC
TZE +0100	If (AC) Q, P, 1-35 = 0, (TZE) 21-35 → IC
XCA +0131	(AC) S, 1-35 → (MQ). (MQ) → (AC) S, 1-35. Zeros → (AC) Q, P
XCL -0130	(AC) P, 1-35 → (MQ). (MQ) → (AC) P, 1-35. Zeros → (AC) S, Q
ZAC	Zeros → (AC) S, Q, P, 1-35

APPENDIX

Instruction List By Type

INDEX TRANSMISSION**		
AXC	-0774	ADDRESS TO XR COMPLEMENT 40
AXT	00774	ADDRESS TO INDEX TRUE 40
LAC	00535	LOAD COMP.OF ADDRESS IN XR 39
LDC	-0535	LOAD COMP.OF DECREMENT IN XR 41
LXA	00534	LOAD INDEX FROM ADDRESS 39
LXD	-0534	LOAD XR FROM DECREMENT 40
PAC	00737	PLACE COMP.OF ADDRESS IN XR 40
PAX	00734	PLACE ADDRESS IN INDEX 40
PDX	-0734	PLACE DECREMENT IN INDEX 40
PXA	00754	PLACE INDEX IN ADDRESS 40
PXD	-0754	PLACE INDEX IN DECREMENT 40
SXA	00634	STORE INDEX IN ADDRESS 40
SXD	-0634	STORE INDEX IN DECREMENT 41
SENSE INDICATOR**		
LDI	00441	LOAD INDICATORS 47
OSI	00442	OR STORAGE TO INDICATORS 48
PIA	-0046	PLACE INDICATORS IN AC 48
STI	00604	STORE INDICATORS 48
SIL	-0055	SET INDICATORS OF LEFT HALF 48
SIR	00055	SET INDICATORS OF RIGHT HALF 48
UNCONDITIONAL TRANSFER**		
HTR	00000	HALT AND TRANSFER 49
NOP	00761	NO OPERATION 49
TRA	00020	TRANSFER 49
XEC	00522	EXECUTE 49
ARITHMETIC RESULT TESTING**		
LBT	00760	LOW ORDER BIT TEST 49
PBT	-0760	P BIT TEST 49
TMI	-0120	TRANSFER ON MINUS 49A
TNC	-0140	TRANSFER ON NO AC OVERFLOW 49A
TNZ	-0100	TRANSFER ON NO AC ZERO 49A
TOV	00140	TRANSFER ON AC OVERFLOW 49A
TPL	00120	TRANSFER ON PLUS 49
TQP	00162	TRANSFER ON MQ PLUS 49A
TZE	00100	TRANSFER ON ZERO 49
STORAGE TESTING**		
CAS	00340	COMPARE AC WITH STORAGE 50
LAS	-0340	LOGICAL COMPARE AC/STORAGE 50
NZT	-0520	STORAGE NON-ZERO TEST 50
ZET	00520	STORE ZERO TEST 50
INDEX TESTING**		
TIX	02000	TRANSFER ON INDEX REGISTER 51
TNX	-2000	TRANSFER ON NO INDEX 51
TSX	00074	TRANSFER AND SET INDEX 50
TXH	03000	TRANSFER ON INDEX HIGH 51
TXI	01000	TRANSFER WITH XR INCREMENTED 51
TXL	-3000	TRANSFER ON XR LOW OR EQUAL 51
FIXED POINT ARITHMETIC**		
ACL	00361	ADD AND CARRY LOGICAL WORD 27
ADD	00400	ADD 25
CHS	00760	CHANGE ACCUMULATOR SIGN 26
CLS	00502	CLEAR AND SUBTRACT 26
DVP	00221	DIVIDE OR PROCEED 26
MPY	00200	MULTIPLY 26
SUB	00402	SUBTRACT 26

Instruction List By Type

FLOATING POINT ARITHMETIC**		
FAD	&0300	FLOATING ADD 28
FDP	&0241	FLOATING DIVIDE OR PROCEED 28
FMP	&0260	FLOATING MULTIPLY 28
FSB	&0302	FLOATING SUBTRACT 28
REGISTER SHIFTING**		
ALS	&0767	ACCUMULATOR LEFT SHIFT 29
ARS	&0771	ACCUMULATOR RIGHT SHIFT 29
LGL	-0763	LOGICAL LEFT SHIFT 29
LGR	-0765	LOGICAL RIGHT SHIFT 30
RQL	-0773	ROTATE MQ LEFT 30
WORD TRANSMISSION**		
CAL	-0500	CLEAR AND ADD LOGICAL WORD 35
CLA	&0500	CLEAR AND ADD 35
LDQ	&0560	LOAD MQ REGISTER 34
SLW	&0602	STORE LOGICAL WORD 35
STA	&0621	STORE ADDRESS 35
STD	&0622	STORE DECREMENT 35
STL	-0625	STORE INST. LOC. COUNTER 35
STO	&0601	STORE ACCUMULATOR 35
STP	&0630	STORE PREFIX 35
STQ	-0600	STORE MQ REGISTER 34
STZ	&0600	STORE ZEROS 35
XCA	&0131	EXCHANGE AC AND MQ REGISTERS 35A
XCL	-0130	EXCHANGE LOGICAL AC AND MQ 35A
ZAC		ZEROS TO ACCUMULATOR 35A
LOGICAL**		
ANA	-0320	AND TO ACCUMULATOR 31
ANS	&0320	AND TO STORAGE 31
ERA	&0322	EXCLUSIVE OR TO ACCUMULATOR 32
ORA	-0501	OR TO ACCUMULATOR 32
ORS	-0602	OR TO STORAGE 32
SENSE LIGHT AND SENSE INDICATOR CONTROL**		
SLT	-0760	SENSE LIGHT TEST 52
SLN	&0760	SENSE LIGHT ON 52
TIF	&0046	TRANSFER IF INDICATORS OFF 52
TIO	&0042	TRANSFER IF INDICATORS ON 52
TRAPPING CONTROL**		
ENB	&0564	ENABLE FROM Y 54
ETM	&0760	ENTER TRAPPING MODE 53
LTM	-0760	LEAVE TRAPPING MODE 53
RCT	&0760	RESTORE CHANNEL TRAPS 53
TTR	&0021	TRAP TRANSFER 53
INPUT/OUTPUT CONTROL**		
LCH		LOAD CHANNEL 69
RDS	&0762	READ SELECT 70
RCH		RESET AND LOAD CHANNEL 69
SCH		STORE CHANNEL 68
TCO		TRANSFER ON CHAN IN OPERATION 75
TEF		TRANSFER ON CHAN END OF FILE 75
TRC		TRANSFER ON CHAN REDUN CHECK 75
WRS	&0766	WRITE SELECT 70
INPUT/OUTPUT COMMANDS**		
IOCD	0	I/O WITH COUNT AND DISCONNECT 71
IOCP	4	I/O WITH COUNT AND PROCEED 72
IOCT	5	I/O WITH COUNT AND TRANSFER 72
IORP	2	I/O OF A RECORD AND PROCEED 72
IORT	3	I/O OF A RECORD AND TRANSFER 73
IOSP	6	I/O UNTIL SIGNAL THEN PROCEED 73
IOST	7	I/O UNTIL SIGNAL THEN TRANSFER 73
TCH	1	TRANSFER IN CHANNEL 74

NUMERIC INSTRUCTION LIST

HTR	60000	HALT AND TRANSFER	49
TRA	60020	TRANSFER	49
TTR	60021	TRAP TRANSFER	53
TIO	60042	TRANSFER IF INDICATORS ON	52
TIF	60046	TRANSFER IF INDICATORS OFF	52
PIA	-0046	PLACE INDICATORS IN AC	48
SIR	60055	SET INDICATORS OF RIGHT HALF	48
SIL	-0055	SET INDICATORS OF LEFT HALF	48
TSX	60074	TRANSFER AND SET INDEX	50
TZE	60100	TRANSFER ON ZERO	49
TNZ	-0100	TRANSFER ON NO AC ZERO	49A
TPL	60120	TRANSFER ON PLUS	49
TMI	-0120	TRANSFER ON MINUS	49A
XCL	-0130	EXCHANGE LOGICAL AC AND MQ	35A
XCA	60131	EXCHANGE AC AND MQ REGISTERS	35A
TOV	60140	TRANSFER ON AC OVERFLOW	49A
TNO	-0140	TRANSFER ON NO AC OVERFLOW	49A
TQP	60162	TRANSFER ON MQ PLUS	49A
MPY	60200	MULTIPLY	26
DVP	60221	DIVIDE OR PROCEED	26
FDP	60241	FLOATING DIVIDE OR PROCEED	28
FMP	60260	FLOATING MULTIPLY	28
FAD	60300	FLOATING ADD	28
FSB	60302	FLOATING SUBTRACT	28
ANS	60320	AND TO STORAGE	31
ANA	-0320	AND TO ACCUMULATOR	31
ERA	60322	EXCLUSIVE OR TO ACCUMULATOR	32
CAS	60340	COMPARE AC WITH STORAGE	50
LAS	-0340	LOGICAL COMPARE AC/STORAGE	50
ACL	60361	ADD AND CARRY LOGICAL WORD	27
ADD	60400	ADD	25
SUB	60402	SUBTRACT	26
LDI	60441	LOAD INDICATORS	47
OSI	60442	OR STORAGE TO INDICATORS	48
CLA	60500	CLEAR AND ADD	35
CAL	-0500	CLEAR AND ADD LOGICAL WORD	35
ORA	-0501	OR TO ACCUMULATOR	32
CLS	60502	CLEAR AND SUBTRACT	26
ZET	60520	STORE ZERO TEST	50
NZT	-0520	STORAGE NON-ZERO TEST	50
XEC	60522	EXECUTE	49
LXA	60534	LOAD INDEX FROM ADDRESS	39
LXD	-0534	LOAD XR FROM DECREMENT	40
LAC	60535	LOAD COMP.OF ADDRESS IN XR	39
LDC	-0535	LOAD COMP.OF DECREMENT IN XR	41
LDQ	60560	LOAD MQ REGISTER	34
ENB	60564	ENABLE FROM Y	54
STZ	60600	STORE ZEROS	35
STQ	-0600	STORE MQ REGISTER	34
STO	60601	STORE ACCUMULATOR	35
SLW	60602	STORE LOGICAL WORD	35
ORS	-0602	OR TO STORAGE	32
STI	60604	STORE INDICATORS	48
STA	60621	STORE ADDRESS	35
STD	60622	STORE DECREMENT	35
STL	-0625	STORE INST.LOC.COUNTER	35
STP	60630	STORE PREFIX	35
SXA	60634	STORE INDEX IN ADDRESS	40
SXD	-0634	STORE INDEX IN DECREMENT	41
PAX	60734	PLACE ADDRESS IN INDEX	40
PDX	-0734	PLACE DECREMENT IN INDEX	40
PAC	60737	PLACE COMP.OF ADDRESS IN XR	40
PXA	60754	PLACE INDEX IN ADDRESS	40
PXD	-0754	PLACE INDEX IN DECREMENT	40
CHS	60760	CHANGE ACCUMULATOR SIGN	26
ETM	60760	ENTER TRAPPING MODE	53
LBT	60760	LOW ORDER BIT TEST	49A
RCT	60760	RESTORE CHANNEL TRAPS	53
SLN	60760	SENSE LIGHT ON	52
LTM	-0760	LEAVE TRAPPING MODE	53
PBT	-0760	P BIT TEST	49A
SLT	-0760	SENSE LIGHT TEST	52
NOP	60761	NO OPERATION	49

RDS	60762	READ SELECT	70
LGL	-0763	LOGICAL LEFT SHIFT	29
LGR	-0765	LOGICAL RIGHT SHIFT	30
WRS	60766	WRITE SELECT	70
ALS	60767	ACCUMULATOR LEFT SHIFT	29
ARS	60771	ACCUMULATOR RIGHT SHIFT	29
RQL	-0773	ROTATE MQ LEFT	30
AXT	60774	ADDRESS TO INDEX TRUE	40
AXC	-0774	ADDRESS TO XR COMPLEMENT	40
TXI	61000	TRANSFER WITH XR INCREMENTED	51
TIX	62000	TRANSFER ON INDEX REGISTER	51
TNX	-2000	TRANSFER ON NO INDEX	51
TXH	63000	TRANSFER ON INDEX HIGH	51
TXL	-3000	TRANSFER ON XR LOW OR EQUAL	51
IOCD	0	I/O WITH COUNT AND DISCONNECT	71
TCH	1	TRANSFER IN CHANNEL	74
IORP	2	I/O OF A RECORD AND PROCEED	72
IORT	3	I/O OF A RECORD AND TRANSFER	73
IOCP	4	I/O WITH COUNT AND PROCEED	72
IOCT	5	I/O WITH COUNT AND TRANSFER	72
IOSP	6	I/O UNTIL SIGNAL THEN PROCEED	73
IOST	7	I/O UNTIL SIGNAL THEN TRANSFER	73
LCH		LOAD CHANNEL	69
RCH		RESET AND LOAD CHANNEL	69
SCH		STORE CHANNEL	68
TCO		TRANSFER ON CHAN IN OPERATION	75
TEF		TRANSFER ON CHAN END OF FILE	75
TRC		TRANSFER ON CHAN REDUN CHECK	75
ZAC		ZEROS TO ACCUMULATOR	35A

ALPHABETIC INSTRUCTION LIST

ACL	60361	ADD AND CARRY LOGICAL WORD	27
ADD	60400	ADD	25
ALS	60767	ACCUMULATOR LEFT SHIFT	29
ANA	-0320	AND TO ACCUMULATOR	31
ANS	60320	AND TO STORAGE	31
ARS	60771	ACCUMULATOR RIGHT SHIFT	29
AXC	-0774	ADDRESS TO XR COMPLEMENT	40
AXT	60774	ADDRESS TO INDEX TRUE	40
CAL	-0500	CLEAR AND ADD LOGICAL WORD	35
CAS	60340	COMPARE AC WITH STORAGE	50
CHS	60760	CHANGE ACCUMULATOR SIGN	26
CLA	60500	CLEAR AND ADD	35
CLS	60502	CLEAR AND SUBTRACT	26
DVP	60221	DIVIDE OR PROCEED	26
ENB	60564	ENABLE FROM Y	54
ERA	60322	EXCLUSIVE OR TO ACCUMULATOR	32
ETM	60760	ENTER TRAPPING MODE	53
FAD	60300	FLOATING ADD	28
FDP	60241	FLOATING DIVIDE OR PROCEED	28
FMP	60260	FLOATING MULTIPLY	28
FSB	60302	FLOATING SUBTRACT	28
HTR	60000	HALT AND TRANSFER	49
IOCD	0	I/O WITH COUNT AND DISCONNECT	71
IOCP	4	I/O WITH COUNT AND PROCEED	72
IOCT	5	I/O WITH COUNT AND TRANSFER	72
IORP	2	I/O OF A RECORD AND PROCEED	72
IORT	3	I/O OF A RECORD AND TRANSFER	73
IOSP	6	I/O UNTIL SIGNAL THEN PROCEED	73
IOST	7	I/O UNTIL SIGNAL THEN TRANSFER	73
LAC	60535	LOAD COMP.OF ADDRESS IN XR	39
LAS	-0340	LOGICAL COMPARE AC/STORAGE	50
LBT	60760	LOW ORDER BIT TEST	49A
LCH		LOAD CHANNEL	69
LDC	-0535	LOAD COMP.OF DECREMENT IN XR	41
LDI	60441	LOAD INDICATORS	47
LDQ	60560	LOAD MQ REGISTER	34
LGL	-0763	LOGICAL LEFT SHIFT	29
LGR	-0765	LOGICAL RIGHT SHIFT	30
LTM	-0760	LEAVE TRAPPING MODE	53
LXA	60534	LOAD INDEX FROM ADDRESS	39

LXD	-0534	LOAD XR FROM DECREMENT	40
MPY	60200	MULTIPLY	26
NOP	60761	NO OPERATION	49
NZT	-0520	STORAGE NON-ZERO TEST	50
ORA	-0501	OR TO ACCUMULATOR	32
ORS	-0602	OR TO STORAGE	32
OSI	60442	OR STORAGE TO INDICATORS	48
PAC	60737	PLACE COMP.OF ADDRESS IN XR	40
PAX	60734	PLACE ADDRESS IN INDEX	40
PBT	-0760	P BIT TEST	49A
PDX	-0734	PLACE DECREMENT IN INDEX	40
PIA	-0046	PLACE INDICATORS IN AC	48
PXA	60754	PLACE INDEX IN ADDRESS	40
PXD	-0754	PLACE INDEX IN DECREMENT	40
RCH		RESET AND LOAD CHANNEL	69
RCT	60760	RESTORE CHANNEL TRAPS	53
RDS	60762	READ SELECT	70
RQL	-0773	ROTATE MQ LEFT	30
SCH		STORE CHANNEL	68
SIL	-0055	SET INDICATORS OF LEFT HALF	48
SIR	60055	SET INDICATORS OF RIGHT HALF	48
SLN	60760	SENSE LIGHT ON	52
SLT	-0760	SENSE LIGHT TEST	52
SLW	60602	STORE LOGICAL WORD	35
STA	60621	STORE ADDRESS	35
STD	60622	STORE DECREMENT	35
STI	60604	STORE INDICATORS	48
STL	-0625	STORE INST.LOC.COUNTER	35
STO	60601	STORE ACCUMULATOR	35
STP	60630	STORE PREFIX	35
STQ	-0600	STORE MQ REGISTER	34
STZ	60600	STORE ZEROS	35
SUB	60402	SUBTRACT	26
SXA	60634	STORE INDEX IN ADDRESS	40
SXD	-0634	STORE INDEX IN DECREMENT	41
TCH	1	TRANSFER IN CHANNEL	74
TCO		TRANSFER ON CHAN IN OPERATION	75
TEF		TRANSFER ON CHAN END OF FILE	75
TIF	60046	TRANSFER IF INDICATORS OFF	52
TIO	60042	TRANSFER IF INDICATORS ON	52
TIX	62000	TRANSFER ON INDEX REGISTER	51
TMI	-0120	TRANSFER ON MINUS	49A
TNO	-0140	TRANSFER ON NO AC OVERFLOW	49A
TNX	-2000	TRANSFER ON NO INDEX	51
TNZ	-0100	TRANSFER ON NO AC ZERO	49A
TOV	60140	TRANSFER ON AC OVERFLOW	49A
TPL	60120	TRANSFER ON PLUS	49
TQP	60162	TRANSFER ON MQ PLUS	49A
TRA	60020	TRANSFER	49
TRC		TRANSFER ON CHAN REDUN CHECK	75
TSX	60074	TRANSFER AND SET INDEX	50
TTR	60021	TRAP TRANSFER	53
TXH	63000	TRANSFER ON INDEX HIGH	51
TXI	61000	TRANSFER WITH XR INCREMENTED	51
TXL	-3000	TRANSFER ON XR LOW OR EQUAL	51
TZE	60100	TRANSFER ON ZERO	49
WRS	60766	WRITE SELECT	70
XCA	60131	EXCHANGE AC AND MQ REGISTERS	35A
XCL	-0130	EXCHANGE LOGICAL AC AND MQ	35A
XEC	60522	EXECUTE	49
ZAC		ZEROS TO ACCUMULATOR	35A
ZET	60520	STORE ZERO TEST	50

Alphabetic Instruction List With Flow Notation

Page	Code	Instruction Name	Flow Notation
40	PXD	Place index in decrement	Index register \rightarrow (AC) 3-17. (AC) Q, P, S, 1, 2, 18-35 set to zeros
53	RCT	Restore channel traps	Turns on Trap Control
30	RQL	Rotate multiplier-quotient left	(MQ) shifted left n places. (MQ) S \rightarrow (MQ) 35
48	SIL	Set indicators of left half	(SIL) 18-35 and (SI) 0-17 are compared. If a given position of either group is a 1, a 1 \rightarrow (SI) 0-17. If both are 0, a 0 \rightarrow (SI) 0-17.
48	SIR	Set indicators of right half	(SIL) 18-35 and (SI) 0-17 are compared. If a given position of either group is a 1, a 1 \rightarrow (SI) 18-35. If both are 0, a 0 \rightarrow (SI) 18-35.
52	SLNx	Sense light x on	(SLN) 18-35 turn on sense light x.
52	SLTx	Sense light x test	If light x is on, it is turned off and IC - 1 \rightarrow IC
35	SLW	Store logical word	(AC) P, 1-35 \rightarrow (Y)
35	STA	Store address	(AC) 21-35 \rightarrow (Y) 21-35
35	STD	Store decrement	(AC) 3-17 \rightarrow (Y) 3-17
48	STI	Store indicators	(SI) \rightarrow (Y)
35	STO	Store accumulator	(AC) S, 1-35 \rightarrow (Y)
35	STP	Store prefix	(AC) P, 1, 2 \rightarrow (Y) S, 1, 2
34	STQ	Store multiplier quotient	(MQ) \rightarrow (Y)
35	STZ	Store zeros	Zeros \rightarrow (Y)
26	SUB	Subtract	(AC) - (Y) \rightarrow (AC)
40	SXA	Store index in address	Index register \rightarrow (Y) 21-35
41	SXD	Store index in decrement	Index register \rightarrow (Y) 3-17
52	TIF	Transfer if indicators off	If indicators are off; (TIF) 21-35 \rightarrow IC
52	TIO	Transfer if indicators on	If indicators are on; (TIO) 21-35 \rightarrow IC
51	TIx	Transfer on index	If (XR) > (V); (XR) - (V) \rightarrow (XR) and (TIx) 21-35 \rightarrow IC
49a	TMI	Transfer on minus accumulator	If (AC) S = 1, (TMI) 21-35 \rightarrow IC
49a	TNO	Transfer on no overflow	If no AC overflow; (TNO) 21-35 \rightarrow IC
51	TNX	Transfer on no index	If (XR) \leq (V); (TNX) 21-35 \rightarrow IC and (XR) - (V) \rightarrow (XR)
49a	TNZ	Transfer on No zero	If (AC) Q, P, 1-35 = 0, (TNZ) 21-35 \rightarrow IC
49a	TOV	Transfer on overflow	If AC overflow; (TOV) 21-35 \rightarrow IC
49a	TPL	Transfer on plus accumulator	If (AC) S = 0, (TPL) 21-35 \rightarrow IC
49a	TQP	Transfer on plus MQ register	If (MQ) S = 0, (TOP) 21-35 \rightarrow IC
49	TRA	Transfer	(TRA) 21-35 \rightarrow IC
50	TSX	Transfer and set index	2's complement of TSX location \rightarrow index register; (TSX) 21-35 \rightarrow IC
53	TRR	Trap transfer	(TRR) 21-35 \rightarrow IC
51	TXH	Transfer on index high	If (XR) > (V); (TXH) 21-35 \rightarrow IC
51	TXI	Transfer with xr incremented	(V) added to (XR) (XR), (TXI) 21-35 \rightarrow IC
51	TXL	Transfer on index low	If (XR) \leq (V); (TXL) 21-35 \rightarrow IC
49a	TZE	Transfer on zero accumulator	If (AC) Q, P, 1-35 = 0, (TZE) 21-35 \rightarrow IC
35a	XCA	Exchange accumulator and MQ registers	(AC) S, 1-35 \rightarrow (MQ) and (MQ) \rightarrow (AC) S, 1-35. Zeros \rightarrow (AC) Q, P
35a	NCL	Exchange logical AC and MQ registers	(AC) P, 1-35 \rightarrow (MQ) and (MQ) \rightarrow (AC) P, 1-35. Zeros \rightarrow (AC) S, Q
49	XEC	Execute	See page 49
35a	ZAC	Zeros to accumulator	Zeros are set into (AC) S, Q, P, 1-35
50	ZET	Storage zero test	If (Y) 1-35 are = 0, IC + 1 \rightarrow IC

NOTE: When any transfer instruction does not meet its test, the next sequential instruction is taken.

Alphabetic Instruction List With Flow Notation

Page	Code	Instruction Name	Flow Notation
27	ACL	Add and carry logical word	$(Y) + (AC) P, 1-35 \rightarrow (AC) P, 1-35$
25	ADD	Add	$(Y) + (AC) S, 1-35 \rightarrow (AC) S, 1-35$
29	ALS	Accumulator left shift	$(AC) Q, P, 1-35$ shifted left n places
31	ANA	AND to accumulator *	$(Y) \text{ ANDed with } (AC) P, 1-35 \rightarrow (AC) P, 1-35$
31	ANS	AND to storage *	$(Y) \text{ ANDed with } (AC) P, 1-35 \rightarrow (Y)$
29	ARS	Accumulator right shift	$(AC) Q, P, 1-35$ shifted right n places
40	AXC	Address to index complemented	2's complement of $(AXC) 21-35 \rightarrow$ index register
40	AXT	Address to index true	$(AXT) 21-35 \rightarrow$ index register
35	CAL	Clear and add logical word	$(Y) \rightarrow (AC) P, 1-35$. Zero $\rightarrow (AC) S, Q$
56	CAS	Compare accumulator with storage	If $(AC) > (Y)$; $IC + 1 \rightarrow IC$. If $(AC) = (Y)$; $IC \rightarrow IC$. If $(AC) < (Y)$; $IC + 3 \rightarrow IC$.
35	CLA	Clear and add	$(Y) \rightarrow (AC) S, 1-35$. Zero $\rightarrow (AC) Q, P$
26	DVP	Divide or proceed	(AC) and $(MQ) \div (Y)$. Quotient $\rightarrow (MQ)$, remainder $\rightarrow (AC)$
54	ENB	Enable from Y	(Y) enables the type of traps
32	ERA	Exclusive OR to accumulator *	$(Y) S, 1-35$ exclusive ORed with $(AC) P, 1-35$. If given positions of each register are not $=$, a 1 $\rightarrow (AC)$. If $=$, a 0 $\rightarrow (AC)$
53	ETM	Enter trapping mode	Traps may now occur
28	FAD	Floating point add	$(Y) + (AC) \rightarrow (AC)$
28	FDP	Floating point divide or proceed	$(AC) \div (Y)$. Quotient $\rightarrow (MQ)$, remainder $\rightarrow (AC)$
28	FMP	Floating point multiply	$(Y) \times (MQ)$. Product $\rightarrow (AC)$ and (MQ)
28	FSB	Floating point subtract	$(AC) - (Y) \rightarrow (AC)$
49	HTR	Halt and transfer	Stops computer. $(HPR) 21-35 \rightarrow IC$ with start key depression
39	LAC	Load complement of address in index	2's complement $(Y) 21-35 \rightarrow$ index register
50	LAS	Logical compare ACC with storage	If $(AC) > (Y)$; $IC + 1 \rightarrow IC$. If $(AC) = (Y)$; $IC + 2 \rightarrow IC$. If $(AC) < (Y)$; $IC + 3 \rightarrow IC$.
41	LDC	Load complement of dec. in index	2's complement $(Y) 3-17 \rightarrow$ index register.
47	LDI	Load indicators	$(Y) \rightarrow (SI)$
34	LDQ	Load multiplier-quotient	$(Y) \rightarrow (MQ)$
29	LGL	Logical left shift	$(AC) Q, P, 1-35$ and (MQ) shifted left n places
30	LGR	Logical right shift	$(AC) Q, P, 1-35$ and (MQ) shifted right n places
53	LTM	Leave trapping mode	Traps may not now occur
39	LXA	Load index from address	$(Y) 21-35 \rightarrow$ index register
40	LXD	Load index from decrement	$(Y) 3-17 \rightarrow$ index register
26	MPY	Multiply	$(Y) \times (MQ)$. Product $\rightarrow (AC)$ and (MQ)
49	NOP	No operation	Dummy instruction
50	NZT	Storage not zero test	If $(Y) 1-35 > 0$, $IC + 1 \rightarrow IC$. If $(Y) 1-35 = 0$, next instruction
32	ORA	OR to accumulator *	(Y) ORed with (AC) . If a given position of either register is a 1, 1 $\rightarrow (AC)$. If both positions are 0, 0 $\rightarrow (AC)$.
32	ORS	OR to storage *	(Y) ORed with (AC) . If a given position of either register is a 1, 1 $\rightarrow (Y)$. If both positions are 0, 0 $\rightarrow (Y)$.
38	OSI	OR storage to indicators *	(Y) ORed with (SI) . If a given position of either register is a 1, 1 $\rightarrow (SI)$. If both positions are 0, 0 $\rightarrow (SI)$
40	PAC	Place comp. of address in xr	2's comp $(AC) 21-35 \rightarrow (XR)$
40	PAX	Place address in index	$(AC) 21-35 \rightarrow$ index register
40	PDX	Place decrement in index	$(AC) 3-17 \rightarrow$ index register
48	PLA	Place indicators in accumulator	$(SI) \rightarrow (AC) P, 1-35$. $(AC) S, Q$ set to zeros.
40	PXA	Place index in address	Index register $\rightarrow (AC) 21-35$. $(AC) S, Q, P, 1-20$ set to zeros

(Y) means the entire contents of storage location Y. (AC) 21-35 means only positions 21-35 of the AC are affected.
 \rightarrow Gives the direction of flow. IC means instruction counter. SI means the sense indicator register.

Examples of OR, AND, and exclusive OR:

OR --	1101 ORed with	AND --	1101 ANDed with	Exclusive OR --	1101 ORed with
	0100 gives		0100 gives		0100 gives
	1101		0100		1001

INTRODUCTION & PROGRAMMING QUIZ

1. Interpret the following program and indicate the final contents of:

- a. Acc
- b. Instr. Ctr.
- c. Sense Indicator Reg.
- d. Index Reg. A, B, C'

```

0000   LDI       0020
      1   AXT       7, 1
      2   AXT       4, 2
      3   CLA       0100
      4   SUB       0101
      5   TIO       0010
      6   TIX       4, 1, 1
      7   RIS       0070
     10   NOP
     11   RFT       777777
     12   TRA       0400
     13   TRA       0300

0020   00000.....7
0021   007007007007

0070   000000007777

0100   OCT        30
0101   OCT         2

0201   00000000077
0202   00000000077

0300   XEC        0301
0301   ADD        0201
0302   IDI        0021
0303   ONT        0021
0304   HTR
0305   HTR

0400   XEC        0401
0401   ADD        0202
0402   LDI        0021
0403   OFT        0021
0404   HTR
0405   HTR

```


INTRODUCTION & PROGRAMMING QUIZ

Define the following instructions and by the use of symbols, show what the result of the instruction is.

1. CLS
2. SLW
3. LXA
4. XCA
5. ENK
6. TTR
7. TNZ
8. CAS
9. AXT
10. ETM
11. RQL
12. CLM
13. TCOA
14. ACL
15. TXI
16. LGL
17. NZT
18. TQP
19. MSE
20. BTT
21. XEC
22. PAC
23. TIX
24. TLQ
25. PBT



INTRODUCTION AND PROGRAMMING QUIZ

For each program step, briefly describe what the instruction will do. Also fill in the octal values of the registers indicated below as they occur. At the bottom of the program record the final register values are to be filled in.

<u>Storage Loc.</u>			<u>XR₁</u>	<u>XR₂</u>	<u>XR₄</u>	<u>AC</u>
		ORG 77 ₈				
	START	LXA Z, 2				
100		CAL A, 2				
101		ADD B				
102		STO D, 2				
103		TIX START + 1, 2, 1				
104	ALPHA	CLA* D				
105	SUB	SUB E				
106		PAC 0, 5				
107		TPL ALPHA + 5, 2				
110		HTR 0				
111		HTR 0				
112	A	OCT 50				
113	B	DEC 2				
114	C	DEC 0				
115	D	OCT 112				
116	E	OCT 5				
117	Z	OCT 1				

 FINAL REGISTER VALUES

INTRODUCTION AND PROGRAMMING QUIZ

The following programs have been generated. Diagnose the program and determine the contents of the PC, PR, ACC, and XRA, B, and C upon completion of the program.

If it is determined that the program will not halt, indicate this as an endless loop.

1.

	ORG	77770 ₈
GO	CLA	A
	LLS	5
	CAS	A + 1
	TSX	GO + 6, 1
	TRA	A + 3
	HTR	
NOW	TXI	NOW + 1, 1, 10
	ADD	A + 2
	TIX	NOW + 1, 1, 1
	HTR	
A	OCT	40
	OCT	1770
	OCT	5
	HTR	

```
2.          ORG      777708
           GO      PXD      , 0
           CL      CLA      A
           ST      STT      GO + 4
           SUB     SUB      A + 1
           TSX     TSX      NOW + 4, 0
           NOW    STZ      A
           LDQ     LDQ      A
           LGR     LGR      5
           HTR     HTR
           TXI     TXI      NOW, 7, 10
           A      OCT      7777777
           OCT     OCT      0707070
           HTR     HTR
```

3.

	ORG	77700 ₈
GO	TSX	GO + 4, 5
	CLA	A
	ADM	A + 1
	HTR	
	ETM	
	CLA	A
	TPL	Beta + 3
	ADD	Beta
	STO	Beta
	HTR	
A	OCT	10
	OCT	-40
Beta	OCT	-17
	OCT	1
00000	HTR	
	CLA	0
	ADD	Beta + 1
	STO	5
	TTR*	5

4.		ORG	378
	GO	TXI	GO + 3, 7, 10
		STO	A + 5
		HTR	
		CLA	A
		ADD	A + 1
	NOW	TIX	GO + 4, 1, 1
		TXH	A + 4, 4, 5
		TSX	NOW + 4, 7
		HTR	
		STA	NOW
		TRA*	NOW
		HTR	
	A	OCT	2
		OCT	10
		OCT	50
		HTR	
		HTR	
		HTR	

5.		ORG	77770 ₈
	GO	CLA	ALPHA
		ALS	10
		TPL	GO + 4
		TMI	BETA + 2
		TSX	LOOM, 2
		ADD	ALPHA + 1
		TIX	GO + 5, 1, 1
		TXI	LOOM + 5, 7, 10
		AXT	50, 6
		ACL	ALPHA
	LOOM	LXA	GO + 1, 1
		ADM	ALPHA + 1
		SUB	ALPHA + 1
		TNZ	GO + 5
		HTR	
		STZ	GO + 7
		CAS	BETA + 1
		TRA	BETA + 3
		TRA	GO + 5
		TXI	GO + 8, 7
		HTR	
	ALPHA	OCT	1
		OCT	-4
	BETA	HTR	
		OCT	2010



INTRODUCTION AND PROGRAMMING QUIZ

1. In writing a symbolic program, which of the following would pass as legal symbols?
 - a. LECTER
 - b. MUSICAL
 - c. MIP*A
 - d. RAJAH
 - e. \$ DOUGH
 - f. MELLON
 - g. HA

2. Add the following octal numbers and work each in binary as the machine would perform the addition.
 - a.
$$\begin{array}{r} +524 \text{ Acc} \\ +377 \\ \hline \end{array}$$

 - b.
$$\begin{array}{r} +235 \text{ Acc} \\ -426 \\ \hline \end{array}$$

3. Subtract the following octal numbers and work out in binary as the machine would perform the subtraction.
 - a.
$$\begin{array}{r} +123 \text{ Acc} \\ +256 \\ \hline \end{array}$$

 - b.
$$\begin{array}{r} +341 \text{ Acc} \\ -657 \\ \hline \end{array}$$

4. Multiply in binary.
$$\begin{array}{r} +37 \\ + 7 \\ \hline \end{array}$$

5. Convert to octal.
$$3754_{10}$$

6. Where in the 7090 word are the following fields located?

- a. Tag
- b. Flag
- c. Decrement
- d. Prefix

7. Explain the function performed by each of the following instructions.

- a. CLA 1000
- b. SBM ALPHA
- c. ADD* NET
- d. TRA *+5
- e. MPY RATE, 3
- f. TPL* WHT, 1
- g. LXA MIN, 7
- h. ALS 3
- i. TIX *, 1, 1

8. Change to octal.

- a. 3333_{10}
- b. 1125_{10}
- c. 3641_{10}

9. Change to decimal.

- a. 177_8
- b. 1325_8
- c. 4327_8

TRUE OR FALSE

- 10. All instructions may use address modification.
- 11. The maximum number of shifts which may be performed by an ALS is 72, decimal.
- 12. All information going to or coming from core storage passes thru multiplexor.

13. The instruction ADM treats both factors as positive numbers.
14. The OR and AND instructions always treat the factors as logical words.
15. There is no difference in the operation code (S, 1-11) of a write select instruction for two different data channels.
16. An RCH instruction is necessary following a read select in order to start the selected I/O device in motion.
17. An RCHA will always load channel A with a command.
18. An LCH can be used to synchronize CPU program with an I/O program.
19. A TNX will not alter the specified index register unless the decrement portion of the instruction is equal to or larger than the contents of the Index Register.
20. A TXI is a conditional transfer.
21. Show, by diagram, the configuration of records that will be written on tape by the following program.

	WTDA	1
	RCHA	CW
	LCHA	CW + 2
	HTR	0
CW	IOCP	A, , 20
	IOSP	B, , 10
	IORP	C, , 40
	IOSP	D, , 6
	IORT	E, , 10
	IOCP	F, , 30
	IOCD	G, , 10

22. Location TEMP contains six 6-bit numbers, the left hand position of each number being the sign of the number. Write a program, in symbolics, which will add these numbers together and store the answer in location SUM.

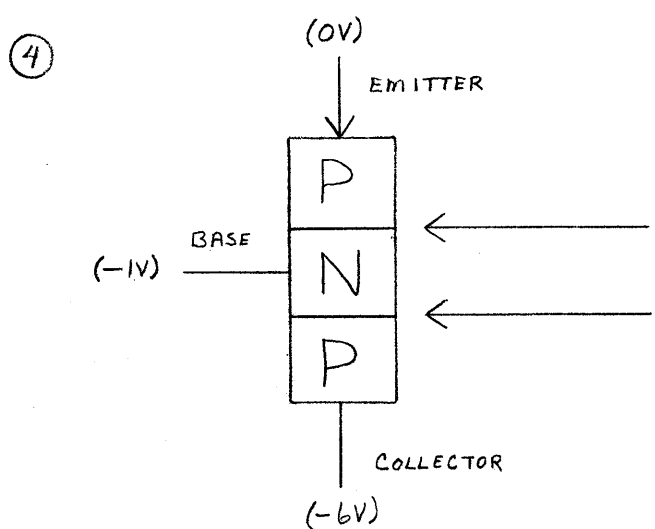
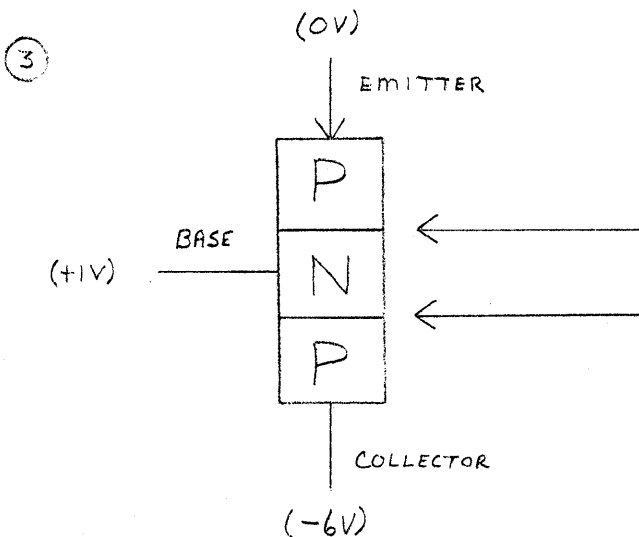
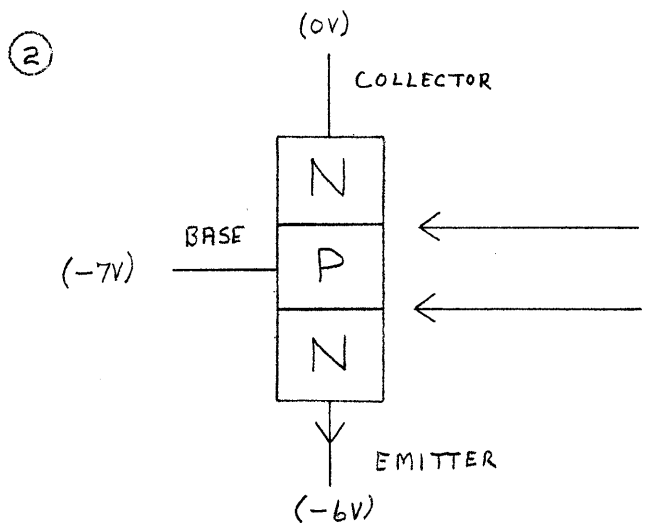
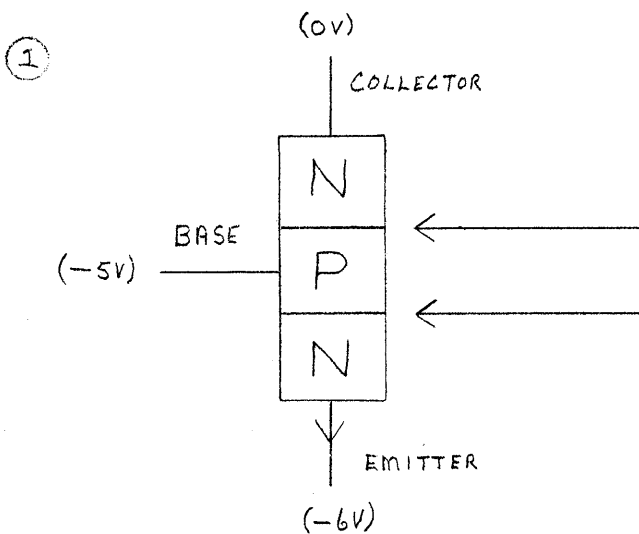
23. On tape #1 there are five 10-word records. The following program is in core. What will be accomplished by the program?

```
          ORG    24
          AXT    5,2
GO        RTBA   1
          RCHA   RC
          TCOA   *
          AXT    9,1
          CAL    A
          ACL    A + 10, 1
          TIX    * - 1, 1, 1
          SLW    A + 10
          WTBA   2
          RCHA   WC
          TCOA   *
          TIX    GO, 2, 1
          HPR
RC        IOCD   A,, 10
WC        IOCD   A,, 11
A         BSS    11
          END    24
```

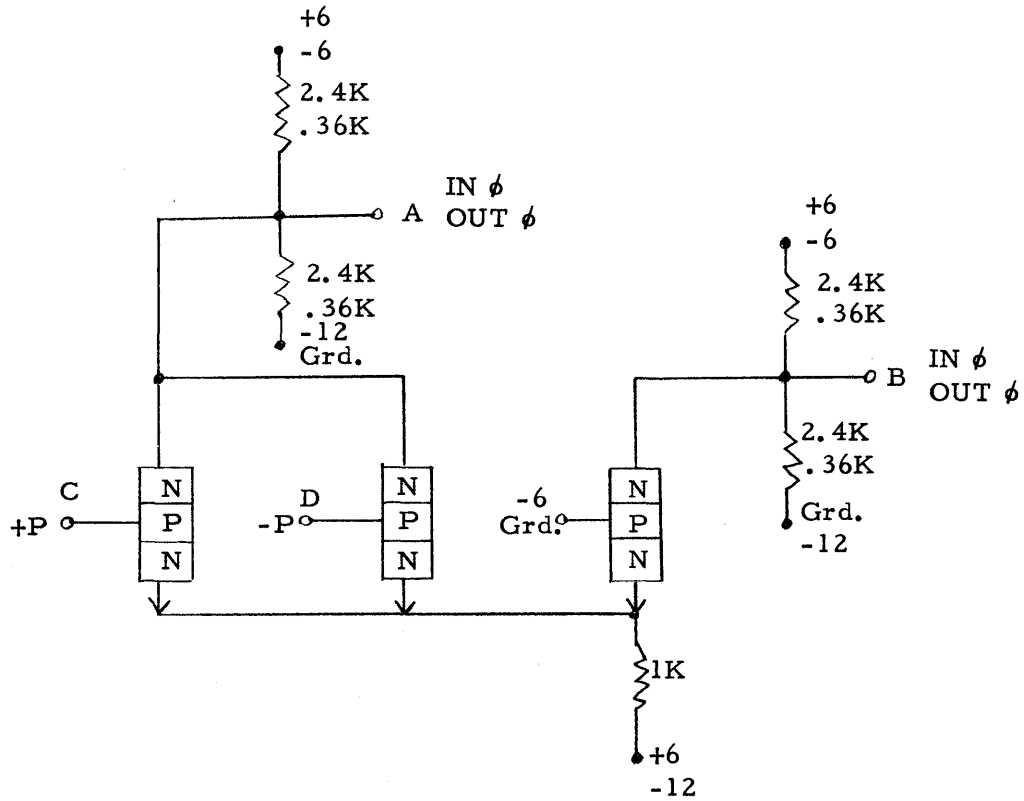
POP QUIZ #1

NAME _____

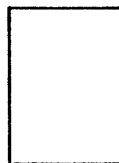
FOR EACH OF THE FOLLOWING TELL WHETHER THE INDICATED JUNCTIONS ARE FORWARD OR REVERSE BIASED. IF THE TRANSISTOR WILL CONDUCT WITH THE VOLTAGES SHOWN, DRAW AN ARROW SHOWING THE PATH AND THE DIRECTION OF THE MAJOR ELECTRON FLOW.



POP QUIZ #2

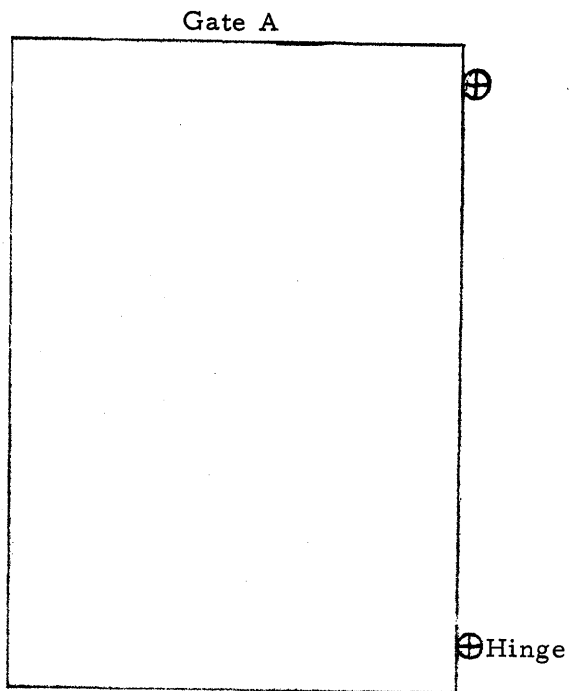


1. Select the correct voltage and resistance values (Black out the wrong resistor or voltage value so that the correct values remain).
2. Shade in the transistor that is conducting.
3. Select the correct name for outputs A and B (Black out incorrect name.)
4. Draw the logic block configuration below for the above circuit and show inputs and outputs.

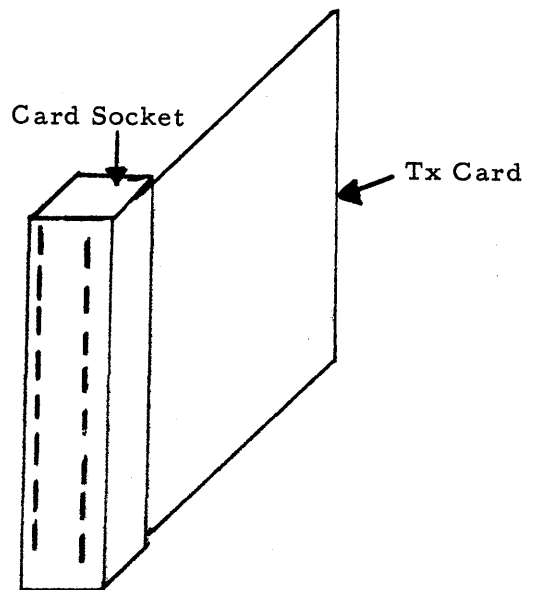


PACKAGING & COMPONENT CIRCUITS QUIZ

CLOSED BOOK



1



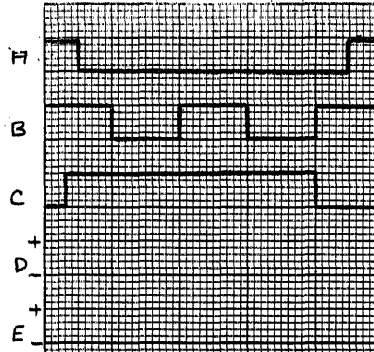
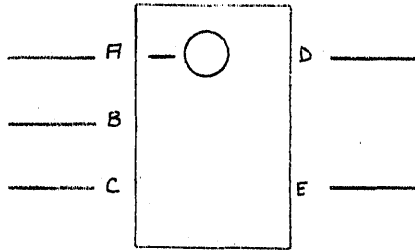
2

label pin connections
on socket

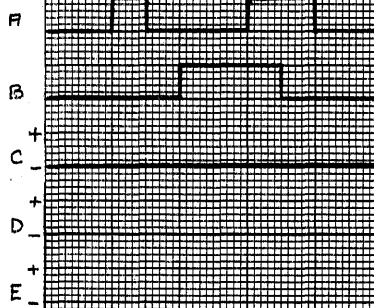
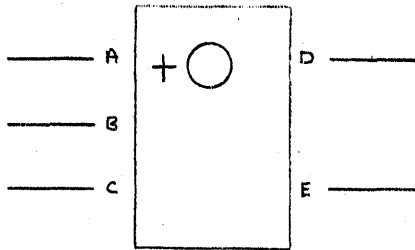
1. Identify on the above gate the following:

- (a) all chassis
- (b) the rows of chassis 1 (label)
- (c) label the columns of chassis 2
- (d) ribbon cables from the tail gate connect to Row _____ on the upper chassis
- (e) Ribbon cables from the tail gate connect to Row _____ on the lower chassis

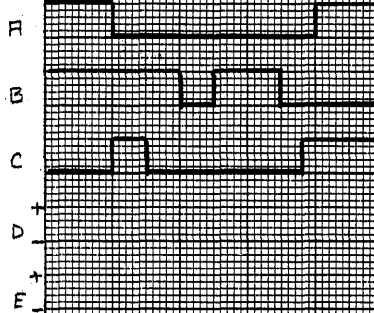
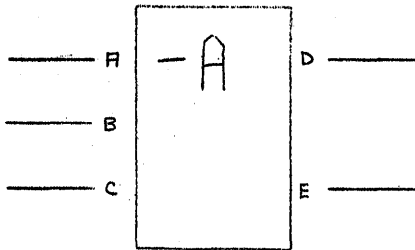
3



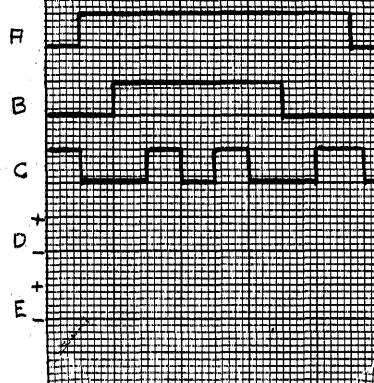
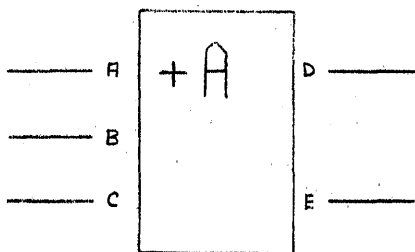
4



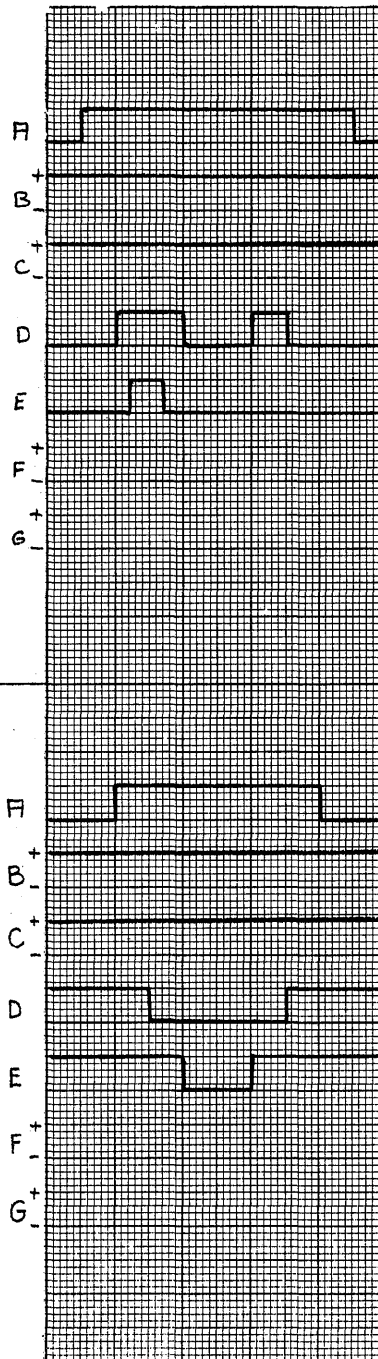
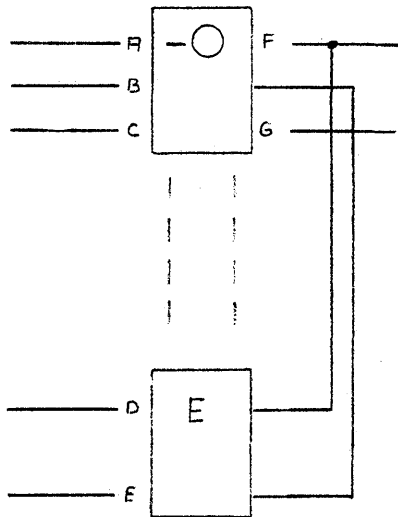
5



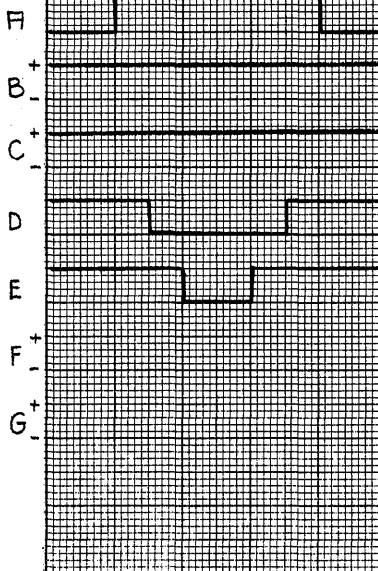
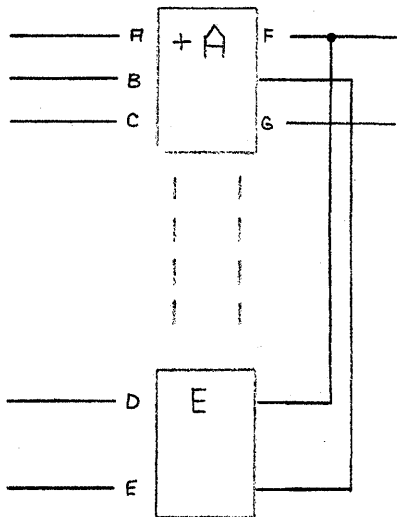
6



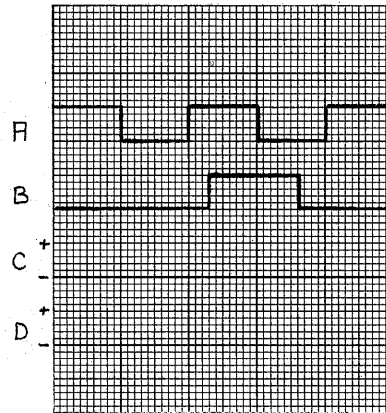
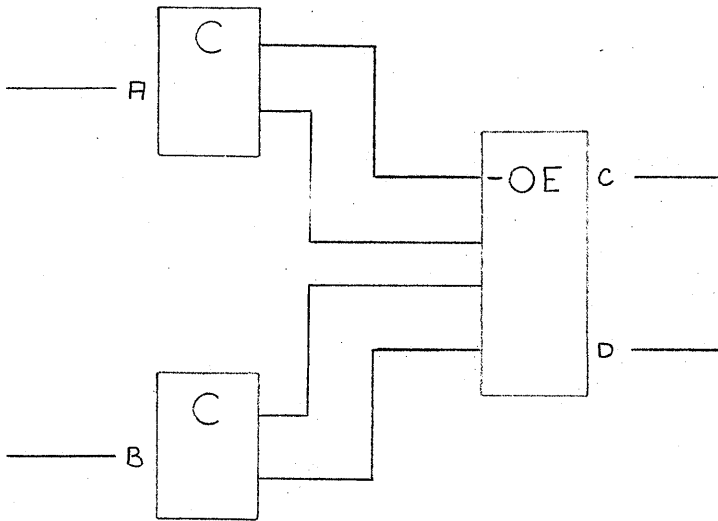
7



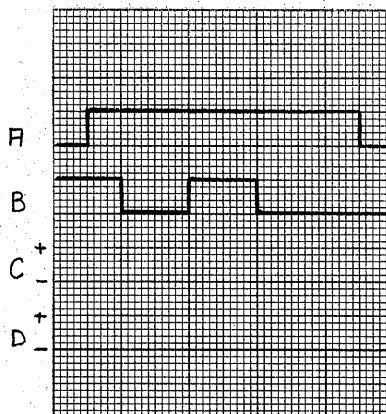
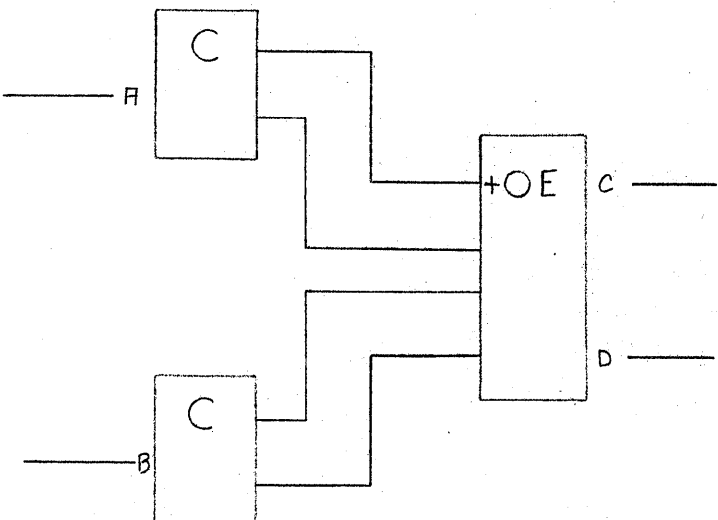
8



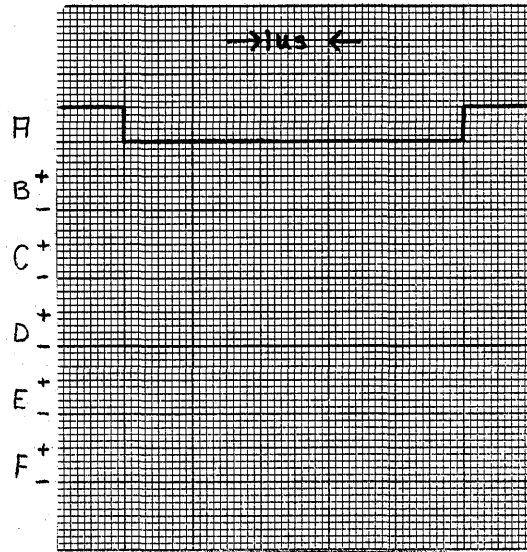
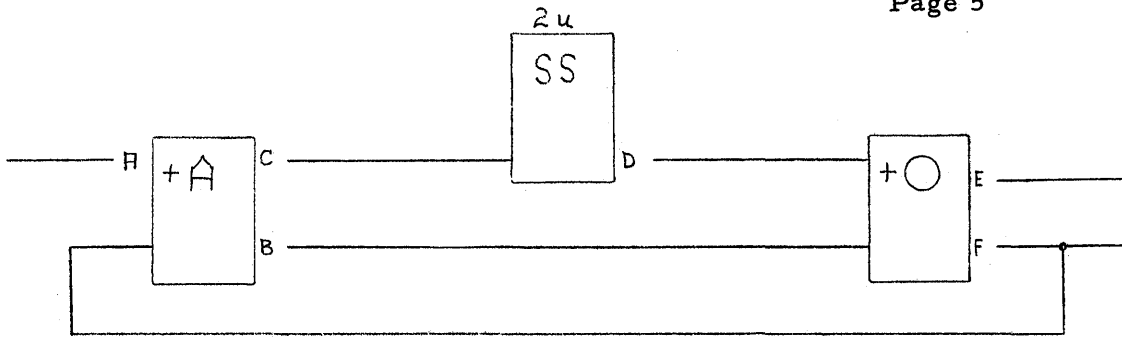
9



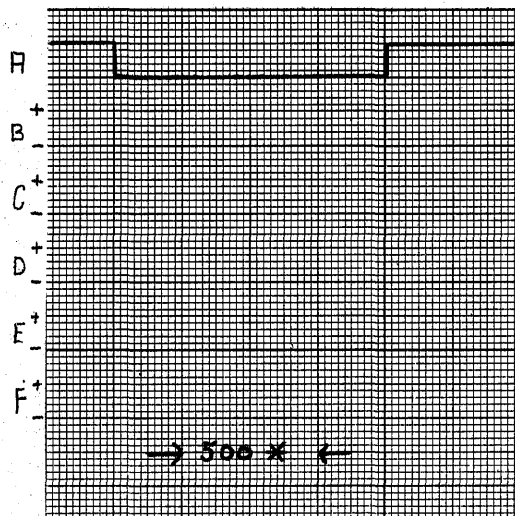
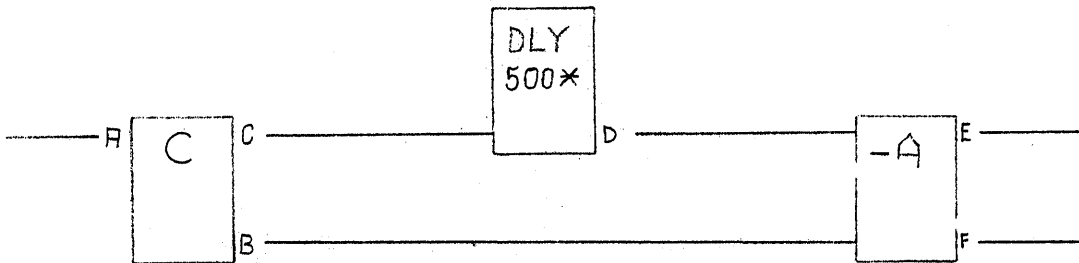
10



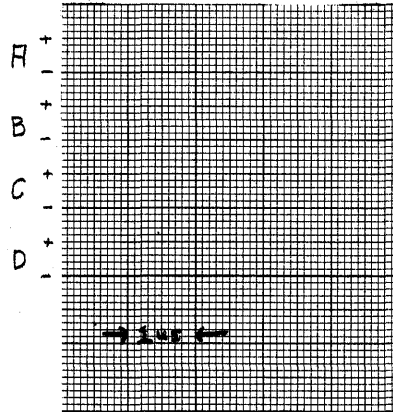
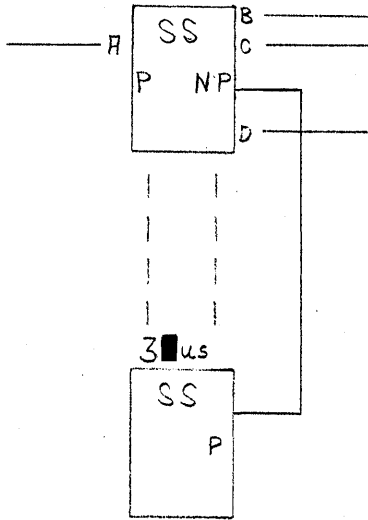
11



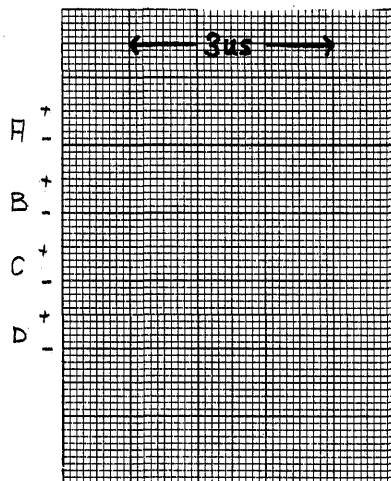
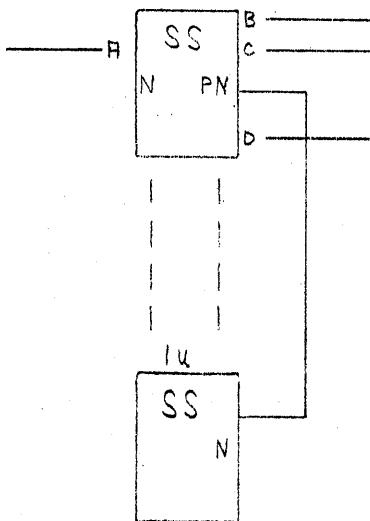
12



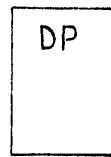
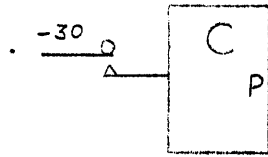
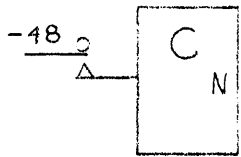
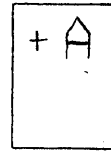
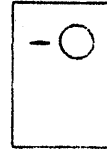
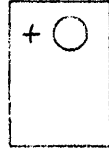
- ⑬ DRAW A 1μS INPUT TRIGGERING PULSE AND THE RESULTING OUTPUT PULSES.



- ⑭ DRAW A 3μS INPUT TRIGGERING PULSE AND THE RESULTING OUTPUT PULSES.

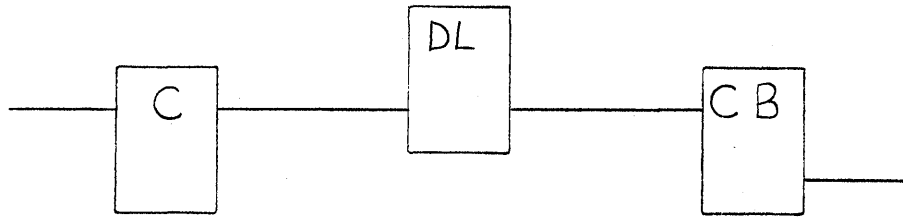


①⑤ WITHIN EACH BLOCK SHOW THE CORRECT LINE REFERENCE SYMBOLS.

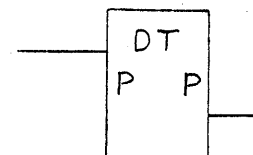
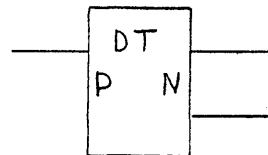
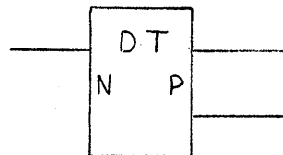
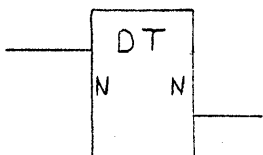


①⑥ WILL THIS CIRCUIT FUNCTION PROPERLY? _____

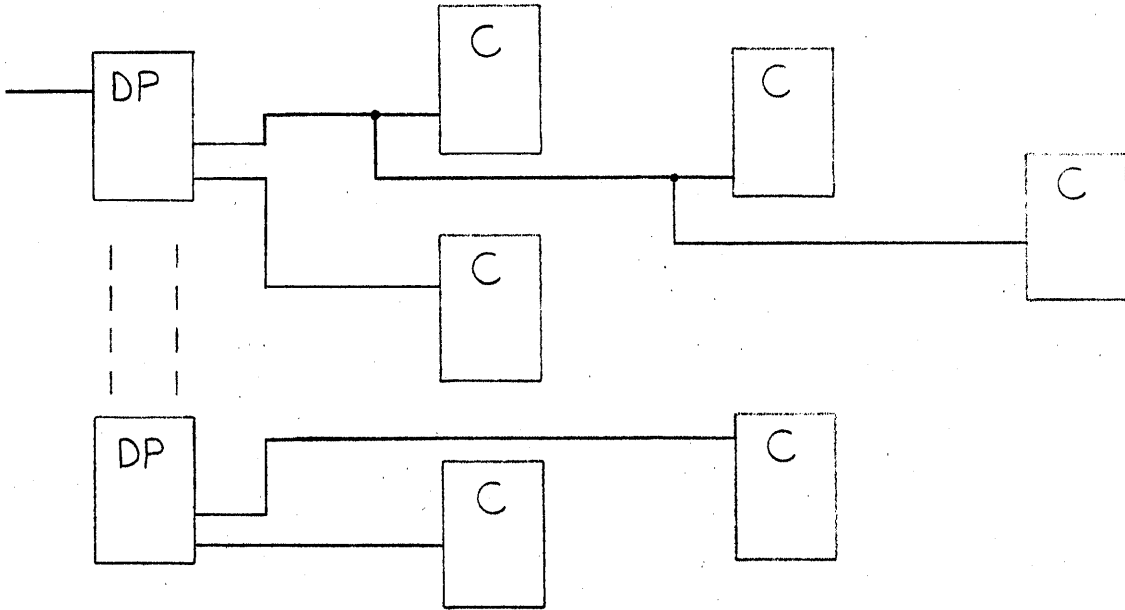
IF NOT, MAKE THE NECESSARY MODIFICATIONS.



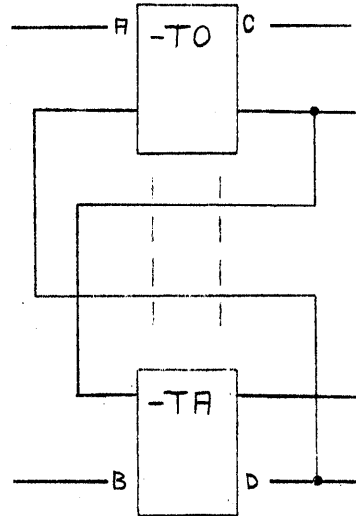
①⑦ CORRECT THE ERRORS (IF ANY) IN THE FOLLOWING DIAGRAMS.

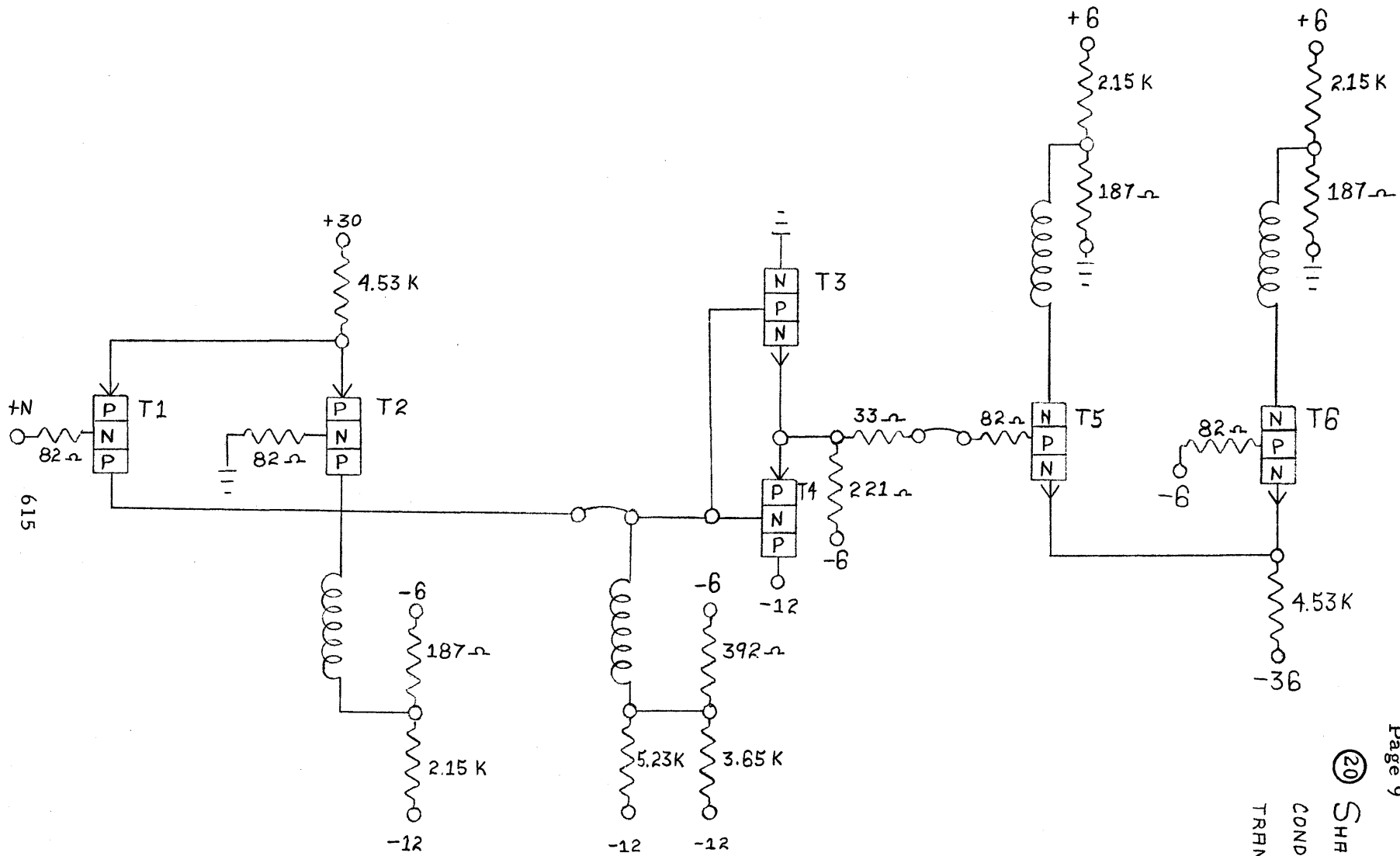


18. CORRECT THE ERRORS (IF ANY) IN THE FOLLOWING DIAGRAM.



19. A. INPUT "A" IS USED TO TURN THIS TRIGGER _____.
- B. INPUT "B" IS USED TO TURN THIS TRIGGER _____.
- C. WHEN THIS TRIGGER IS ON, OUTPUT "D" IS _____.
- D. WHEN THIS TRIGGER IS OFF, OUTPUT "C" IS _____.
- E. LABEL AS PLUS OR MINUS THE INPUT LINES (A + B) FOR THE INACTIVE OR NON-CHANGING STATE OF THE TRIGGER.
- F. THIS TRIGGER IS TURNED ON BY A (NEG, POS) PULSE.
- G. THIS TRIGGER IS TURNED OFF BY A (NEG, POS) PULSE.





(20) SHRDE IN THE
 CONDUCTING
 TRANSISTORS.

POWER SUPPLY QUIZ

OPEN BOOK

ANSWER ALL QUESTIONS TRUE OR FALSE

I. M. G.

1. The MG thermal SW is located under the Drive MTR.
2. MG CB1 must be reset after an emergency off by first restoring the EMERG. OFF key, depressing the PWR ON RESET key, and then resetting CB1.
3. MG CB1 Reset SW is reset by lifting from the tripped position to the ON position.
4. The voltage output of the generator may be adj. with P1 on the MG regulator Control Panel in PCU.
5. The generator field is flashed when the output of the generator falls below 200 volts.

II. PCU

1. There are two supplies in PCU added together to get minus 48 V.
2. The 400 cycle portion of the minus 48 V supply will try to keep power from dropping during momentary line power failures.
3. The DR1 relay which interlocks power from the Channel to the tape drives is in PCU.
4. The HR relays are located vertically between the 208 Bus Bars in the center of PCU.
5. All the CB's for the frame blowers are in PCU.

III. SMS SUPPLIES

1. All the output voltages of the CPU 1 AB supply are low. T4 taps in that supply must be adjusted.
2. The marginal supplies may be adjusted by changing taps on T4 in the standard SMS supplies.
3. A loose fuse clip may cause CB1 to trip in the supply.
4. Minus 12V driver bias must fail for 1 to 5 milli seconds to drop the Core Special Supply CB 2.
5. An open Control Winding in a MAG. AMP. card will cause CB 1 to drop in that supply.

IV. TROUBLE SHOOTING

POWER SUPPLY QUIZ

1. All the Power ON switches on the individual units (CPU 1 and 2, MX, Core, Data Channel) are on. The Power ON Button on the Console is pushed. It is noted the power comes up on all units, but CPU 1 goes right back down. Which of the following could cause this trouble?
 - a. CPU #1 power ON switch 9 not making. 09.02.03.1
 - b. HR10-2 N/O does not make. 09.02.03.1
 - c. HR9-5 N/O does not make. 09.02.03.1
 - d. DR29 N/O points do not make. 09.02.01.1
 - e. HR12-2 N/O points do not make. 09.02.03.1

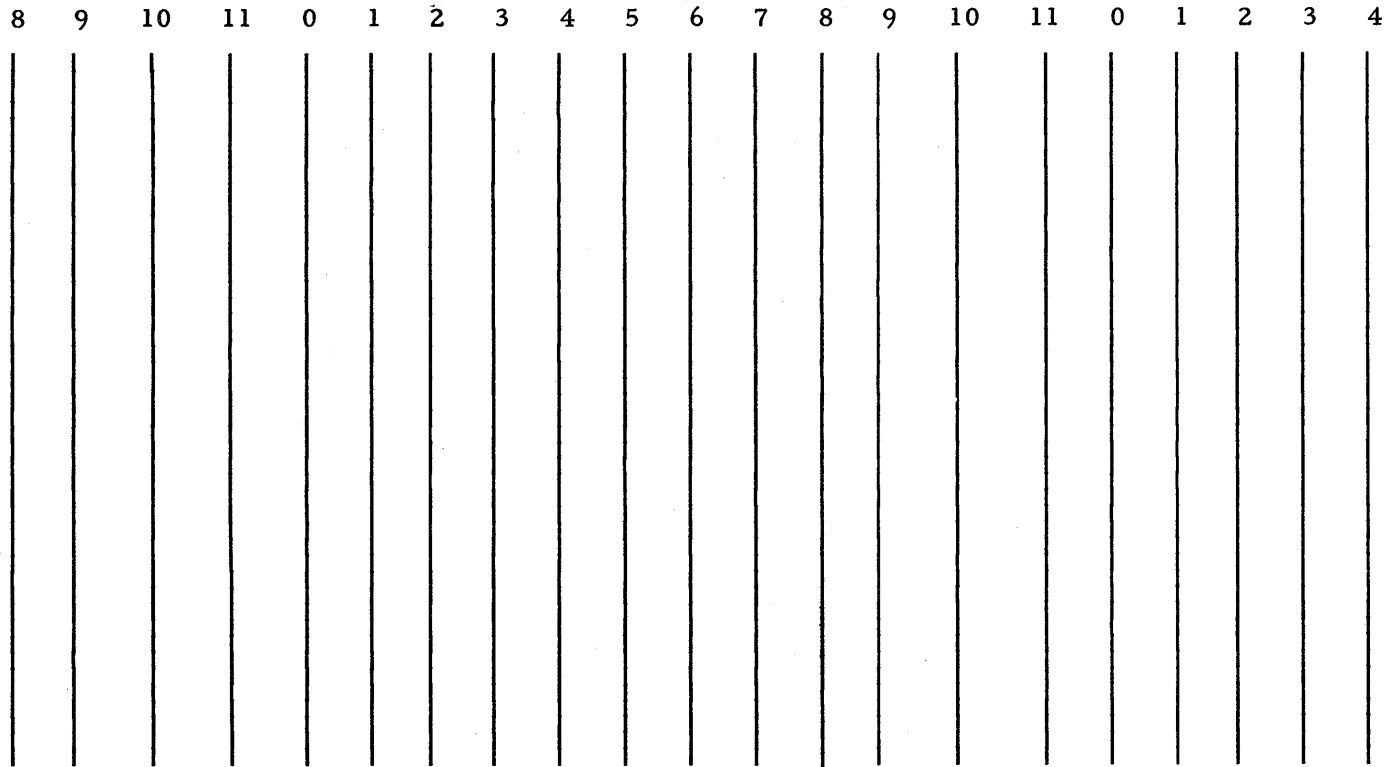
2. All the Power ON switches on the individual units are ON. When the Power ON button on the Console is pushed, the power does not come up at all to CPU 2. Which of the following could cause this? (All other units work normally.)
 - a. HR 23-3 N/O does not make. 09.02.02.1
 - b. HR 23-2 N/O does not make. 09.02.04.1
 - c. CB4 SSW is open. 09.02.04.1
 - d. HR 37 N/O points do not make. 09.02.02.1
 - e. HR 23-5 N/O does not make. 09.02.04.1

3. While the system is running a program, power drops on Data Channel A. On investigating we find that CB1 for gate A-B supply & CB1 for gate C-D supply are tripped. Which of the following could cause the trouble?
 - a. Blower motor on gate A short circuited
 - b. -12V fuse on the A-B power supply is open (Fuse 6 on 09.03.20.1)
 - c. HR1-1 N/C fails to make (09.02.02.1)
 - d. DR31-1 N/O fails to make (09.02.02.1)
 - e. DR16 coil open (09.02.02.1)

4. When taking a Power ON sequence, the MG starts but DC does not come up on any module. Upon investigating we find that the Power ON Variac (09.02.02.1) does not go through a cycle. Which of the following could cause this failure?
 - a. HR1-3 fails to make (09.02.02.1)
 - b. CB29 is tripped (09.02.02.1)
 - c. CB35 is tripped (09.05.31.1)
 - d. HR1-4 fails to make (09.02.02.1)
 - e. DC OFF SW is open (09.02.01.1)

CYCLE AND "I" TIME QUIZ

1. Clock trigger zero turns off when clock trigger _____ turns on.
2. A loss of either the odd or even clock drive pulses would result in a clock running at half its normal speed. (True or False?)
3. An A0 (D1) pulse in multiplexor would be an _____ pulse going to data channel even though the same pulse would be an _____ pulse in main frame.
4. Explain the purpose of the CP set pulses.
5. Bringing up "Go to I Time" signifies next cycle will be an I cycle. (True or False?)
6. What logical purpose are the multiple time error trigger circuits used for? Be specific.
7. An A0 (D1) pulse is the result of gating the Zero Clock Trigger on with odd clock drive. (True or False?)
8. What determines the cycle following the I cycle of any multicycle instruction? Be specific.
9. Systems 08.00.47.1. What is the purpose of the delay at 4C?
10. What method is used to send the machine to "I" time automatically after completion of the previous instruction?



Draw a sequence chart of I time showing the timing and duration of the following pulses

- | | |
|-------------------|--------------------|
| 1. Advance P. C. | 6. S. B. to P. R. |
| 2. S. B. to S. R. | 7. A. D. to A. S. |
| 3. A. S. to A. R. | 8. Set tag reg. |
| 4. P. C. to A. S. | 9. S. R. to A. D. |
| 5. A. R. to P. C. | 10. A. S. to A. R. |

1. Why do we take the A. R. to P. C. during an I cycle?
2. What type of instruction causes A. S. to S. C. ? What is in the A. S. at the time we set the S. C. on an I cycle?
3. What prevents the P. C. from going to the A. S. if the next cycle is not an I cycle?
4. What prevents sending the A. R. through Multx. Address Switches to Memory ? Why is it the only thing to prevent gating out the A. R. ?
5. What positions of the P. R. is POD developed from? What positions of the P. R. is SOD developed from and when do we gate it out for use?

CYCLE AND "I" TIME QUIZ

1. Give the Primary Op and Secondary Op decoder lines, with systems pages, that will be up for the following instructions:
 - a. Rewind 1201
 - b. LLS 27₈
 - c. TXI 100; 1, 1
 - d. CLS 100
2. Explain in your own words the flow of information during the I cycle in the 7090.
3. What is the purpose of routing address registers to program counter during the I cycle?
4. Positions 1 and 2 of an operation code are checked in the SR to determine where to position the operation code in the program register. (True or False?)
5. After the information is set into the program register, it remains there until _____.
6. The program register is loaded from _____.
7. What determines if the shift counter will be set during I time?
8. All secondary operation codes end in even numbers. (True or False?)
9. On 3.05.07.1 +O block at 3D, Pin B is shorted to -N. Program Counter is reset to zeros. What is in the Program Counter after 8 step pulses?
10. CLA 00040 is located at address 00130. On 3.06.16.1, Pin H of C block at 5E is shorted to +P. Where will the next instruction come from?
11. CLA 00040 is located at address 00000. Where will the second instruction come from if pin B of +A at 5F is shorted to +P? Page 3.06.05.1
12. A TIX instruction is located at address 00000. When it is brought from core to the Program Register, what instruction will be executed with the following bug on the machine?
Page 3.04.03.1, +O at 3B
Pin A shorted to +N

Cycle and "I" Time Quiz

-2-

13. After 5 octal had been set to the program counter, the line "minus to step PC 15" systems page 03.05.07.1 shorted to a minus N level. What would the contents of the program counter be after it is stepped once?
14. What is the purpose of the "SC Set Gate" on page 03.04.17.1?
15. We have the following program in main frame. Assume a B cycle is asked for during each I cycle. Draw new chart showing when B cycles occur.

Add	I
	E
Mul	I
	E
	L
	L
CAQ	I
	L
	E
	E

CYCLE AND "I" TIME QUIZ

1. Define the following terms:
 - a. Indirect addressing
 - b. Effective addressing
2. If an indirectly addressed instruction brought out an instruction that was indirectly addressed, the CPU would force 2 consecutive IA-E cycles. (True or False)
3. What input to 3G on 2.12.19.1 generates XR to AD during an IA-E cycle?
4. On 08.00.12.1, the +A at 5B lets the machine go to E time from an IAE cycle. What AND circuit's only purpose is to put the machine into L time after an IAE cycle?
5. Index Register A contains 00010. On 2.12.76.1 pin B of 2B is shorted to +P. The contents of what location will be placed in the Accumulator as the result of executing the instruction located at 00100?

00100	CLA*	00200.1
00140	SUB*	00150.1
00150	STO*	00150.1
00200	STQ*	00150.1

6. On page 2.10.65.1 pin B of 2B is labeled +P IA Tgr. on, it goes to 2.12.16.1. What is the purpose?
7. The following program is in core:

00000	AXT	3,1
1	CLA	7
2	ADD	10,1
3	TIX	2,1,1
4	SUB	7,1
5	HTR	1
6	OCT	10
7	OCT	12
10	OCT	1

The machine has the following bug on it. Page 03.04.03.1, + O at 3B, pin A is shorted to +N. What is the contents of the Accumulator when the machine halts? Where will the machine halt?

8. All three index registers have all ones in them. On 02.12.19.1, Convert Block at 2E, pin B is shorted to -P. CLA 0440 is executed. Will the contents of 0440 be placed in the accumulator?
9. Which of the following will not turn off the Master Stop Trigger?
 - a. Clear
 - b. Display Storage
 - c. Display Effective Address
 - d. Single Step
 - e. Enter Instruction
 - f. Enter MQ
10. Why is the program Register reset during the initial I cycle under each depression of the Single Step key?
11. Why is the Op. Panel Control Trigger turned on during the first I cycle of a Display Storage operation?
12. Why is the Tag Register reset during a Display Effective Address operation?
13. Why is it necessary for the release of any key to reset the Manual Control Trigger?

PRELIMINARY INSTRUCTIONS QUIZ

1. With the following program in Core Storage, Reset and Start is depressed in that order. The computer halts with 0007 in the PC. This result was noted in the AC +050000000560₈. Note: All addresses are octal.

0000	CLA	500
0001	ADD	501
0002	SUB	502
0003	STL	503
0004	ADD	501
0005	XEC	503
0006	HPR	
500	+100 ₈	
501	- 50 ₈	
502	-100 ₈	
503	ADM 0002	

- a. What should the result in AC be with no machine malfunction?
Answer in Octal.
- b. What machine malfunction would result in the answer that was in the AC? Designate which Systems page, logic block and line level.

PRELIMINARY INSTRUCTIONS QUIZ

Note: Multiple choice questions MAY have more than one correct answer. If so, give all such answers.

1. Instruction: CLA 00500
At 00500 -000----055 C(AC) + 000-----000
After completion of the instruction the Acc contains -000,....000

Which of the following could cause this?

- a. 02.12.31.1 pin C of 3B stays at -P level.
- b. 02.12.15.1 pin C of 1E stays at -P level.
- c. 02.12.14.1 pin G of 1E stays at -N level.
- d. 02.12.50.1 pin H of 1D stays at +N level.

2. Instruction: ACL 00100
At 00100 (S-35) 000000000001
ACC (P-35) before 777777777777
ACC (P-35) after 777776000000

Which of the following could cause this?

- a. 02.03.19.1 pin G of 1H stays -N.
- b. 02.03.19.1 pin G of 3F stays -N.
- c. 02.03.19.1 pin G of 1F stays +N.
- d. 02.02.19.1 -O at 01B3D04 pin G stays plus P and pin H stays minus P.
- e. 02.03.19.1 DP at 2G pin F stays plus P.

3. Instruction: STO 00100
ACC (S, 1-35) 777777777777
00100 before 123456765432
00100 after 000000000000

Which of the following could cause this?

- a. 02.09.00.1 pin G of 5A stays -N.
- b. 02.12.01.1 pin G of 2A stays -P.
- c. 02.09.01.1 pin B of 2E stays -P.
- d. 02.09.01.1 pin B of 3A stays -P.
- e. 02.05.00.1 pin G of 3H stays +N.

4. Instruction: STZ 00100
00100 before 123456765432
00100 after 000000700000

4. Cont.

Which of the following could cause this?

- a. 02.09.01.1 pin A of 4I stays -N.
- b. 02.09.01.1 pin G of 2B stays -N.
- c. 02.09.01.1 pin G of 2F stays +N.
- d. 01.00.00.1 pin GH of 5G stays -P.
- e. 01.00.00.1 pin B of 3D stays +N.

5. Instruction: 00100 TRA 00500
Next instruction taken from 00000

Which of the following could cause this?

- a. 03.06.16.1 pin G of 2E stays -P.
- b. 03.06.16.1 pin BA of 3A stays -N.
- c. 03.06.05.1 pin B of 2H stays -N.
- d. 02.10.09.1 pin F of 3A stays -P.
- e. 02.10.07.1 pin G of 2F stays -N.

6. Instruction: 00715 STL 00500
00500 before 252525252525
00500 after 000000000716

Which of the following could cause this?

- a. 02.09.00.1 pin GH of 5D stays -N.
- b. 02.09.00.1 pin G of 4B stays +P.
- c. 02.09.00.1 pin GH of 5G stays +P.
- d. 02.09.00.1 pin G of 5A stays +N.
- e. No error has occurred. This is the normal operation of STL.

7. The following program is in storage:

```
00001 CLA 00100
00002 ADD 00101
00003 STO 00102
00004 STL 00102
00005 HPR
    ---
    ---
    ---
    ---
00100 111111111111
00101 222222222222
00102 333333333333 after the STO instruction
00102 333333333337 after the STL instruction
```

7. Cont.

Which of the following could cause this?

- a. 02.09.01.1 pin A of 4I stays -N.
- b. 02.09.00.1 pin G of 5A stays -N.
- c. 02.15.62.1 pin C of 3F stays +P.
- d. 02.12.50.1 pin B of 4B stays -N.
- e. None of the above.

7302 CORE STORAGE QUIZ

1. Inputs to 7302 Core Storage are: (Select the correct answer (or answers) from the following.)
 - a. MAR lines from Multiplexor Address Switch
 - b. Strobe Pulse
 - c. X and Y driver timing pulse
 - d. Memory Select pulse
 - e. Store control lines
 - f. Data in gate
 - g. Rd/Wr trigger set and reset
 - h. Memory data in Bus
2. A Core Storage cycle is initiated by _____.
3. During the read portion of a Core Storage cycle, 72 positions of the core array are read out. Determination of which 36 positions will be operated upon is done by _____.
4. In a "Store" operation, the unwanted portions of the addressed word are destroyed during read time by:
 - a. Blocking the DIG to the specified positions of MDR.
 - b. Resetting MDR after the Read portion of the cycle and before the new word is brought in.
 - c. Blocking the strobe pulse to the specified Sense Amplifiers.
 - d. Blocking the DOG to the specified positions of MDR.
 - e. Blocking X and Y drive line current to the specified portion of the addressed word.
5. How many input lines are feeding any one X matrix switch?
 - a. 2
 - b. 16
 - c. 32
 - d. 33
 - e. 256
6. Selection of a particular Y matrix switch within any one segment is accomplished by
 - a. The X segment gate
 - b. The Y segment gate
 - c. Coincidence of the X and Y segment gates
 - d. Physical winding of the matrix switch cores
 - e. MAR 11 through 17 outputs

7. Name 3 methods used to reduce noise due to half-select currents.
8. During rewrite time, the decision to write "1" or "0" into storage is determined by
 - a. Z driver timing
 - b. MDR outputs
 - c. DOG and MDR outputs
 - d. MAR 4 & 5 outputs
9. "Or-to-Storage" is accomplished by
 - a. Taking two Memory Cycles, setting MDR from Sense Amplifiers on the first cycle and then setting MDR from MDBI on the second cycle.
 - b. Writing into the selected address from MDBI without first reading out of the selected address.
 - c. Taking a normal "Store" operation and turinging on the Read-out trigger with an ORS line.
 - d. Taking a Store cycle and holding the store partial word lines in their inactive condition with an ORS line.
 - e. Taking a Read-out cycle, OR'ing the words in CPU and then taking a Store cycle to store the "OR" configuration.
10. Strobe Pulses are
 - a. Generated from the matrix switches of the X drive lines and time staggered through the core at nine different time intervals.
 - b. Generated from the X and Y drive lines and staggered through the core storage at five different time intervals.
 - c. Generated from the X matrix switches and staggered through the core core storage at four different time intervals.
 - d. Generated from Y matrix switches and time staggered through core storage at nine different intervals.
 - e. Generated from Y matrix switches and time staggered through core at eight different intervals.

CORE STORAGE QUIZ

1. The three types of signals received by the 7302 from C. P. U. are:
 - a.
 - b.
 - c.

2. Under the following conditions, the largest number of strobe pulses are generated during the _____ .

3.
 - A. CLA 100
 - B. STO 500
 - C. STA 501
 - D. STP 502

3. With the above conditions, the smallest number of strobe pulses will be generated during the _____. This operation will generate _____ strobe pulses.

4. During the read portion of a core cycle, the _____ drive line is activated first. Why?

5. How is the strobe pulse initiated?

6. A core diagnostic shows that the following addresses always contain all 1's even when all zeros were supposed to have been stored.

30000 through 37777 and
70000 through 77777

This trouble could be caused by:
 - a. The IRG's at 3E and 3F always have a minus on Pin S - Systems 01.18.00.1
 - b. A constant down level at Pin D of convert 4A - Systems 01.16.01.1
 - c
 - c. A constant +P from the +A at 3B - Systems 01.07.00.1
 - d. A constant -P from the +A at 3H - Systems 01.07.00.1

7. Name the data inputs to the memory data register and explain what gates them into the register.

Core Storage Quiz

06.02
Page 2

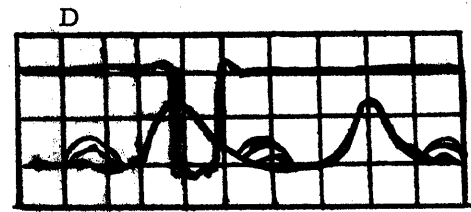
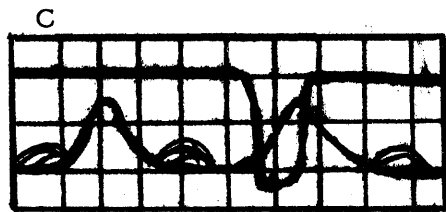
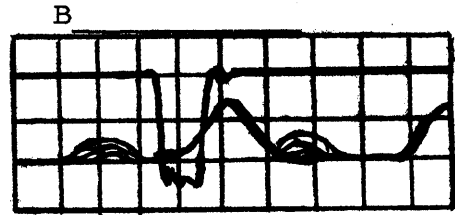
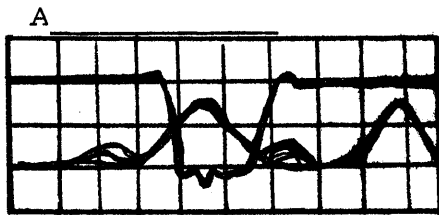
8. The following Y G core drivers should conduct during the read portion of a cycle when the address 54545 is decoded. (Answer each True or False).
 - a. DC at 1C Systems 01.10.01.1
 - b. DC at 3F Systems 01.10.05.1
 - c. DC at 3F Systems 01.10.01.1
 - d. DC at 2E Systems 01.10.01.1

9. The timing error trigger (Systems 01.12.02.1) prevents the data channel from starting a core cycle when the core is already performing a cycle for C. P. U. (True or False -- If False, what does prevent this?)

10. Can the memory data register ever receive more than one data input in a single cycle? If so, under what condition?

06.10.1 7302 A EXAM.

1. The _____ and _____ volt supplies are in one power supply unit.
2. The X and Y drivers are located in gate _____ panel _____.
3. In reference to the Array, where are the Z terminating resistors located?) _____
4. A minimum D level is _____ to _____ volts.
A maximum D level is _____ to _____ volts.
5. DEFL card pins J and _____ are used for _____. Pins _____, _____, _____, and _____ are used for voltage.
6. An external load is always used to develop the output level of an _____ (AOE, AOC, AOL).
7.
 - A. X seg. selection is from MAR positions _____.
 - B. X address decoding is from MAR positions _____.
 - C. Y seg. selection is from MAR positions _____.
 - D. Y address decoding is from MAR positions _____.
 - E. Z seg. decoding is from MAR _____.
 - F. Sense segments are denoted by MAR _____.
8. If you were scoping the sense amps. to look at the strob timing in relation to the data bit read which of the following wave forms would you expect to see IF THE 7302 A WERE WORKING NORMALLY?



.2 Usec/Div.

Answer True or False--There may be more than 1 correct.

9. The system is working normally. Power goes down on the 7302A. You depress the PWR ON key but the PCU variac doesn't even start a cycle. Without any other information, which of the following could be the cause?

- A. Air flow sw. in B2 stays open.
- B. The -20, +20V. sensing relay coil is open
- C. PWR ON SW 1 stays open.
- D. -12V. CP is tripped.
- E. CB2 ssw-c stays open.
- F. IBM 091 transistor on ALD page 01.07.02.0 is open.

- 10 The one Check runs OK but the Zero Check gets an error stop for every address from the CE panel. All customer and diagnostic programs run OK Which could be the cause?

- A. ALD 01. 10. 05. 1 3H pin D stays at +D
- B. ALD 01. 10. 07. 1 5A pin E stays at +D
- C. ALD 01. 24. 04. 1 1I pin Q stays at -P
- D. ALD 01. 24. 04. 1 1H pin H stays at -D
- E. ALD 01. 24. 02. 1 3F pin C stays at +D

11. The 7302A has been functioning properly, taking a cycle each time it received a Memory Select pulse. Suddenly it stops and no longer takes a cycle even though memory select pulses are being received. Which of the following could be the cause?

- A. ALD 01. 10. 01. 1 3C pin E stays +D
- B. ALD 01. 10. 01. 1 3C pin E stays -D
- C. ALD 01. 10. 02. 1 5G pin Y stays +D
- D. ALD 01. 10. 01. 1 5C pins EG stay +D
- E. ALD 01. 10. 01. 1 4B pin B stays +D

12. Writing ones and testing from the CE panel we find that we can't write any bits in the following addresses.

16,000 to 17,777

36,000 to 37,777

56,000 to 57,777

76,000 to 77,777

Which of the following could be the cause?

- | | | | |
|-------------------|----|------------|-------------------|
| <u> </u> | A. | 01.10.01.1 | 1A pin L stays +D |
| <u> </u> | B. | 01.10.05.1 | 5E pin P stays -D |
| <u> </u> | C. | 01.13.06.1 | 5H pin G stays +D |
| <u> </u> | D. | 01.12.06.1 | 5H pin G stays +D |
| <u> </u> | E. | 01.13.01.1 | 4E pin L stays +D |

TRUE or FALSE - Explain all FALSE answers

1. During a MPY instruction it may be necessary to perform a complement add.
2. During the "E" cycle of MPY, the MQ is loaded with the multiplier.
3. To realize any increased speed during MPY, the multiplier must contain at least two successive zeros.
4. If AC35 could not be set to a one, the AC and MQ would always contain all zeros after the completion of MPY.
5. The accumulator is complemented prior to the reduction cycles on DVP.
6. A MPY instruction with a multiplier of all zeros will end operation and not attempt a multiplication.
7. AC overflow is possible during MPY.
8. If a Q carry is noted during the divide test of DVH, the machine will halt.
9. The divisor is brought to the SR during the "E" cycle of a DVH or DVP.
10. The sign of the remainder is set to agree with the sign of the quotient.

Essay - State Facts BRIEFLY

11. Convert the following Octal numbers into a normalized Floating Point format, Octal.
 - (a) 73.43_8
 - (b) 21.4_8
 - (c) $.0034_8$
12. On the 7090 only address switches 12-17 are gated to the shift counter on VDH + 0224 or VDP + 0225. Why?
13. By what method is the AC cleared on MPY?
14. What occurs if the multiplicand is zero during MPY?
15. What is considered the fraction portion of a F. P. number? The characteristic?
16. What characteristic would indicate an exponent of zero?

17. What limitations are placed on the size of the characteristics? When these limits are exceeded, what will occur?
18. What occurs during 1st and 2nd step of FAD that is necessary before two F. P. #'s may be added?
19. What is meant by 7090 double precision?
20. Is overflow possible during second step L time of FAD? Explain.
21. During third step L time of FAD, with signs alike, why is a possible adder 9 carry allowed to go into Adder 8?
22. Can the accumulator overflow on a FAD instruction? If yes, give example of Floating Point numbers that would overflow.
23. What is the purpose of taking ones to adders Q, P, 1 on 1st step of UFM and FMP?
24. Why is the MQ zero tested during the third step L time of a floating add instruction?
25. During FMP second step "L" time, why does the instruction only attempt to normalize one place?
26. During MPY and FMP, what function does the Pre-Sense tgr. serve?
27. Under what conditions is MQ underflow possible during FMP?
28. Why is the AC zero tested during 4th step of FAD?
29. When normalizing the result during 5th step FAD, what must be done to the AC characteristic?
30. What indication will the programmer have as to what type of F. P. trap occurred?

DATA CHANNEL

QUIZ I

1. What circuit makes it impossible for an RCH to hang up in L time if the addressed channel is connected to the system?
2. What is the purpose of having the line "busy" initiate a non data disconnect?
3. What conditions are necessary to turn on the channel in Use Trigger? When this trigger is on, what unit does it send its indication to?
4. What is the purpose of the Delay Counter?
5. The BCW or BDW tgr. must be on to initiate a B cycle. True or False?
6. Define data selected.
7. List four non-data selects.
8. A select instruction will hang up in MF unless it can turn on the _____ in DC.
9. The channel interlock prevents all Data Select instructions and all non-Data Select instructions from ending operation. True or False?
10. If there are 3 channels on a 7090 system, what prevents the most remote channel from taking priority from the least remote channel when the least remote channel is trying to retain priority at the same time the most remote channel has requested a BDW cycle?
11. What determines whether or not the tape unit triggers may be set?
12. Explain why the channel priority trigger is turned on during E time of an RCH or LCH.

DATA CHANNEL

QUIZ 2

Which item(s) in Col. B will allow the item in Col. A to take place?

A	B
1. Reset R/W Reg.	1. Write Demand
2. Tape Demand	2. BDW Req'd
3. Write Pulse	3. WD 320
4. Step tape group counter	4. Non Data Disc
5. Chan Input SW to WC	5. DR not loaded
6. Set R/W Reg.	6. WC 1
7. Non-Data Disc.	7. WD 50
8. BDW Req'd	8. WC 5
9. CAC to CAS	9. RL-LC Gate
10. Reset Chan Unit Tgrs.	10. WC 3
11. DR to TR	11. BDW tgr. on
12. Set Class and Unit	12. Write 1st word test
	13. WC 9
	14. Demand Gate Tgr.
	15. WC 14
	16. End Op Cntl. Tgr.
	17. T Selected
	18. RL mode

Example:

A	B
1. Go to I	1. 19
2. Step P. C.	2. I 7
3. SB to SR	3. End Op Tgr.
	4. E 7

Answers:

1. 3
2. 1
3. 2, 4

DATA CHANNEL

QUIZ 3

1. On a WRS operation, what line simulates the 1st demand to bring the 1st word to be written from DR to TR?
2. What is the purpose of having the line "busy" initiate a non data disconnect?
3. In what mode or modes is the delay counter run during a WRS operation?
4. True or False. Write Condition Tgr. in TAU controls the write clock on a WRS operation.
5. What functions does the write tgr. release tgr. control?
6. If several channels all request a BDW cycle at the same time, what determines which channel will use the 1st B cycle?
7. True or False. The BCW or BDW tgr. must be on to initiate a B cycle.
8. Name all the errors checked on a WRS operation.
9. After execution of the TCH at ALPHA, what will be in the Op. Reg., Word Counter, Channel Address Counter, and Location Counter?

WRS A 1	ALPHA	TCH	200, 4
RCHA ALPHA	177	IOCD	300, 4, 10
HLT	200	IOCP	250, 0, 25
	247	IOCP	320, 4, 50
	250	IOCD	1000, , 200
	251	IORP	252, , 300
	252	IOCD	0, 0, 0

10. What is the maximum number of records that may be read with an IORP?
11. Can an IOSP be used to read more than 1 record? Why?
12. What signals the system that it is trying to execute an IA command?
13. List three ways that B cycle demands may be satisfied by CPU.
14. Why do we need 4 oscillators for driving the write clock?
15. What registers are stored on a SCH operation?

16. A tape WRS is given, followed by an RCH with IOCD WC = 1. The write operation appeared to perform normally except for a tape redundancy indication. Further investigation disclosed that no Check Character had been written. This caused the LRCR error during the read checking operation. A source of trouble could be:
- a. G output of +A at 5B shorted to -P 61.30.20.1
 - b. G output of +A at 4E is shorted to -P 61.30.50.1
 - c. D input of Convert block at 2D shorted to -P 61.60.32.1
 - d. D input to -TO at 3C shorted to +N 61.60.32.1
 - e. None of the above

17. A Tape Write Select instruction is given followed by a reset and load channel with an IOCD word count of 2. Tape starts moving, never stops and nothing has been written on tape. A source of trouble could be:
- a. A output of Driver Terminator at 5B is shorted to plus level. 60.36.03.1
 - b. G output of Convert block at 1C is shorted to minus level. 60.36.03.1
 - c. G output of Convert block at 1C is shorted to plus level. 60.36.03.1
 - d. G output of -AND block located at 5H is shorted to plus level. 60.36.04.1
 - e. H output of +OR block at 36 is shorted to minus level 60.36.01.1

18. A Tape Write Select instruction is given, followed by reset and load channel with IOCD word count of 2. Core storage contains the following octal representation of the two words.

1st Word	21	02	27	40	33	04
2nd Word	13	41	74	62	01	16

The operation is performed without any apparent problems. A later comparison shows the words actually written were:

1st Word	21	23	27	61	33	25
2nd Word	13	53	77	73	13	17

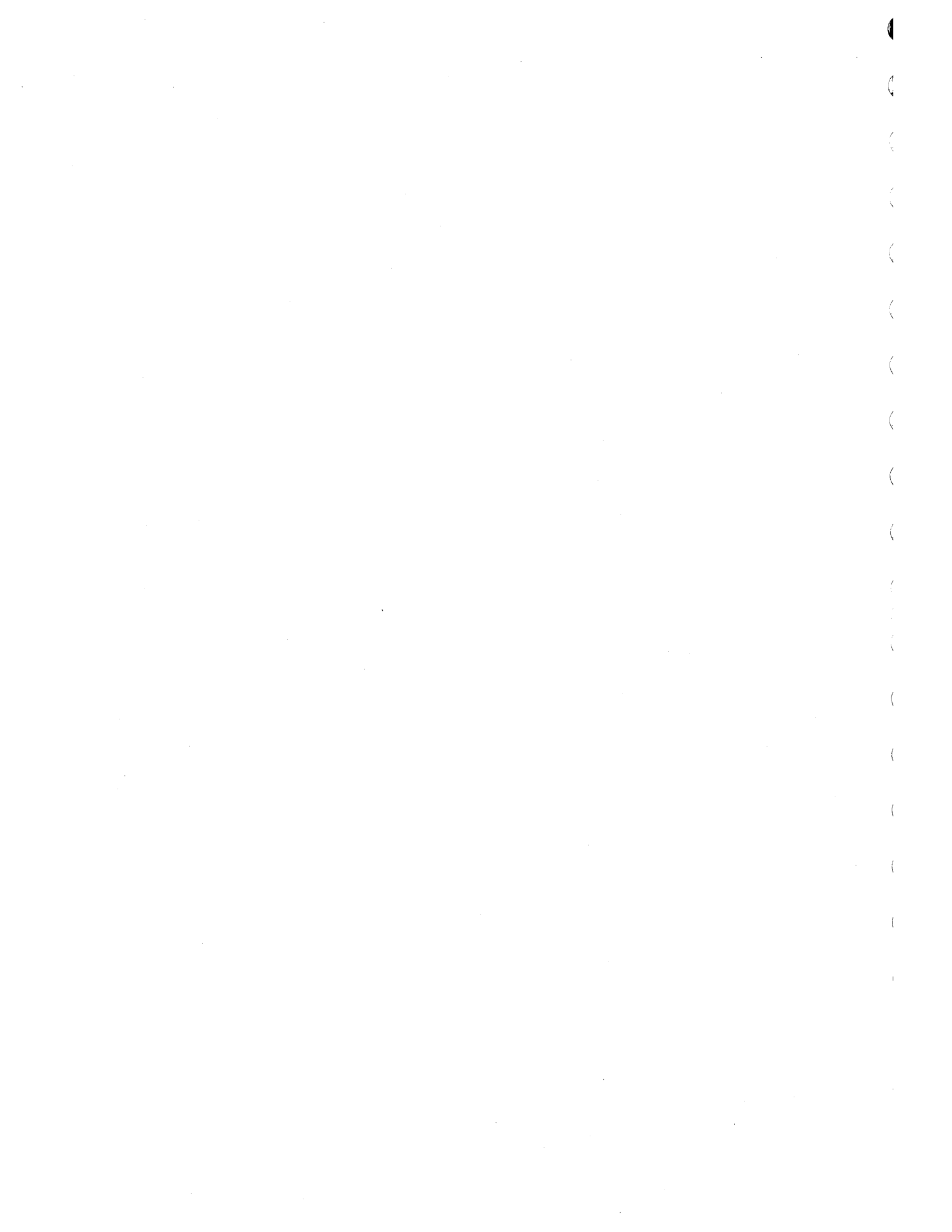
A source of trouble could be:

- a. HG output of Delay block located at 2F is shorted to -N. 60.25.13.1
- b. H output of Driver Emitter block at 1F is shorted to -N. 60.25.13.1
- c. B output of Convert block at 1E is shorted to -P. 60.25.13.1
- d. A output of Driver Emitter located at 3A is shorted to +N. 60.25.14.1
- e. G output of -AND circuit at 4A is shorted to +N. 61.20.30.1

DATA CHANNEL

QUIZ 4

1. What prevents an error indication from words that have been read after an IOCD WC less than record length brings up "set data disc gate"?
2. What is the purpose of the sync trigger on an RDS operation?
3. What is the purpose of the "1st word test" trigger? (-Tgr. at 5F and 5G 60.36.02.2)
4. What prevents the TU select triggers from being reset when reading over an EOR gap?
5. When reading 4 words from a 100 word record with an IOCD WC = 4, when will the 4th word be moved from TR to DR? During which character of which word will the demand be generated that will initiate a disconnect?
6. When does "busy" fall on an IOCD WC less than record length?
7. Give two functions of the 1st Char TM trigger. (60.36.07.1)
8. What moves the last word from TR to DR when WC equals record length?
9. When reading in the binary mode, a 2 bit and an A bit were read into skew A. Which, if any, of the exclusive OR circuits on 60.51.10.1 will be conditioned?
10. If a tape were started at load point in an RDS operation and two record gaps were read over, would one expect to find the load point trigger on or off while in the 2nd record? Why?



DATA CHANNEL

QUIZ 5

1. During the first BCW cycle of a TCH Command, the routing of Channel Address switches to Memory address register is suppressed. (T or F)
2. An indirectly addressed TCH Command will take _____ BCW cycles.
3. During the IA BCW cycle of an indirectly addressed command, the channel input switches are gated to _____.
4. Why is the TCH command sent to DC, if all the decoding is done in the MX?
5. On a rewind operation, Read Status is set in the tape drive at D64 provided it was in Write Status prior to REW instruction & the Go tgr. is set at D160 through AND circuit 4D of systems page 61.60.10.1. True or False?
6. On a Backspace Operation, we start the RDD delay counter. If the counter gets to RDD 36 it will turn on the check character trigger and start a normal end-of-record operation. True or False?
7. On a backspace record operation, the Go tgr. is always reset at RDD 26. True or False?
8. On a Backspace Record operation the delay counter is run in _____ mode. An _____ indicates that tape has moved into an EOR gap.
9. What use is the CC of two on a backspace file operation?
10. Power is just brought up to the system. What density status (high or low) are M4 tape drives in?
11. In a one sentence explanation, what is accomplished by the instruction +0776....3205?
12. CPU cannot END OP on a REWIND instruction. Which of the following could cause this?
 - a. E input pin of -A at 5G of 60.36.02.1 shorted to -P.
 - b. G output pin of C at 3H of 60.60.03.1 is shorted to +P.
 - c. G output pin of +A at 4C of 60.50.01.1 is shorted to -P.
 - d. G output pin of C at 4B on 60.50.03.1 is shorted to -N.

DATA CHANNEL AND TAPE I/O
COMPREHENSIVE QUIZ

General (2Points Each)

1. The channel interlock prevents all Data Select instructions and all non-data select instructions from ending operation. T or F?
2. The SCH and BCW operations cause the Location Counter to be gated through the Channel Address Switches T or F?
3. The TAU "Busy" line causes "T Selected" to initiate a "non-data disc". This provides for stacking a non-data select instruction or accepting a subsequent non-data select into channel. T or F?
4. A RCH instruction will only hang up on "L Time" if the "Channel Interlock" line is up. T or F?
5. The BCW or BDW Tgr. must be on to initiate a B cycle. T or F?

WRS Operation (5 Points Each)

6. With the following program, how many tape demands will occur before disconnect?

WRS 1201	100	IOCP WC = 3
RCHA 100	101	IORP WC = 5
HTR	102	IORT WC = 6
	103	IOCD WC = 5

7. "Write Demand" (60.36.06.1, 1C) gates the first word from the DR to the TR. T or F?
8. When WRS is directly followed by RCHA (IOCD 100, , 6) the WC is 5 and the CAC is 101 when "Write Cond" is established. T or F?
9. The first "Write Demand" following WC = 0 develops a "Disc Call".
10. A Tape WRS is given, followed by a RCH with IOCD WC = 1. The write operation appeared to perform normally except for a tape redundancy indication. Further investigation disclosed that no Check Character had been written. This caused the LRCR error during the read checking operation. A source of trouble could be:
 - a. G output of +A at 5B is shorted to -P 61.30.20.1
 - b. G output of +A at 4E is shorted to -P 61.30.50.1
 - c. D input of Convert block at 2D shorted to +P 61.60.32.1
 - d. D input to -TO at 3C shorted to +N 61.60.32.1

11. A Tape Write Select instruction is given, followed by reset and load channel with IOCD word count of 2. Core storage contains the following octal representation of the two words:

1st Word 21 02 27 40 33 04

2nd Word 13 41 74 62 01 16

The operation is performed without any apparent problems. A later comparison shows the words actually written were:

1st Word 21 23 27 61 33 25

2nd Word 13 53 77 73 13 17

A source of trouble could be:

- a. HG output of Delay block located at 2F is shorted to -N 60.25.13.1
- b. H output of Driver Emitter block at 1F is shorted to -N. 60.25.13.1
- c. B output of Convert block at 1E is shorted to -P. 60.25.13.1
- d. A output of Driver Emitter located at 3A is shorted to +N. 60.25.14.1
- e. G output of -AND circuit at 4A is shorted to +N. 61.20.30.1

RDS Operations (2 Points each)

- 12. Skew Reg. A VRC turns on the TAU Error trigger. T or F?
- 13. When reading successive records, between each record a Read Delay of 30 is executed. (T or F)
- 14. The check character is VRC checked in the R/W register. T or F?
- 15. A MII tape drive, low density operation causes the Read clock to be driven by a 240KC oscillator. T or F?
- 16. The Read Translators are used to change a 7090 Internal BCD character to its external BCD equivalent. T or F?

RDS Operations (5 points each)

- 17. A tape demand with WC not at zero and "DR loaded tgr" on will cause an I/O check. T or F?
- 18. a. When reading 4 words from a 100 word record with an IOCD WC = 4, when will the 4th word be moved from TR to DR?
b. During which character of which word will the demand be generated that will initiate a disconnect.

19. a. When does "Data Disc" occur when reading a 100 word record with an IOCD WC = 17?
b. When does "Data Disc" occur when reading a 17 word record with an IOCD WC = 17?
20. In the following sequence, label each statement True or False. Assume that we are performing a Read Only operation and we have just started the last character in Skew Reg. A.
- a. 1st bit starts the Read Clock
b. RC 7 sets the RDD tgr. and starts delay ctr. running
c. RDD 36 samples the R/W reg. VR checker for an error in the check character VR.
d. RDD 50 will turn on the check character trigger.
e. RDD 128 resets read condition

Non-Data Select Operations (5 points each)

21. On which instructions will the CPU hang up in "L time" during the following program:

0	W RSA	1	X	IOCD WC = 50
1	RCHA	X	Y	IOCD WC = 0
2	TCOA	*		
3	REWA	1		
4	REWA	1		
5	BSRA	1		
6	RDSA	1		
7	RCHA	Y		
10	WEFA	1		
11	REWA	1		
12	RUNA	1		
13	HPR			

22. On a backspace file operation the Read Cond. tgr. is not reset between records. True or False?
23. On a backspace operation, we start the RDD delay counter. If the counter gets to RDD 36 it will turn on the check character trigger and start a normal end-of-record operation. True or False

Data Channel Trap Operations (2 Points Each)

24. A separate instruction must be given for each channel that is to be enabled. True or False?

25. If an ENB instruction is given to enable a channel for CWT traps sometime after an EOF has been read, and before a TEF is executed, the channel will perform the trap.
26. If channel A were enabled to trap on CWT and an EOF condition arose, the CWT enable trigger would be reset when the trap occurs. True or False?
27. 2 Points Each Part

12 All zeros	SUB	RCT
13 TRA SUB	REWA	1
14 HTR	HTR	
:		
:		
:		
:		
100 CLA A		
101 ADD All		
102 TCOA 102		
103 HTR		

Channel A is enabled to trap on TCT. A redundancy error occurs and channel asks for a trap during the E cycle of CLA. If the out-of-phase output of the +TO at 2E on 2.10.56.1 were shorted to a +N level:

- a. $C(12_8)$ at the halt is?
- b. $C(PC)$ at the halt is?

Manual Operations (5 Points)

28. The customer could expect normal system operation if Channel B had the CSRO switch on as long as the channel is in automatic. True or False?

7090 COMPREHENSIVE QUIZ

1. Explain the rules of addition as applicable to the 7090 (for signs alike and signs unlike).
2. On an add magnitude instruction:
 - a. Complement addition cannot occur
 - b. The sign of the number brought from storage is inverted
 - c. The added numbers are treated as logical numbers
 - d. The sum is a logical, not arithmetic, sum
 - e. None of the above
 - f. All of the above
3. MPR checks position _____ of the _____ to determine whether or not to adjust the _____.
4. In both TIX and TNX the contents of the index register specified is reduced by the decrement whenever the contents of the index register is greater than the decrement. True or False?
5.

GO	CLA	A
	XEC	GO+5
	HTR	
	ALS	5
	HPR	
	LBT	
	ADD	A
	HTR	GO
A	OCT	15

Where will the above program halt?
6. The maximum number of shifts that can be accomplished by any one instruction is 35, decimal. True or False?
7. 172.4 octal, is the floating point representation of $.4 \times 2^{-6}$. True or False?
8. The "OR and "AND" instructions in the 7090 always treat the factors as logical words. True or False?
9. What channel registers are stored by a store channel instruction?
10. A data channel's operation register decoded output determines whether the data channel will read or write information. True or False?

11. A reset and load instruction is necessary following a read select instruction to start the I/O unit in motion. True or False?
12. There is no difference in the operation code for a WRS for two different data channels. True or False?
13. How can the beginning of tape trigger be turned on?
14. The location counter contains the address field (21 - 35) plus 1 of a reset and load channel instruction after the reset and load channel instruction has been executed. True or False?
15. Given the following instructions and commands, what will be written on tape? All word counts are decimal.

WRS	1221	0100	IOSP	200, , 20
RCHA	100	0101	IORP	250, , 15
LCHA	150	0102	IOCP	300, , 10
HPR		0103	IOST	350, , 20
		0104	IOCD	0, , 0
		0150	IOSP	400, , 10
		0151	IOCD	450, , 5

16. When the reference transistor of a plus OR circuit is in conduction, the out-of-phase output is: (+, -)
17. If all inputs to a plus AND circuit are at a minus level, the reference transistor is in conduction. True or False?
18. A negative binary trigger is reset off by a (+, -) reset pulse. The next (+, -) binary input will turn the trigger on. This on condition will be recognized by a (+, -) in-phase output.
19. Slide connectors leaving tailgate F are routed to the A row of panels 3 and 4 in gates A-D. True or False?
20. An undesirable affect that may result from not following the wire routing spelled out on the automated add list is _____.
21. The bellows switch is used to indicate tape in both columns. True or False?
22. When the nylon ring is in position on the file reel, information may not be written on tape. True or False?
23. Too small a drive gap on the left prolay can cause glitching when performing a backward start-stop operation. True or False?

24. If the forward start and forward stop are adjusted correctly (end of record gaps within tolerance), which prolay gap requires adjusting to correct a backward creep failure indicated by the 9T55 printout?
25. In what status is each of the prolays on a 729 II or IV during a backward go condition?
26. There are three diaphragm switches in each column of a 729 II or IV. Give the function of each switch.
27. After a load-rewind operation, the tape is sitting at load point in a read status and the prolays energized left stop and right neutral. True or False?
28. What is means by the term "count of five"?
29. Where should the check be made for 8.8V pp output from the read pre-amps? In what density should this check be made.
30. When adjusting electrical skew should the write skew or the read skew be adjusted first?
31. Define:
 - a. ECW
 - b. BCW
 - c. BDW
32. In data channel, the _____ and the _____ are count up counters. The _____ is a step down counter. The _____ and _____ are stepped every BDW cycle while the _____ is stepped every BCW cycle.
33. The information coming out of core during an E cycle appears at the input to all channel input switches. True or False?
34. When writing tape, BCD has an even vertical redundancy and binary an odd vertical redundancy. True or False?
35. The only consideration that defines how many characters per inch is the model of tape drive. True or False?
36. The _____ in data channel must be turned on before a select operation can end operation in CPU.
37. What is the primary purpose of decoding TCH in the MULTX look-ahead circuits?
38. Name the error conditions possible on a WRS tape operation.
39. What determines whether or not the tape unit select triggers may be set?

40. Why does busy reset the channel unit select triggers?
41. What is the function or functions of the tape group counter?
42. What is the purpose of the sync trigger?
43. On a WRS operation, what line simulates the 1st demand to bring the 1st word to be written from DR to TR?
44. The decision to write a "C" bit is accomplished by sampling the output of the R/W Register VRC. True or False?
45. What prevents the data register from being loaded with a command word on a ECW or BCW cycle?
46. Space tape may be achieved by which of the following procedures?
 - a. Write select followed by IOCD, or IORP, or IOST, word count equal to zero.
 - b. Write select followed by IOCP, or IORP, or IOSP, word count equal to zero.
 - c. Write select followed by IOCP, or IOCD, word count equal to zero.
 - d. Write select followed by IOCD, or IORP word count equal to zero.
47. With the following program, how many tape demands will occur prior to the demand that initiates the disconnect?

WRS	1201	100	IOCP	WC = 10
RCHA		101	IORP	WC = 10
HPR	100	102	IOCD	WC = 5

48. What four operations gate the location counter switches to the location counter.
49. What operation(s) cause the location counter to be gated through the Channel address switches?
50. On a Backspace Record operation the delay counter is run in _____ mode. If the counter gets to _____ the system is signalled that a record has been backspaced.
51. What does a character count of two indicate during a backspace file operation?
52. What conditions may cause a data channel trap?
53. Reading an end-of-file mark will always cause a data channel trap. True or False?

54. 100 ADD
 101 STO
 102 RDS
 103 RCHA 1221
 104 LDQ
 105 MPY
 106 HTR

Channel A has been enabled for all trap operations and has been in a read operation prior to the decoding of the ADD instruction. If a trap demand occurred during the L cycle of the RDS operation, what would be stored in the address field of location 12 when the trap is executed?

55. Name all registers in the machine made up of shift cells.
56. Decoding a halt and transfer or a halt and proceed always turns off the automatic light. True or False?
57. Clock trigger zero turns off when clock trigger _____ turns on.
58. The cyclic makeup of an indirectly addressed CLA will be _____.
59. All information going to or coming from storage must pass through the multiplexor. True or False?
60. A loss of either the odd or even clock drive pulses would result in a clock running at half its normal speed. True or False?
61. Positions 1 and 2 of an operation code are checked in the storage register to determine where to position the operation code in the program register. True or False?
62. Which instructions made the I3D1 gating of the CPU address register to the program counter necessary?
63. The end operation trigger being on signifies that the next cycle will be an _____ cycle unless a _____ cycle is allowed.
64. If the following program were executed what would be the cyclic makeup? Also give the address that would be in the AR at 11 time of each cycle.

0000	TRA	0002		<u>Example of Answer</u>
0002	TNZ	0004	(ACC = +5	I 1000
0003	TPL	0004		L 2000
0004	AXC	0005		E 3000
0005	STO	0100		I 5000
0006	HTR	0010		E 4000

65. After the execution of a PAC instruction the specified index register will contain the _____ form of the address portion of the Accumulator.
66. The instructor following a STR will be taken from location _____.
67. The following instruction is executed in the 7090. What will be the result of the execution of this instruction?

<u>Instr</u>	<u>Tag</u>	<u>Address</u>
CLA	7	0006
XRA	75000	
XRB	02730	
XRC	00041	

68. How many L cycles will be needed to accomplish an ARS with 15 decimal shifts?
69. What, if anything, is in the shift counter at I 11 of an AXC instruction?
70. All transfers encountered while in the transfer trap mode will cause the next instruction to come from location one. True or False?
71. Why is it necessary to block PC→AS at I 10 time of an Execute instruction?
72. The "On Test for Indicators" instruction will skip the next instruction only if all 36 indicators are on. True or False?
73. To realize any increased speed during MPY, the multiplier must contain at least two successive zeroes. True or False?
74. If AC35 could not be set to a one, the MQ would contain all zeroes after the completion of MPY. True or False?
75. If a Q carry is noted during the divide test of DVH, the machine will halt. True or False.
76. Put the octal number 32.743 into floating point format as it would appear in the AC in normalized form.
77. During FMP second step "L" time, why does the instruction only attempt to normalize one place?
78. The original contents of the MQ is lost during the execution of a CAQ instruction. True or False?

NOTE: All core storage questions pertain to oil cooled memory.

79. A core storage cycle is initiated by _____.
80. During rewrite time, the decision to write "1's" or "0's" into storage is determined by:
- a. Z driver timing
 - b. MDR outputs
 - c. DOG and MDR outputs
 - d. MAR 4 and 5 outputs
81. What are the differences between the read portion of a Read out cycle and a store cycle?
82. Selection of a particular Y matrix switch core within any one Y segment is accomplished by:
- a. The X segment gate
 - b. The Y segment gate
 - c. Coincidence of the X and Y segment gates
 - d. Physical winding of the matrix switch cores
 - e. MAR 11 through 17 outputs.
83. Why is it necessary to have 32 primary windings feeding each block of 16 matrix switches?
84. How many MDBO lines are there from the 7302 to the MX?
85. What is the purpose of the bias winding through the G switch?
86. Timing pulses for the core storage unit are generated by the MX clock? True or False?
87. What positions of MAR determine X segment selection and X matrix switch selection?
88. If the timing error trigger is left on at the end of a core cycle, memory cannot be selected. True or False.
89. During the execution of an STD 55630, the store decrement lines prevent strobing of planes 3 - 17. True or False?
90. If a CLA 500 were executed and the R/W trigger in the 7302 could not be turned on, location 500 would contain all zeroes at the end of the E cycle regardless of what was previously in loc. 500. True or False.
91. The load cards key, upon depression automatically _____ the reader, forces an _____ command with a word count of _____ into channel and takes the next command from _____.

92. Locations 00000-00027 contain a card image. If words 00000, 00001, and 00027 contain all ones, what would be printed on the 716 using a write printer operation and an IOCD with a word count of 30g?
93. Reselection of card machines for continuous feed cycles is done by:
 - a. After 8 CB pulse
 - b. End of record pulse
 - c. Reset of record control trigger
 - d. Reset of CB counter
94. What is the function of the Card Ring?
95. What is the function of the CB counter?
96. Upon depression of the Normal Off Button, all modular supplies are dropped simultaneously. True or False?
97. Dropping power on a Data Channel drops power to the Tape Drives and DC bus of the card machines. True or False?
98. How much of the system loses power when a fuse blows in CPU #1 modular supply CD?
99. What two voltages are available in each module (i. e., CPU1, CPU2, MX, etc.) following a normal off sequence.
100. The following diagnostic routine picked up a machine failure during the execution of the instruction MPY.

02624	444770606060		BCD	1MPY	MPY TEST
02625	0560 00 0 06674	B47	LDQ	PONES	MULTIPLY ALL ONES
02626	0200 00 0 06674		MPY	PONES	BY ALL ONES
02627	0400 00 0 06701		ADD	ONE	ADD ONE, 2-35 HAVE ONES
02630	0140 00 0 02631		TOV	B47+4	SET OVFL
02631	0400 00 0 06701		ADD	ONE	RIPPLE CARRY FOR ACC
02632	0140 00 0 02635		TOV	B47+8	OK, DID RIPPLE
02633	0074 00 4 07041		TSX	ERROR-1, 4	
02634	0020 00 0 02625		TRA	B47	

Fill in the blanks for test location and error location in the following diagnostic printout that occurred as a result of the failure discovered by the above routine.

TEST LOC _____, OPN MPY _____, ERROR LOC _____, 0 LOC 002000000034,
 SW 000001 LITE 0000, MQ 377740760401, XRA 07777, XRB 07777, XRC 75145, TRAP TGR 0
 ACC +, Q 0, P 0, 377777777777, DIV CK 0, ACC OVFL 0,
 INDS 000000000000, KEYS 002000003033

R23-2926



International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601

7090 Student Guide Printed in U.S.A. R23-2926