File Number 1130-36
Form C26-3717-4

IBM

Systems Reference Library

IBM 1130 Disk Monitor System, Version 2, Programming and Operator's Guide

Program Numbers 1130-OS-005
1130-OS-006

This manual contains the operating and maintenance procedures for the IBM 1130 Disk Monitor System, Version 2. An introductory section acquaints the user with the IBM 1130 System. A section on programming tips and techniques assists the user in utilizing the Monitor system.

Monitor system control records are described in detail. An appendix contains all error messages generated by the system.

PREFACE

This publication provides the IBM 1130 System user with the information required to operate and maintain the IBM 1130 Disk Monitor programming system. It is recommended that the user familiarize himself with the terms contained in the Glossary at the back of this manual. It is important that these terms be understood in the context of the Monitor system.

All hexadecimal addresses in this manual are shown in the form /XXXX.

Symbolic addresses rather than absolute addresses are used throughout this manual. Certain constants are also denoted symbolically. A table of equivalences is provided in Appendix H, Resident Monitor.

$XXXX	All symbolic labels whose first character is a dollar sign are found in the Resident Communications Area (COMMA)
#XXXX	All symbolic labels whose first character is a pound sign are found in the Disk Communications Area (DCOM).
@XXXX	All symbolic labels whose first character is a commercial at sign are considered to have absolute values, i.e., @HDNG refers to the page heading sector (sector 7) and thus has a value of 7.

NOTE: The number sign (#) and commercial at sign @ are not included in the 1403 Printer or 1132 Printer character set; therefore, an equal sign (=) replaces the # and an apostrophe (') replaces the @ in the printer listings.

MINIMUM SYSTEM CONFIGURATION

The minimum system configuration required to operate the 1130 Disk Monitor system is as follows:

IBM 1131 Central Processing Unit, Model 2, with 4096 words of core storage, and one of the following input/output devices

IBM 1442 Card Read Punch, Model 6 or 7

IBM 2501 Card Reader, in combination with an IBM 1442 Card Punch, Model 5, or an IBM 1442 Card Read Punch, Model 6 or 7

IBM 1134 Paper Tape Reader in combination with an IBM 1055 Paper Tape Punch.

PUBLICATIONS

The following publications will assist the user in utilizing the Monitor system.

IBM 1130 Functional Characteristics (Form A26-5881)

IBM 1130 Computing System Input/Output Units (Form A26-5890)

IBM 1130 Assembler Language (Form C26-5927)

IBM 1130/1800 Basic FORTRAN IV Language (Form C26-3715)

IBM 1130 Subroutine Library (Form C26-5929)

IBM System/360 Operating System and 1130 Disk Monitor System: System/360 1130 Data Transmission for FORTRAN (Form C27-6937)

IBM System/360 Operating System and 1130 Disk Monitor System: User's Guide for Job Control From an IBM 2250 Display Unit Attached to an IBM 1130 System (Form C27-6938)

Fourth Edition (Feb 1969)

This is a major revision of and makes obsolete C26-3717-3 Changes to the text and small changes to illustrations are indicated by a vertical line to the left of the change; changed or added illustrations are denoted by the symbol o to the left of the caption.

This edition applies to version 2, modification 5 of IBM 1130 Disk Monitor Programming System, to Version 1, modification 1, IBM 1130 Remote Job Entry Work Station Program, and to all subsequent versions and modification until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the specifications herein; any such changes will be reported in subsequent revisions or Technical Newsletters. Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Nordic Laboratory, Technical Communications, Box 962, 181 09 Lidingö, Sweden.

The 1130 Disk Monitor System provides for the continuous operation of the 1130 Computing System, with minimal set-up time and operator intervention, in a stacked job environment. The Monitor system consists of eight distinct but interdependent elements -- Supervisor, Disk Utility Program, Assembler, FORTRAN Compiler, RPG compiler, Core Load Builder, Core Image Loader, and System Library.

The Supervisor performs control functions for the Monitor system and provides the linkage between user programs and Monitor programs.

The Disk Utility Program (DUP) is a group of IBM-supplied programs that performs operations involving the disk such as storing, moving, deleting, and dumping data and/or programs.

The Assembler converts source programs written in Assembler language into machine-language object programs.

The FORTRAN Compiler translates source programs written in 1130 Basic FORTRAN IV language into machine-language object programs.

The RPG Compiler translates program written in 1130 RPG language into machine-language programs.

The Core Load Builder constructs core image programs from mainline object programs. The mainline programs and all necessary subprograms are converted into Disk Core Image format from Disk System format, and the resultant core load is built for immediate execution or for storing for future execution.

The Core Image Loader serves as both a loader for core loads and as an interface for the Monitor programs.

The System Library is a group of disk-resident programs that perform I/O, data conversion, arithmetic, disk initialization, and maintenance functions.

The operating procedures for readying the system I/O units are described below. Following these procedures are instructions to the operator on the various ways of actually getting data in and out of the system and how these methods are utilized by the 1130 Disk Monitor Programming System.

READYING THE IBM 1130 COMPUTING SYSTEM

This section describes the basic operator actions required to ready the IBM 1130 Computing System for operation. The paragraphs on readying the I/O units should be sufficient to allow the operator to prepare the units for selection by the system. Where necessary, illustrations have been provided to supplement the text.

Additional information regarding 1130 system and unit displays and operator functions can be found in the following publications.

IBM 1130 Functional Characteristics (Form A26-5881)
IBM 1130 Input/Output Units (Form A26-5890)
IBM 2501 Card Reader, Models A1 and A2 - Component Description and Operating Procedures (Form A26-5892)
IBM Disk Pack Handling and Operator Procedures (Form A26-5756)

1131 Central Processing Unit

Most operator action will occur at the console of the 1130 system. This console, as well as three I/O devices -- the Keyboard/Console Printer, the console entry switches, and a single disk storage drive -- are all located in or on the 1131 CPU.

System Power On. When the 1131 POWER switch is turned on, the following console operator panel lights will be on: DISK UNLOCK (no cartridge in single disk storage drive) and FORMS CHECK (if there is no paper in the Console Printer). If any other operator panel lights are on, press the RESET key.

To ready the Console Printer, perform the following steps:

1. Open the Console Printer top cover.
2. Pull the paper pressure rod forward (the rod with three rubber rollers that leans against the platen). If the paper is to be pin fed, this rod should remain in this position.
3. Lift up on the left and right platen pin feed pressure plates.
4. Set the paper release lever in the forward position. This lever is located on the top right rear corner of the Console Printer. If the paper is to be pin fed, this lever should remain in this position.
5. Feed the paper in from the rear and guide it under the platen. Make sure that the paper lies over and closes the forms check microswitch. This will turn off the FORMS CHECK light on the console operator panel.
6. Lay the paper back across the top of the Console Printer and guide the paper so that the holes line up with the pin feeds.
7. Close the pin feed pressure plates.
8. Looking directly down into the Console Printer, set the left and right margins. The margin settings can be read on the scale across the front of the unit.
9. Close the top cover.
10. Press CARRIER RETURN.

The Console Printer is now ready to be selected.

To ready the single disk storage drive, perform the following steps.

1. Open the single disk storage access cover located at the front of the 1131 (to the right of the console). The cover swings open to the right.
2. Grasp the handle of the access release mechanism and pull out and down.
3. Pick up the cartridge and, holding the cartridge with the IBM name towards you and on the left, insert the cartridge into the slot.
4. When the cartridge is seated, raise the access release handle to lock the cartridge into place.
5. Turn the DISK switch (leftmost switch on the panel beneath the cartridge enclosure) to the ON position. As the disk starts to turn, the DISK UNLOCK light on the console operator panel will go out.
6. Close the access cover.

When the drive comes up to speed (approximately 90 seconds), the DISK READY indicator on the console operator panel will turn on. The single disk storage drive is now ready to be selected.

1442 Model 6 and 7 Card Read Punch Ready Procedure

Pre-conditions. POWER ON light on, CHECK light off, CHIP BOX light off, stacker not full, and covers closed.

When the system is first powered up, it is good practice to press the NPRO key to ensure that no cards are in the feed path.

Readying the Card Read Punch. When all pre-conditions are met, place the cards to be processed in the hopper, face down, 9-edge first, and press reader START. When the first card is positioned at the read station, the READY light will turn on. The card read punch is now ready to be selected.

1442 Model 5 Card Punch Ready Procedure

Pre-conditions. POWER ON light on, CHECK light off, and ATTENTION light off.

When the system is first powered up, the HOPPER check light is lit. Press NPRO to turn this light off. This action ensures the card path is clear.

Readying the Card Punch. When all pre-conditions are met, place blank cards in the hopper, face down, 9-edge first, and press punch START. Two card feed ycles are taken and the first card is registered at he punch station. When the punch READY light turns on, the card punch is ready to be selected.

2501 Card Reader Ready Procedure

Pre-conditions. POWER ON light on, READ CHECK light off, FEED CHECK light off, and ATTENTION light off.

When the system is first powered up, the FEED CHECK light is lit. Press NPRO to turn this light off. This action ensures that the card path is clear.

Readying the Card Reader. When all pre-conditions are met, place the cards to be processed in the hopper, face down, 9-edge first, and press reader START. When the first card is positioned at the pre-read station, the READY light will turn on. The card reader is now ready to be selected.

1134 Paper Tape Reader Ready Procedure

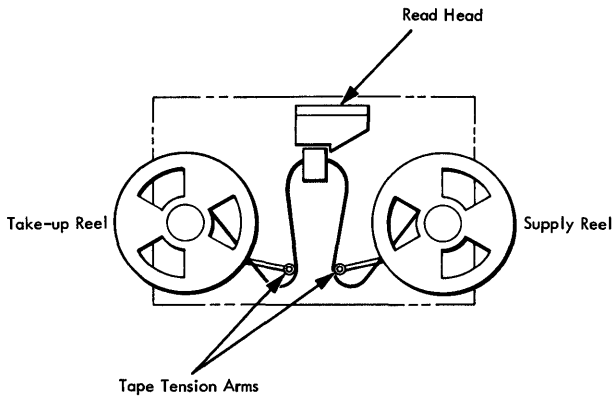
Pre-conditions. System power on.

Readying the Paper Tape Reader. Raise the lever located at the top right side of the read head (see illustration below). Load the reel containing the program tape on the right hand drive and lock the reel in place. The tape must be loaded so that the three-hole side is nearest the operator. With both tension arms in the up position, feed the tape across the read head and position the tape on the drive sprocket.

Position a program tape so that a delete code (all punches) beyond the program ID punched in the leader is under the read starwheels.

Position a tape without a leader (or when starting in the middle of a tape) so that the first character position to be read is one position to the right of the read starwheels.

Lower the lever on the read head, thus bringing the read starwheels in contact with the tape. Wind the leader on the take-up reel and let down the tape tension arms.



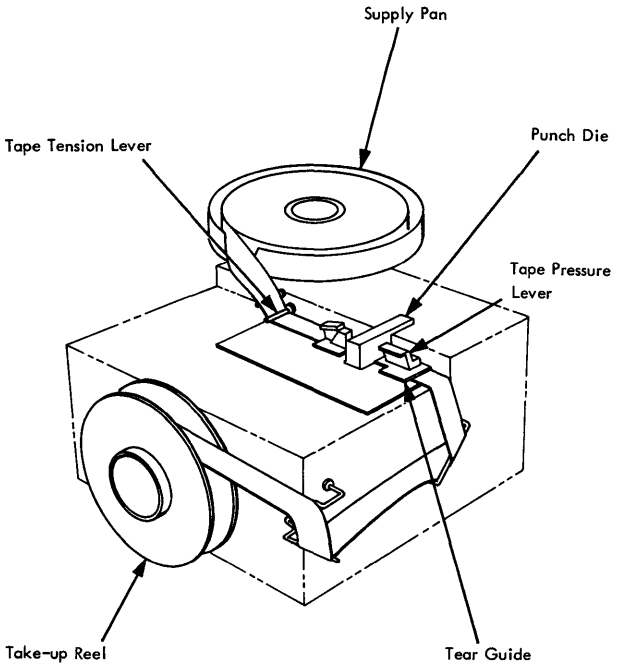
The paper tape reader is now ready to be selected.

1055 Paper Tape Punch Ready Procedure

Pre-conditions. System power on.

Readying the Paper Tape Punch.

1. Place a reel of tape in the supply pan so that the tape feeds out toward the punch die (see illustration below).
2. With the punch die facing forward (unit name plate at the front), pivot the tape pressure lever (right side of die) up and to the right.
3. Feed the tape from the supply pan over the first tape guide, under the tape tension lever, and slide the tape in under the punch die, tear guide, and tape pressure lever.
4. If the punch has a take-up reel, guide the tape over the side of the unit, over the outside of the side guide, and back up towards the front of the unit.
5. The tape now makes a half turn towards the outside and comes up and over the end guide.
6. The tape is then brought up and over to the left and wound over the top of the take-up reel.



After the tape is loaded, a leader (all delete codes) may be made by first pressing and holding the DELETE key. Now press the FEED key and hold until a leader of sufficient length has been punched. Release the FEED key before releasing the DELETE key. The paper tape punch is now ready to be selected.

1132 Printer Ready Procedure

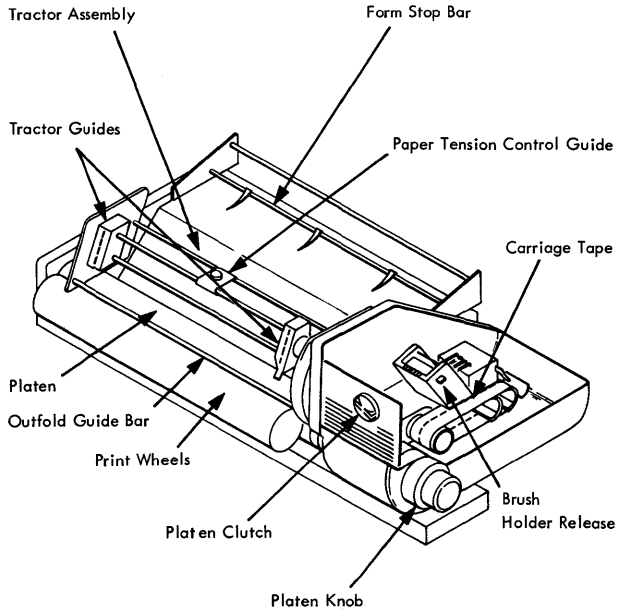
Pre-conditions. POWER ON light on and MOTOR switch on. The FORMS CHECK light will be on if there are no forms in the printer.

Readying the 1132 printer. To load the forms into the printer:

1. Raise the top cover and disengage the PLATEN CLUTCH (set it to OUT). This knob is located on the right side of the print carriage (see illustration below).
2. Turn the outside knob on the right end of the carriage to ensure that the carriage is free.
3. Remove the spring loaded outfold guide bar (the bar across the bottom front of the forms tractor, directly behind the print wheels).
4. Open the left and right tractor pressure plates.
5. Now feed the forms from the rear of the printer under the form stop bar (three levers) and down under the tractor.
6. Use a rocking motion to feed the paper under the platen (if necessary raise the paper tension control guide located in the center of the tractor).
7. When the paper appears in front of the platen, grasp it firmly and pull it up so that it lies evenly across the tractor.
8. Place the holes in the paper on the left and right tractor pins and close the tractor pressure plates.
9. Reinsert the outfold guide bar.
10. Using the knob at the right end of the carriage, feed the paper until a crease between two sheets appears just above the print wheels.

To load the carriage control tape into the printer:

1. Raise the carriage cover directly above the platen clutch knob.
2. Raise the brush holder by pulling the lever on the right side towards you.
3. Insert a carriage control tape (channel one to the left) and close the brush holder.
4. Close the carriage cover.



1132 Carriage Familiarization

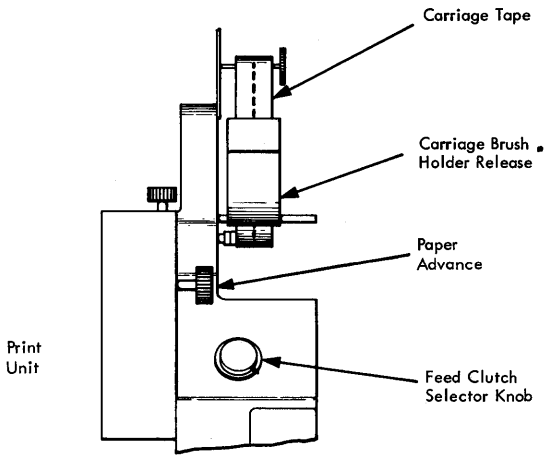
Press the CARRIAGE RESTORE key on the 1132 operator's panel. Engage the platen clutch (set to IN) and close the printer top cover. Press printer START. When the READY light comes on, the 1132 printer is ready to be selected.

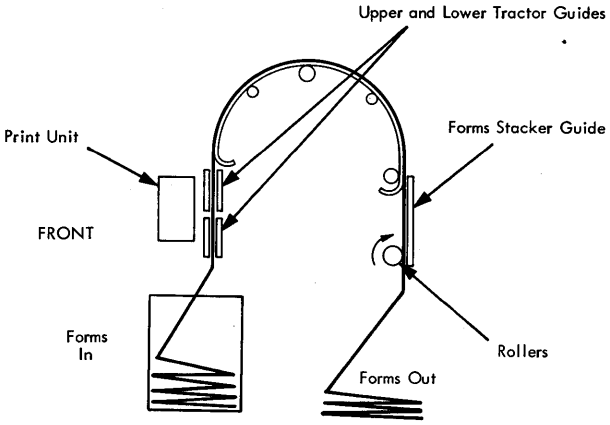
1403 Printer Ready Procedure

Pre-conditions. System power on. The END OF FORMS light will be on if there are no forms in the printer.

Readying the 1403 Printer. To load the forms into the printer:

1. Raise the printer cover and set the feed clutch selector knob to neutral. This knob is located on the right side of the print unit (see illustration below).
2. Unlock the print unit by pulling back on the release lever located on the left side of the unit. The print unit will swing out to the right.
3. Open the upper and lower left and right tractor guides.
4. Lift the forms up from their position below the front of the printer and lay them back across the arched rack at the top of the printer.
5. Line up the holes in the paper with the tractor pins and close all four tractor guides.
6. Close the print unit and lock the print unit release level. The ribbon drive will activate when the print unit is closed.
7. Using the PAPER ADVANCE knob located at the right end of the print unit, advance the paper until a crease between two forms is about 1/2 inch above the print position indicator bar on the print unit.





To load the carriage control tape in the printer:

1. Raise the carriage brush holder by pulling down on the lever on the right side. The carriage brush holder is located to the right and slightly above the print unit (see illustration).
2. Insert the carriage tape (channel one to the left) and close the brush holder.
3. Set the feed clutch selector knob to 6 or 8 lines per inch, whichever is desired.
4. Close the printer cover.

Press the CARRIAGE RESTORE key on the 1403 operator's panel. Continue to restore until sufficient paper has fed over the top arch to extend down the back of the printer. Open the rear cover of the printer and ensure that this paper has fed down between the forms stacker guide and the printer. If the paper has fed properly, the rollers on the forms stacker guide will keep a constant downward pull on the paper. Close the back cover.

Press CHECK RESET and printer START on the operator panel. When the PRINT READY light comes on, the 1403 printer is ready. Set the ENABLE/DISABLE switch on the 1133 to the ENABLE position (READY light on). The 1403 Printer is now ready to be selected.

2310 Disk Storage Ready Procedures

Pre-condition. System power on, CARTRIDGE UNLOCKED lights on the 2310 operator's panel on.

Readying the 2310 Disk Storage Drive.

1. Open the front door of the disk drive.
2. Grasp the handle of the access release mechanism of the drive to be loaded (drive 1 or 3 on top, 2 or 4 on the bottom) and pull out and down.

3. Pick up the cartridge and, holding the cartridge with the IBM name towards you and on the left, insert the cartridge into the slot.
4. When the cartridge is seated, raise the access release handle to lock the cartridge into place. If desired, load the other drive on the 2310 disk storage unit.

Close the front door of the disk storage unit. Turn on the START/STOP switch for the desired drives. The CARTRIDGE UNLOCKED lights will go out when the drives start to turn. When the drives come up to speed (approximately 90 seconds), the indicators showing the drive numbers will light, thus showing that the heads are loaded and the drives are ready.

When the drives are required by the system, set the ENABLE/DISABLE switches on the 2310 disk storage drives and on the 1133 to the ENABLE position (1133 READY light on). The 2310 disk storage drives are now ready to be selected.

1627 Plotter Ready Procedure

Pre-conditions. System power on.

Readying the 1627 Plotter. Load the chart paper using the following procedure.

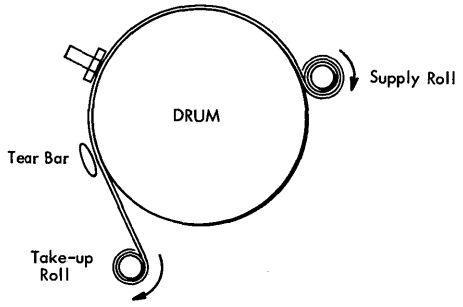
1. Ensure the 1627 Power switch is OFF (1627 power on indicator lamp out).
2. Remove the pen assembly, if installed, by loosening the knurled knob at the bottom of the pen holder and lifting the assembly out of the carriage.

Caution

Use care when handling the pen assembly. This assembly is manufactured to close tolerances for optimum performance.

3. Rotate the right rear chart spool by hand until the drive key is pointing upward.
4. Hold the new roll of chart paper so that the key slot in the core is pointing upward. Place the roll against the spring-loaded left rear idler spool and force the spool to the left.
5. Lower the paper roll into the paper well and slide the right end onto the drive spool. Make certain the drive key engages the key slot in the core. The paper should feed out from under the roll and over the drum (see illustration).
6. Install a paper roll core on the front spool below the drum, in the same manner as with the paper roll.
7. Pull a short length of paper off the roll, slide the end under the carriage rods, under the tear bar, behind the core, and fasten it to the front side of the core with two or three short pieces of cellophane tape. Wind one or two turns of paper onto the core. Make certain the drum sprockets are properly meshed with the sprocket holes on both sides of the paper.
8. Reinstall the pen assembly in the carriage.
9. Turn the 1627 power switch to ON. The 1627 power on indicator will come on.

NOTE: The pen is down when the power is off; therefore, the pen assembly should be installed with the carriage over an area outside the "recording area". If the pen does not raise when power is turned on, turn the pen switch to DOWN, then to UP.



With the pen in the UP position, use the drum (x axis) and carriage (Y axis) controls to position the pen for the first plot. The 1627 plotter is now ready to be selected.

1231 Optical Mark Page Reader Ready Procedure

Pre-conditions. System power on, RESET light on, and READ light off. SYSTEM STOP light on if the CPU is stopped.

Readying the Optical Mark Page Reader. Place the data sheets in the hopper with the side to be read facing up and the top edge positioned to feed first. Set the FEED MODE switch to ON-DEMAND. The settings of the other selector switches on the operator console are dependent on the data being read.

Press PROGRAM LOAD on the 1231 operator's console. This action clears the delay line and conditions the machine for program loading. The PROGRAM LOAD light turns on. Press 1231 RESET. This causes the hopper to raise to the ready position. The RESET light turns off and the 1231 START light turns on. Press

1231 START. The first data sheet in the hopper is fed through the 1231, loading the delay line. The first data sheet is now in the stacker. The PROGRAM LOAD light turns off. Press 1231 START. The 1231 START light turns off.

With all lights off on the 1231 operator's console (the SYSTEM STOP light may be on), the 1231 optical mark page reader is ready to be selected.

USING THE IBM 1130 WITH THE MONITOR SYSTEM
When all I/O units required for a job are on-line and in a ready condition, the user may proceed as follows.

Loading a Program from Card or Paper Tape
On the Console:

- Press IMM STOP (press PROGRAM STOP if the Monitor system is running).
- Press RESET.
- Check that the console Mode switch is set to RUN mode.
- With the reader wired for IPL in a ready state, press PROGRAM LOAD (if the system has a 2501 and a 1442-6 or -7, ensure that the 1442 is not ready). The first record (usually a loader) is read into core starting at location zero. Instructions on this record tell the system what operation is to be performed next, usually the loading of more records from the input device.
- When a card reader goes not ready, press reader START to read in the last card and pass control to the loaded program. This action is not required with paper tape input.

Altering or Displaying the Contents of a Selected Core Location Using the Console Entry Switches

- With the system stopped, set the console Mode switch to LOAD.
- Set the console entry switches to the desired four-character hexadecimal core address. Switches 0-3 constitute the first hexadecimal character, 4-7 the second, etc.
- Press LOAD IAR (the selected address is displayed in the IAR).

To display the contents of the address:
- Set the console Mode switch to DISPLAY.
- Press PROGRAM START.

The contents of the selected location is displayed in the Storage Buffer Register. Succesive pressing of the PROGRAM START key will display consecutive core locations.

To alter the contents of the address:
Set the new data word in the console entry switches.
Set the console MODE switch to LOAD.
Press PROGRAM START.

To return to system operation:
Set the console Mode switch to RUN
Press PROGRAM START.

NOTE: At a Monitor system WAIT, the address of instruction causing the WAIT is at the address displayed in the IAR minus 1.

Reading the Console Entry Switches Under User Program Control
The setting of the console entry switches can be read by an XIO read instruction at any time during the execution of a user-written stored program. The device code of the instruction is set to 00111.

Entering Programs from the Keyboard Under Monitor System Control
A single Monitor control record or an entire program including all required control records and data records can be entered from the 1130 Keyboard using the Monitor System. Control is passed to the Keyboard when a //TYP Monitor control record is read from the principal input device.

Control is returned to the principal input device when a // TEND record is entered from the keyboard.

Keyboard Operation
When the //TYP Monitor control record is read, the Console Printer performs a carrier return and the KB SELECT light on the Keyboard operator's panel turns on. The system is now ready to accept input from the keyboard.

Enter all control records in the correct format. Use the space bar for blanks. The records are printed on the Console Printer as they are entered. Press EOF to end each record. An NL (new line) character is entered, the carrier is restored to a new line, and the keyboard is reselected. This sequence of events continues until a // TEND record is entered. Pressing EOF then returns control to the principal input device.

Up to 80 characters can be entered in each record. If an error is made when entering a record from the Keyboard, the user can elect to backspace and correct the entry or re-enter the entire record.

Backspace. When the backspace key (←) is pressed, the last graphic character entered is slashed and the address of the next character to be read is decremented by +1. If the backspace key is pressed twice consecutively, the character address is decremented by +2, but only the last graphic character is slashed. For example, assume that *DELET has been entered and the backspace key is pressed three times. The next graphic character replaces the L, but only the T is slashed. If the characters FINE are used for replacement, the paper would show *DELETFINE, but *DEFINE would be stored in the buffer.

Re-entry. When the ERASE FIELD key is pressed, a character interrupt signals the interrupt response subroutine that the previously-entered Keyboard record is in error and is to be re-entered. The subroutine prints two slashes on the Console Printer, restores the carrier to a new line, and prepares to replace the old record in the I/O area with the new record. The new record overlays the previous record, character by character. Blanks are placed in the buffer following the NL character which terminated the new record.

Console Functions While Under Monitor System Control

PROGRAM STOP Key. Pressing this key causes a level 5 interrupt and an entry to the PROGRAM STOP key trap providing there are no user-written device subroutines associated with level 5 currently in core. The trap consists of a WAIT and a branch. When the PROGRAM START key is pressed, the interrupt level is turned off and execution resumes following the point of the level 5 interrupt.

The PROGRAM STOP key trap allows the user to stop the entire 1130 system with the ability to continue execution without disturbing the system status or the contents of core storage.

If a higher interrupt level is being serviced when the PROGRAM STOP key is pressed, the PROGRAM STOP key interrupt is masked until the current operation is completed.

INT REQ. Pressing the Interrupt Request key causes the current job to be aborted. Control is returned to the Supervisor, which then searches for the next JOB record in the input stream. The user may program this key differently if he desires.

IMM STOP. Do not press IMM STOP when running under Monitor system control. The contents of a system cartridge can be destroyed, necessitating a regeneration of the system.

Manual Dump of the Monitor System

If a problem occurs during the execution of a core load and the user desires to dump core storage, the dump entry point in the Skeleton Supervisor can be entered by a manually executed transfer to location zero or to the dump entry entry point plus one (location $DUMP+1). A dump of the entire contents of core storage is given in hexadecimal and the dump program (see Supervisor Core Dump Program) executes a CALL EXIT thereby terminating the execution of the core load in progress.

If the dump was necessitated by the introduction of bad data in a Monitor system program, the system may loop rather than perform the dump. If this occurs when DISKZ is in use, the user must manually clear $IOCT and $DBSY before reinitiating the dump.

Before describing the contents of a Monitor system and non-system cartridge, it is necessary to briefly describe the steps to initialize the cartridges for use on the system.

- When the Monitor system is loaded by the System Loader onto a disk cartridge that has been initialized by the Disk Cartridge Initialization Program (DCIP), that cartridge becomes a system cartridge.

- Placement of a system cartridge on any physical drive readies the system for the user-initiated cold start procedure. The cold start establishes the physical drive on which the system cartridge has been placed as logical drive 0, which is, by definition, the system drive. The system cartridge on logical drive 0 is then called the master cartridge.

- The other cartridges on the system (also initialized by DCIP) are called non-system cartridges. If desired, the IBM system can then be loaded on any of these cartridges, thus making them system cartridges. However, once a cold start has been performed and a master cartridge established, all other cartridges, system or non-system, are called satellite cartridges.

The organization of programs and areas on system and non-system cartridges is described and illustrated below.

Sector @IDAD of any Cartridge

This sector, illustrated in Figure 1, contains the defective cylinder table, the cartridge ID, the cartridge copy code, a reserved area, and an Error Message program.

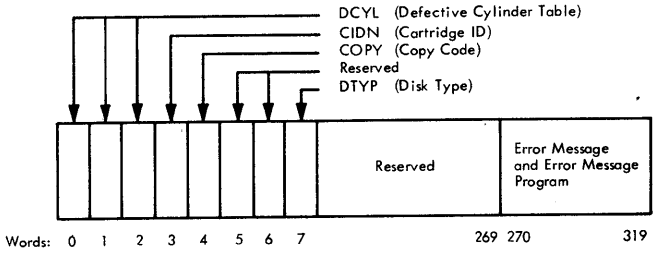
The defective cylinder table contains the addresses of the first sector on any cylinders on the cartridge that are not capable of accurately storing data. The Monitor system can be operated from a cartridge with up to 3 defective cylinders.

The cartridge ID is a hexadecimal number in the range /0001 - /7FFF that uniquely identifies the cartridge.

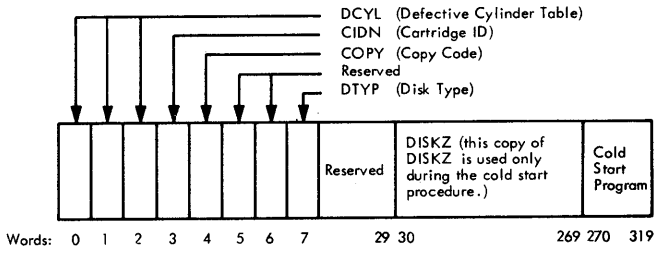
The copy ID (updated by DCIP or COPY) gives the user the ability to identify any given copy of a system or non-system cartridge. Each time a copy is made, word 5 (initially 0) is incremented by one, i.e., word 5 of the copy is always one greater than the source.

The reserved area of sector @IDAD is used by the System Loader when the IBM System is loaded on the cartridge (see Figure 2).

Following initialization by DCIP (or DISC), an error message and the program that causes it to print are stored in sector @IDAD. The error message -- NON-SYST. CART. ERROR -- is printed if an attempt is made to cold start a cartridge that is not a system cartridge. This message and the program that prints it are overlaid by the Cold Start program when the Monitor system is loaded on the cartridge.



- Figure 1. Contents of Sector @IDAD after Initialization by DCIP or DISC



- Figure 2. Contents of Sector @IDAD after the IBM System is on Disk

SYSTEM CARTRIDGE

The system cartridge is divided into three logical areas, which are illustrated in Figure 3. These areas are the IBM System Area, the User Area, and the Working Storage. In addition, the user may define a Fixed Area on disk for the purpose of storing programs and/or data files into permanent locations so they may be referenced by sector address.

IBM SYSTEM AREA

During system generation, the IBM system decks are loaded on disk by the System Loader. The disk areas occupied by the IBM-supplied Monitor programs, and the disk areas reserved for the use of these programs, are collectively known as the IBM System Area.

The contents of the IBM System Area are listed below.

Cylinder 0

The contents of sector @IDAD have already been described (see Figure 2). Sector @DCOM contains the Disk Communications Area, which is described below (see DCOM).

Sector @RIAD contains the Resident Image. The Resident Image is a copy of the Resident Monitor without a disk I/O subroutine, that is, it is a reflection of COMMA and the Skeleton Supervisor (see Resident Monitor in the section Supervisor). The Resident Image is used to initialize the Resident Monitor during a cold start.

The System Location Equivalence Table (SLET) resides on sectors @SLET and @SLET+1. SLET is composed of an identification number, core loading address, word count, and sector address for every phase of every Monitor program.

Sector 5 is reserved.

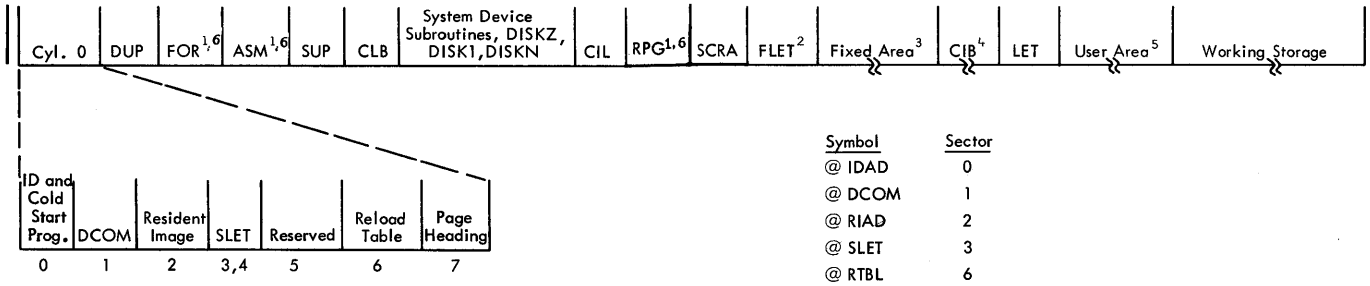
Sector @RTBL contains the Reload Table, which is used by the System Loader during a program reload and by the Disk Utility Program (DUP) when deleting the Assembler, FORTRAN Compiler, or RPG Compiler The Reload Table is established during system generation when the System Loader reads the Type 81 System Loader control card.

Sector @HDNG is used to store the page heading that appears at the top of each page printed by a Monitor program.

DCOM

The Disk Communications area, located in sector @DCOM of a system cartridge, contains the parameters that must be passed from one Monitor program to another and that must be accessed through disk storage (as opposed to core storage). Generally speaking, parameters that are not required when fetching a link stored in Disk Core Image format are found in DCOM. A listing of DCOM is provided in Appendix H, Resident Monitor.

DCOM is divided into two parts. The first part of DCOM contains the parameters that are not related to all the disk cartridges, for example, the core map switch. The second part of DCOM contains the cartridge-related parameters: cartridge ID, LET address, file protect address, etc. Each of the



Symbol	Sector
@ IDAD	0
@ DCOM	1
@ RIAD	2
@ SLET	3
@ RTBL	6
@ HDNG	7

1. Can be deleted from the system by the user
2. Present only if a Fixed Area is defined for this cartridge by the user
3. Optionally defined by the user
4. May not be deleted by the user from a system cartridge.
5. Initially contains only the System Library; user-written programs may be added
6. Optional Monitor program

●Figure 3. Layout of a System Cartridge

parameters in the second part is in the form of a five-word table, one word for the corresponding value for each of the five possible cartridges. The five words of each table, known as a quintuple, are arranged in the order of logical drive numbers; that is, the first is for logical drive 0, the second for logical drive 1, etc.

The parameters for the non-system cartridges are obtained from the DCOM areas of those cartridges and stored in the DCOM on the system cartridge through the use of a merge operation. For example, the file protect address quintuple on the master DCOM is composed of the file protect address from each of the other four logical drives, plus its own file protect address.

The subroutine for performing the DCOM merge operation is called SYSUP and must be called by the user for the purpose of updating the DCOM parameters if cartridges are changed during a job (see SYSUP in the section System Utility Subroutines). A similar subroutine is an integral part of the Monitor Control Record Analyzer and is executed during JOB processing.

During the processing of a JOB record, the DCOMs of only those cartridges listed on the JOB record are merged into the master DCOM. The parameter tables for the other drives are cleared to zero.

DCOM Indicator Words

In the following paragraphs, "set" means that a value is stored in the word in question; "reset" means that it is cleared to zero.

Working Storage Indicator Word. DCOM contains a Working Storage Indicator word for each cartridge on the system. The Working Storage Indicator word for a cartridge contains the disk block count of any DSF program, DCI program, or Data File currently in Working Storage on that cartridge.

The Working Storage Indicator word for a cartridge is set (1) at the completion of a DUP operation in which information is transferred to Working Storage and (2) at the completion of any assembly or successful compilation, at which time the Assembler, FORTRAN Compiler, or RPG Compiler places the assembled/compiled object program in Working Storage.

The Working Storage Indicator word for a specific cartridge is reset 1) following any STORE operation to the User Area on that cartridge and 2) following the building of a core load that requires LOCALs and/or SOCALs. Because the User Area is increased at the expense of Working Storage, it is assumed that any STORE operation to the User Area overlays a part of the Working Storage area with that which was stored. Therefore, the Working Storage Indicator word is reset.

Format Indicator Word. DCOM contains a Format Indicator word for each cartridge on the system. The Format Indicator word for a cartridge indicates the format of any DSF program, DCI program, or Data File currently in Working Storage on that cartridge.

The Format Indicator word for a cartridge is set and reset under the same conditions as the Working Storage Indicator word for the same cartridge.

Temporary Mode Indicator Word. The Temporary Mode Indicator word in DCOM is set by the Supervisor when temporary mode is indicated by the user in the JOB record (see // JOB under Monitor Control Records). Table 1 lists DUP operations and any restrictions that apply when in temporary mode. The temporary mode indicator is set/reset during JOB processing.

Monitor System Disk Areas

Following cylinder 0, the IBM System is loaded onto disk in the order shown in Figure 3. The individual programs are described in the section of this manual entitled Monitor Programs; the disk areas are described below.

System Device Subroutine Area. The System Device Subroutine Area contains the following components.

- The subroutines used by the Monitor programs to operate the following print devices.

  1403 Printer
  1132 Printer
  Console Printer

Table 1. Restrictions on DUP Operations in Temporary Mode

DUP Operations	Restrictions
DUMP	None
DUMPDATA, DUMPDATAE	None
STORE	None
STORECI	To UA only
STOREDATA, STOREDATAE	To UA and WS only
STOREDATACI	To UA only
STOREMOD	Not allowed
DUMPLET	None
DUMPFLET	None
DWADR	Not allowed
DELETE	Not allowed
DEFINE FIXED AREA	Not allowed
DEFINE VOID ASSEMBLER	Not allowed
DEFINE VOID FORTRAN	Not allowed
DEFINE VOID RPG	Not allowed

- The subroutines used by the Monitor programs to operate the following I/O devices.
  2501 Card Reader/1442 Card Punch, model 5, 6, or 7
  1442 Card Read Punch, model 6 or 7
  1134/1055 Paper Tape Reader/Punch
  Keyboard/Console Printer

- The I/O character code conversion subroutines used in conjunction with the I/O subroutine for the following devices.
  2501 Card Reader/1442 Card Punch
  1134/1055 Paper Tape Reader/Punch
  Keyboard/Console Printer

- The disk I/O subroutines.
  DISKZ
  DISK1
  DISKN

All of the subroutines in the System Device Subroutine Area, except the disk I/O subroutines, are naturally relocatable and are intended for use only by Monitor programs.

The disk I/O subroutines are located in this area rather than in the System Library because they are processed by the Core Load Builder differently than those stored in the System Library.

DISKZ is stored twice on the disk, once in sector @IDAD with the Cold Start program, and once in the System Device Subroutine Area with DISK1 and DISKN. Cold Start initializes with the DISKZ in sector @IDAD, in all other cases, DISKZ is fetched from the System Device Subroutine Area.

Supervisor Control Record Area. The Supervisor Control Record Area (SCRA) is the area in which Supervisor control records (LOCAL, NOCAL, FILES, G2250 and EQUAT) are saved. These records, except the EQUAT record, are read from the input stream (following an XEQ or STORECI control record) and are stored in the SCRA for subsequent processing by the Lore Load Builder. The processing of the EQUAT record is simular to that of the other Supervisor control records, but it is read from the input stream following a JOB control record.

Fixed Location Equivalence Table (FLET). This table is a directory to the contents of the Fixed Area for the cartridge on which it appears. There is one FLET entry for:

- Each program stored in Disk Core Image format

- Each Data File stored in Disk Data format

- The padding required to permit a DCI program or Data File to be stored on a sector boundary.

Each FLET entry specifies the name of the DCI program or Data File, its format, and its size in disk blocks.

Each cartridge on the system having a Fixed Area has a FLET. Regardless of the size of the Fixed Area (one cylinder is the minimum requirement), the FLET for a cartridge occupies the cylinder preceding the fixed Area (a minimum of 2 cylinders of Fixed Area may be initially defined. The first cylinder becomes FLET).

The sector address of the first sector of FLET on a given cartridge may be obtained from the LET on the same cartridge. The last LET header contains this sector address.

A FLET dump is illustrated in Appendix G.

Core Image Buffer (CIB). The CIB is the area on disk in which the Core Load Builder builds any portion of a core load that is to reside below location 4096. It is also used by the Core Image Loader to save any COMMON defined below location 4096 during the transfer of control from one link to the next.

Location Equivalence Table (LET). The LET on a cartridge is a directory to the contents of the User Area on that cartridge. There is one LET entry for:

- Each entry point for each program stored in Disk System format

- Each program stored in Disk Core Image format

- Each Data File stored in Disk Data format

- The padding required to permit a DCI program or Data File to be stored on a sector boundary.

Each LET entry specifies the name of an entry point, DCI program, or Data File; its format; and its size in disk blocks.

Each cartridge on the system has a LET. However, a cartridge has a User Area only if there is an entry in the LET on that cartridge other than a dummy entry (1DUMY). On a system cartridge, LET occupies the cylinder preceding the User Area.

COMMA contains the sector address of the first sector of LET for each cartridge being used in a given job.

A LET dump is illustrated in Appendix G.

USER AREA

The User Area (UA) is the area in which the user can store programs in Disk System format or Disk Core Image format and/or Data Files in Disk Data format. The User Area is defined on any cartridge when the cartridge is initialized. However, its size is 0 sectors until the first DSF program, DCI program, or Data

File is stored in the User Area on that cartridge. The User Area occupies as many sectors as are required to contain the DSF programs, DCI programs, and Data Files stored on that cartridge.

When a DSF program, DCI program, or Data File is to be added to the User Area, it is stored at the start of Working Storage, that is, immediately following the end of the User Area. The area occupied by the new DSF program, DCI program, or Data File is then incorporated into the User Area, and Working Storage is decreased by the size of that area.

DSF programs are stored in the User Area starting at the beginning of a disk block; DCI programs and Data Files are stored starting at the beginning of a sector.

The User Area is packed when a DSF program, DCI program, or Data File is deleted from the User Area; that is, the DSF programs, DCI programs, and/or Data Files in the User Area are moved so as to occupy the vacancy (the area formerly occupied by the deleted DSF program, DCI program, or Data File). In packing, DSF programs are moved to the first disk block boundary in the vacancy; DCI programs and Data Files are moved to the first sector boundary in the vacancy. All following DSF programs, DCI programs, and Data Files are similarly packed.

The area gained by packing the User Area is returned to Working Storage.

WORKING STORAGE AREA

Working Storage (WS) is that area on all cartridges that is not defined as the User/Fixed Area and, on a system cartridge, as the IBM System Area. Working Storage is available to Monitor and user programs alike as temporary disk storage. It extends from the sector boundary immediately following the User Area to the end of the cartridge (cylinder 199).

FIXED AREA

The Fixed Area (FX) is the area in which the user may store programs in Disk Core Image format and/or Data Files in Disk Data format if it is desired that these programs and Data Files always occupy the same sectors. The Fixed Area is optionally defined on any cartridge by the use of the DUP operation, DEFINE FIXED AREA. This operation is also used to increase or decrease the size of the Fixed Area.

When a DCI program or Data File is stored in the Fixed Area, it is stored starting at the beginning of a sector. When a DCI program or Data File is deleted from the Fixed Area, no packing of the Fixed Area occurs. Hence, DCI programs and Data Files in this area reside at fixed sector addresses and can be referenced as such by the user.

NON-SYSTEM CARTRIDGE

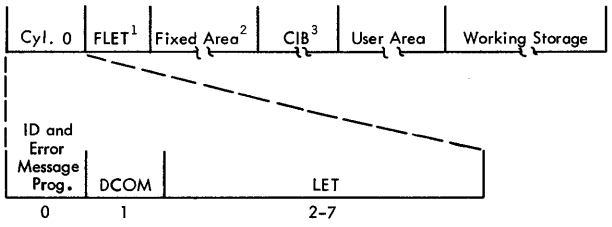
Figure 4 shows the layout of a non-system cartridge, a cartridge that contains no Monitor programs. Such a cartridge on multi-drive 1130 systems can be used exclusively for the storage of data and/or programs and is called a satellite cartridge.

Except for cylinder 0, which is described below, the definitions of the areas present on a non-system cartridge are the same as those previously described for a system cartridge.

Cylinder 0

Sector @IDAD of cylinder 0 on a non-system cartridge contains the parameters established by DCIP or DISC (see Sector @IDAD of any Cartridge). Note however that the Error Message program has not been overlaid since this is not a system cartridge. An attempt to cold start a non-system cartridge will cause the error message to be printed on the Console Printer. Sector @DCOM of cylinder 0 contains only that information from DCOM applicable to this non-system cartridge (see DCOM).

The location equivalence table (LET) for the cartridge (see Location Equivalence Table), occupies the remaining six sectors of cylinder 0.



1. Present only if a Fixed Area is defined for this cartridge by the user
2. Optionally defined by the user
3. May be deleted by the user. However, a CIB must be present on at least one of the cartridges on the system at any given time.

Figure 4. Layout of a Non-System Cartridge

The Monitor programs: Supervisor, DUP, Assembler, FORTRAN Compiler, (optionally, RPG Compiler,) Core Load Builder, and Core Image Loader reside in the IBM System Area on the master cartridge. The following paragraphs briefly describe these programs and the subprograms within them that are of most interest to the user.

SUPERVISOR

The Supervisor is actually a group of programs and areas which are responsible for the control functions of the Monitor system. The Supervisor reads control records included in the stacked job input, decodes them, and calls the appropriate Monitor program to perform the specified operation. The Supervisor initially achieves control of the Monitor system through the user-initiated cold start procedure (see Cold Start).

A portion of the Supervisor is located in core storage. This portion is called the Resident Monitor.

RESIDENT MONITOR

The resident portion of the Monitor system consists of (1) a data area used for system parameters and for communication between Monitor programs (COMMA), (2) the Skeleton Supervisor, and (3) a disk I/O subroutine (either DISKZ, DISK1, or DISKN).

Core Communications Area (COMMA)

In general, COMMA consists of the parameters required by the Core Image Loader to process a CALL LINK to a DCI program without referring to the Disk Communications area (DCOM). This information is interspersed with parts of the Skeleton Supervisor (see Appendix H, Resident Monitor).

Skeleton Supervisor

On any entry to the Resident Monitor (EXIT, LINK, or DUMP), the Skeleton Supervisor calls the Core Image Loader, which determines where the Skeleton Supervisor was entered and either calls the Supervisor if the entry was at EXIT or DUMP or fetches and transfers control to the core load specified in the CALL LINK statement if the entry was at LINK. (If the link to be executed is in Disk System format, it will be necessary to call the Core Load Builder before transferring control to the core load itself.)

The use of the Core Image Loader as an intermediate supervisor allows the Monitor system to achieve efficient link-to-link transfer of control.

The Skeleton Supervisor, which is interspersed with COMMA, consists of the entry points and subroutines described below.

LINK Entry Point. LINK is the entry point in the Skeleton Supervisor that accomplishes link-to-link transfer of control.

EXIT Entry Point. EXIT is the entry point in the Skeleton Supervisor that accomplishes link-to-Supervisor transfer of control.

DUMP Entry Point. DUMP is the entry point in the Skeleton Supervisor that prints out the contents of core storage between specified limits. Dynamic dumps are obtained through the DUMP entry point; terminal dumps are obtained through the DUMP entry point plus 1.

ILS02 Subroutine. The ILS02 subroutine handles the servicing of interrupts on level 2. Only the disk devices on the system interrupt on level 2. Since the Skeleton Supervisor requires the disk, the ILS02 subroutine is a part of the Resident Monitor.

ILS04 Subroutine. The ILS04 subroutine handles the servicing of interrupts on level 4. One of the devices that interrupt on level 4 is the Keyboard. Since the user may perform a console interrupt request at any time, the ILS04 subroutine is a part of the Resident Monitor.

Preoperative Error Trap. The preoperative error trap is entered by all ISS subroutines when an error is detected before an operation has been initiated. The trap consists of a WAIT and a branch. When the PROGRAM START key is pressed, execution resumes at the location following the branch to this trap. Under certain conditions, this trap is entered when no error has occurred, e.g., FORTRAN PAUSE.

Postoperative Error Traps. One of the postoperative error traps (there is one for each interrupt level) is entered by all ISS subroutines when an error is detected after an operation has been initiated. Each trap consists of a WAIT and a branch. When the PROGRAM

START key is pressed, control is returned to the ISS subroutine, which may then retry the operation in error.

PROGRAM STOP Key Trap. The PROGRAM STOP key trap is entered if a level 5 interrupt occurs and there is no user-written device subroutine associated with level 5 currently in core. The trap consists of a WAIT and a branch. When the PROGRAM START key is pressed, the interrupt level is turned off and execution resumes following the point of the level 5 interrupt.

This trap allows the user to stop the entire 1130 system with the ability to continue execution without disturbing the system status or the contents of core storage.

If a higher interrupt level is being serviced when the PROGRAM STOP key is pressed, the PROGRAM STOP key interrupt is masked until the current operation is completed.

Interrupt Request Key

When the INT REQ key is pressed, all busy indicators are turned off and a switch in COMMA is set to instruct the Supervisor to pass input records until a JOB record is encountered. Parts of the Monitor which should not be interrupted before completion, e.g., SYSUP, delay the interrupt request until they have completed their operation.

Disk I/O Subroutine

The disk I/O subroutine required by the program in control resides in core storage following the Skeleton Supervisor. The following table lists the disk I/O subroutines, their approximate sizes, and the corresponding addresses of the end of the Resident Monitor plus 1.

Subroutine (in Core)	End of Resident Monitor +1 (Core Location) Decimal	Hexadecimal
DISKZ	480	/01E0
DISK1	660	/0294
DISKN	930	/03A2

DISKZ is the disk I/O subroutine used by all system programs. DISKZ is initially loaded with the Resident Monitor.

Prior to the execution of a core load requiring DISK1 or DISKN, the Core Image Loader overlays DISKZ with the required disk I/O subroutine. When control is returned to the Supervisor, the Core Image Loader overlays the disk I/O subroutine currently in core (if DISK1 or DISKN) with DISKZ. User programs, including those written in FORTRAN or RPG, may use any of the three disk I/O subroutines; however, only one disk I/O subroutine may be referenced in a given core load. In this context "core load" includes column 19 of the XEQ record (the entry in column 19 of the XEQ record specifies the version of the disk I/O subroutine to be used by the core load during execution).

DISK-RESIDENT SUPERVISOR PROGRAMS

The programs described below are the disk-resident programs that constitute the Supervisor. One of these programs is fetched and given control by the Core Image Loader, depending upon the entry made in the Skeleton Supervisor; the Monitor Control Record Analyzer is called following an EXIT entry, the DUMP program following a DUMP entry.

Monitor Control Record Analyzer

The Monitor Control Record Analyzer (1) reads a Monitor control record or Supervisor control record from the input stream, (2) prints the control record on the principal print device, and (3) fetches the required Monitor program and transfers control to it. Supervisor control records are stored on disk in the Supervisor Control Record Area.

MONITOR CONTROL RECORDS

Monitor control records perform the load and control functions of the Monitor system. The individual control records are described in the paragraphs that follow.

Where shown in the control record format, the character "b" indicates that the column must be blank. Remarks may be punched in the card columns listed as "not used" in the control record formats.

// JOB

The JOB control record defines the start of a new job. It causes the Supervisor to perform the job initialization procedure, which includes:

- The initialization of COMMA

- The initialization of the parameters in DCOM
- The setting of the Temporary Mode Indicator if a T is present in column 8 of the JOB control record (reset if no T in column 8). If set, the temporary mode indicator causes all DSF programs, DCI programs, or Data files stored in the User Area by DUP during the current job to be deleted automatically from that area at the end of the job (that is, at the beginning of the next job). See DCOM for DUP restrictions while in the temporary mode.
- The definition of the cartridges to be used during the current job. IDs 1 through 5 on the JOB control record specify the cartridges to be used. These cartridges may be mounted on the physical drives in any order. The order of the IDs in the JOB control record specifies the logical assignments for the cartridges. IDs 1 through 5 correspond to logical drives 0 through 4, and they must be specified consecutively. If only three drives are to be used IDs 1-3 only are specified. The cartridge-related entries of COMMA and DCOM (quintuples) are filled in according to the logical order specified by the user. The first ID may be left blank, in which case the master cartridge for the last JOB will also be the master for this JOB.
- The definition of the cartridge on which the Core Image Buffer for the current job is to be found. The ID of the cartridge containing the CIB must follow the field of the fifth cartridge ID. If the CIB ID is omitted, the CIB on the master cartridge is used. Core image programs can be built faster if the CIB is assigned to a cartridge other than the master cartridge.
- The definition of the cartridge containing the Working Storage to be used by the Monitor programs (System Working Storage). The ID of the cartridge to be used for Working Storage by the Monitor System must follow the CIB ID. If the Working Storage ID is omitted, all Monitor programs use the Working Storage on the master cartridge (except when otherwise specified, see DUP Control Records). Core Image programs can be built faster if the System Working Storage is on a cartridge other than the master cartridge. They can be built even faster if the CIB, the system Working Storage, and the Monitor system itself are all on separate cartridges. Assemblies are also faster if System Working Storage is on a separate cartridge. (See *FILES, page 23, for Working Storage for user programs.)
- The definition of the cartridge containing the unformatted I/O ($$$$$) disk buffer area to be used with this job.
- The starting of a new page on the principal print device. A skip to channel 1 is executed on the 1132 or 1403 Printer; or five consecutive carriage returns are made on the Console Printer. The page count is reset to 1, and the surrent page heading is replaced with whatever appears in column 51-58 of the JOB control record. HDNG (assembler language) statements and **(FORTRAN control record) records will cause additional information to be printed.

- The reading of the Supervisor control records of EQUAT type, if any, and writing them on disk in the Supervisor Control Record Area (SCRA).

The format of the JOB control record is as follows.

Card Column	Contents	Notes
1-6	//bJOB	
7	Reserved	
8	Temporary mode indicator	T or blank. A T indocates that temporary mode is desired for this job.
9-10	Reserved	
11-14	First ID	This is the ID of the master cartridge (logical drive 0).
15	Reserved	
16-19	Second ID	This is the ID of the cartridge on logical drive 1.
20	Reserved	
21-24	Third ID	This is the ID of the cartridge on logical drive 2.
25	Reserved	
26-29	Fourth ID	This is the ID of the cartridge on logical drive 3.
30	Reserved	
31-34	Fifth ID	This is the ID of the cartridge on logical drive 4.
35	Reserved	
36-39	CIB ID	This is the ID of the cartridge containing the CIB to be used during this job.
40	Reserved	
41-44	Working Storage ID	This is the ID of the cartridge containing the Working Storage to be used by the monitor during this job. See *FILES, p. 23, for details on Working Storage for user programs.
45	Reserved	
46-49	Unformatted disk I/O ID	This is the ID of the cartridge containing the unformatted disk I/O area to be used during this job.
50	Reserved	
51-58	Date, Name, etc.	This information is printed at the top of every page of the listing on the principal print device during this job.
59	Not used	
60-61	EQUAT record count	This number specifies how many EQUAT records follow this JOB record.
62-80	Not used	

// ASM

This control record causes the Supervisor to read the Assembler into core storage and transfer control to it. Any Assembler control records and the source statements to be assembled must follow this control record. Comments control records (// *) may not follow this control record.

The format of the ASM control record is as follows.

Card Column	Contents	Notes
1-6	//bASM	
7-80	Not used	

(See *FILES, p. 23 for working storage for user programs.)

// FOR

This control record causes the Supervisor to read the FORTRAN Compiler into core storage and transfer control to it. Any FORTRAN control records and the source statements to be compiled must follow this control record. Comments control records (// *) may not follow this control record.

The format of the FOR control record is as follows.

Card Column	Contents	Notes
1-6	//bFOR	
7-80	Not used	

//RPG

This control record causes the Supervisor to read the RPG Compiler into core storage and transfer control to it. The RPG control card and RPG specification statements to be compiled must follow this control record. Comments control records (//*) may not follow this control record.

The format of the RPG control record is as follows.

Card Column	Contents	Notes
1-6	//bRPG	
7-80	Not used	

// DUP

This control record causes the Supervisor to read the control portion of the Disk Utility Program into core storage and transfer control to it. A DUP control record must follow this control record. Only one // DUP control record is required to process a stack of DUP control records, provided no Monitor control record other than the Comments control record (// *) is encountered.

The format of the DUP Monitor control record is as follows.

Card Column	Contents	Notes
1-6	//bDUP	
7-80	Not used	

// XEQ

This control record causes the Supervisor to initialize for core load execution. If the name specified in this control record (columns 8 through 12) is that of a mainline program stored in Disk System format, the Supervisor reads the Supervisor control records (LOCAL, NOCAL, FILES, or G2250), if any, from the input stream and writes them in the Supervisor Control Record Area (SCRA). The Core Load Builder is then called to build a core image program from the mainline program.

If no name is specified on this control record, a mainline program in Disk System format is assumed to be stored in the Working Storage of the cartridge specified in columns 21-24. The Supervisor then processes the Supervisor control records and calls the Core Load Builder via the LINK entry point in the Resident Monitor.

After the Core Image program has been built, or if the name in the control record is that of a program already stored on disk in DCI format, the Core Image Loader is called to read the core load into core storage and transfer control to it.

If an L is punched in column 14 of this control record, a core map is printed by the Core Load Builder during the building of the core image program. In addition, a core map is printed for all DSF links during the execution (see Reading a Core Map and a File Map for an example of a core map). These core maps include:

- The execution address of the mainline program
- The names and execution addresses of all subprograms in the core load
- All file allocations, with the file number, sector address (relative to first sector of Working Storage for files in Working Storage, absolute otherwise), sector count, and either cartridge ID or the address of Working Storage. (If the file is in Working Storage, the address of Working Storage will be included; otherwise, the name of the file is printed.)

Columns 16 and 17 of this control record contain the right-justified decimal count of Supervisor control records to be read by the Supervisor before calling the Core Load Builder.

Column 19 contains a character that identifies the disk I/O subroutine to be used by the core load during execution. If column 19 contains zero or one, DISK1 is fetched by the Core Image Loader along with the core load. If Column 19 contains an N, DISKN is fetched. If column 19 contains a blank or a Z, no disk I/O subroutine is fetched (that is, DISKZ, which is in core storage for use by the Monitor programs, is used by the core load). Any other character is illegal and will cause the execution to be bypassed. All links in Disk System format that are called during a given execution must utilize the same disk I/O subroutine as the link that precedes them in execution.

A punch in column 26 makes it possible for a LOCAL to call another LOCAL, provided the restrictions listed in "Programming Tips and Techniques" are met.

A punch in column 28 indicates that special ILSs are to be used for this core load. If column 28 is blank, the standard set of ILSs is used. The special ILSs, named with an X before the number (e.g., ILSX4), save and restore index register 3 and set index register 3 to point to the transfer vector, in addition to the functions of the standard ILSs.

The use of special ILSs opens up the possibility of using index register 3 in programs. Special ILSs require 5 more words per ILS than standard ILSs. Note that level 2 and level 4 ILSs, which normally are located in the Resident Monitor, will be loaded, together with other subroutines, as if they were user-written ILSs. The user can replace either ILS with a user-written ILS.

Comments control records (// *) may not follow an XEQ control record.

The format of the XEQ control record is as follows.

Card Column	Contents	Notes
1-6	//bXEQ	
7	Reserved	
8-12	Name	This is the name (left-justified) of the DSF program or DCI program to be executed.
13	Reserved	
14	Core Map Indicator	L or blank. An L indicates that a core map is to be printed for this and all following DSF links during this execution.
15	Reserved	
16-17	Count	This is the right justified decimal number of Supervisor control records (LOCAL, NOCAL, FILES and G2250) that follow.
18	Reserved	
19	Disk I/O subroutine indicator	This column specifies the disk I/O subroutine to be loaded into core by the Core Image Loader for use by the core load during execution.
20	Reserved	
21-24	Cartridge ID	The ID of the cartridge that contains the mainline program in its Working Storage (valid only if no name is specified in columns 8-12; blanks in this field indicate the System Working Storage).
25	Not used	
26	LOCAL call LOCAL indicator	A punch in this column enables a LOCAL program to call another LOCAL. Without a punch, a LOCAL cannot call another LOCAL.
27	Not used	
28	Special ILS indicator	A punch in this column indicates that ILSs for this core load should be chosen from the special ILSs.
29-80	Not used	

// PAUS

This control record causes the Supervisor to WAIT. When PROGRAM START is pressed, the Supervisor continues processing Monitor control records from the input stream.

The format of the PAUS control record is as follows.

Card Column	Contents	Notes
1-7	//bPAUS	
8-80	Not used	

// TYP

This control record causes the Supervisor to temporarily assign the Keyboard as the principal input device. The Keyboard replaces the card or paper tape reader as the principal input device until a TEND control record is entered from the Keyboard.

The format of the TYP control record is as follows.

Card Column	Contents	Notes
1-6	//bTYP	
7-80	Not used	

With the Keyboard as the principal input device, the keyboard functions are identical to those discussed for TYPEZ and TYPE0 (System Library Subroutines) with one exception. The END-OF-MESSAGE character causes the rest of the buffer to be filled with blanks. Therefore, at the completion of a new message, nothing will remain of any previously - entered message.

// TEND

This control record causes the Supervisor to reassign the card or paper tape reader as the principal input device. The reassignment is to whichever unit was the principal device prior to the detection of a TYP control record.

The TEND control record must be entered from the Keyboard. The format of the TEND control record is as follows.

Card Column	Contents	Notes
1-7	//bTEND	
8-80	Not used	

//EJECT

This control record causes the 1403 Printer or 1132 Printer, whichever is the principal print device, to skip to a new page and print the page header. When Console Printer is assigned as principal printer, or a // CPRNT record has been processed a space of 5 lines and printing of page header will be performed. Control is then returned to the Supervisor, which reads the next record in the input stream.

The EJECT control record itself is printed.

Card Column	Contents	Notes
1-8 9-80	//bEJECT Not used	

//* (comments)

This control record allows the user to print alphameric text on the listing printed on the principal print device by the Supervisor and DUP. The Supervisor and DUP simply print the control record and continue reading control records from the input stream. The Comments control record may not immediately follow an XEQ, ASM, RPG, or FOR control record.

When Supervisor uses Console Printer to print records and the principal printer is another printer, DUP is going to print the comments on the principal printer and Supervisor on the Console Printer.

The format of the Comments control record is as follows.

Card Column	Contents	Notes
1-4 5-80	// b* User comments	Any alphameric characters may be used.

// CPRNT

This control record causes the Supervisor to print all Monitor and Supervisor control records that it reads on the Console Printer. Printing by all other Monitor programs will be on the principal print device.

Once the CPRNT control record has taken effect, all Monitor and Supervisor control records will be printed as described above. To return the printing of Monitor and Supervisor control records to the principal print device, a reload function must be performed by the System Loader to redefine the principal print device.

This control record causes the // EJECT record to affect the Console Printer rather than the principal printer.

The format of the CPRNT control record is as follows.

Card Column	Contents	Notes
1-8 9-80	//bCPRNT Not used	

SUPERVISOR CONTROL RECORDS

The control records described below (LOCAL, NOCAL, FILES, EQUAT and G2250) are used by the Core Load Builder to:

- Provide for subprogram overlays during execution (LOCAL)
- Include subprograms not called in the core load (NOCAL)
- Equate disk storage files defined in the mainline program during compilation or assembly to specific files stored on the disk (FILES)
- Provide Graphic Display Capabilities (G2250)
- Substitute a subroutine with another subroutine

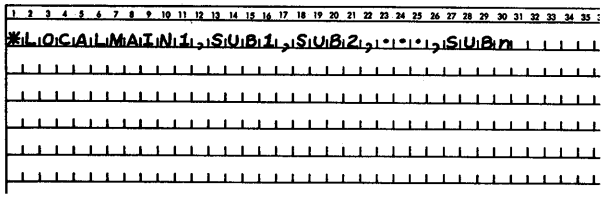
The first four control records are placed in the input stream following an XEQ Monitor control record that names a mainline program stored in Disk System format or following a STORECI control record. In either case the control records are written on disk in the Supervisor Control Record Area (SCRA), from which the Core Load Builder reads them for processing.

The fifth type of control record (EQUAT) is placed after a JOB Monitor control record and maintains its function until the next Job Monitor control record is met. Supervisor will read the EQUAT control records and write them on disk in the SCRA, from which the Core Load Builder reads them for processing.

Up to 99 Supervisor control records may follow the XEQ or STORECI control record. There is no specified order (by type) to be followed; however, the types may not be intermixed.

*LOCAL

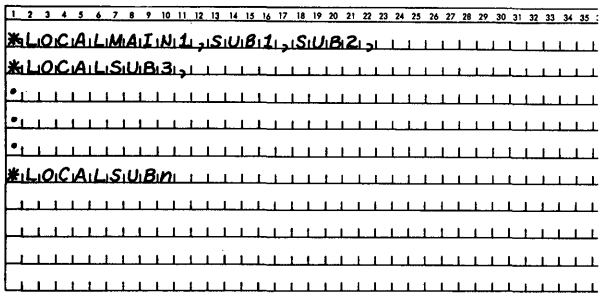
LOCAL (load-on-call) subprograms are subprograms specified by the user to be read, one at a time, as they are called during the execution, into a LOCAL overlay area. The LOCAL subprograms are specified on the LOCAL control record as follows:



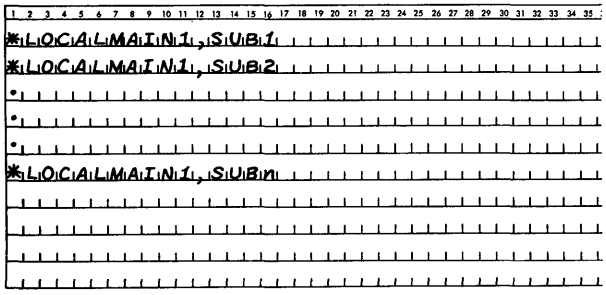
where

MAIN1 is the name of the mainline program already stored on disk. SUB1 through SUBn are the names of the LOCAL subprograms used with that mainline program.

In the case illustrated below, all the LOCAL control records except the last end with a comma (continuation character) and the mainline program name appears on the first LOCAL control record only.

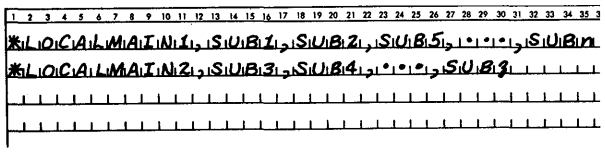


The same results would have been obtained if the records had been:



All the LOCAL subprograms for each mainline program in an execution must be specified on the LOCAL control records that follow the XEQ Monitor control record initiating the execution.

Separate LOCAL control records must be used for each mainline program in the execution that calls LOCAL subprograms. For example,

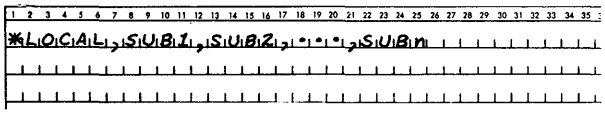


where

MAIN2 is a link called by MAIN1.

If the mainline program is to be executed from Working Storage, the mainline program name must be omitted from the LOCAL control record.

For example,



This LOCAL control record format must be used if LOCALs are to be specified with the DUP operation STORECI.

No embedded blanks are allowed in the LOCAL control record.

*NOCAL

NOCAL (load-although-not-called) subprograms are subprograms specified by the user to be included in the core load, even though they are not called. They are specified on NOCAL control records using the same format that applies to LOCAL control records except that *NOCAL is used in place of *LOCAL.

Rules for LOCAL and NOCAL Usage

The user must observe the following rules in the usage of LOCAL and NOCAL control records:

- A subprogram cannot be specified as a LOCAL subprogram if it causes another subprogram, also specified as a LOCAL subprogram in the same mainline program, to be called. For example, if A calls B and B calls C, and A is a LOCAL subprogram, neither B nor C can be specified as a LOCAL subprogram for the same mainline program. This restriction can be avoided by using the LOCAL-calls-LOCAL option (see "// XEQ control record" and "Programming Tips and Techniques").

- If a subprogram is specified as a LOCAL subprogram and system overlays (SOCALs) are employed, the subprogram is made a LOCAL subprogram, even if it would otherwise have been included in one of the SOCALs.

- If a subprogram is specified as a LOCAL subprogram, it is included as a LOCAL subprogram in the core image program even if it is not otherwise called.

- The information on all the LOCAL control records for an execution may not exceed M+2(C+1), where M is the number of mainlines and C is the number of commas. This restriction also applies to NOCAL control records.

- Only subprograms types 3, 4, 5 and 6 can be named on LOCAL and NOCAL control records. Subprogram types 3 and 5 are referenced by LIBF statements, types 4 and 6 with CALL statements. Types 5 and 6 are ISSs; types 3 and 4 are subprograms. See Appendix C for a description of subprogram types.

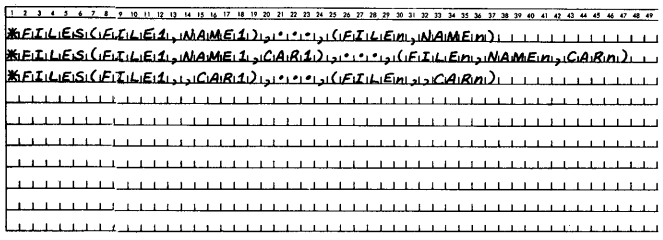
- Conversion tables, e.g., EBPA, HOLTB, may not be used as LOCALs.

*FILES

By means of FILES control records the file numbers specified in FORTRAN DEFINE FILE statements or in Assembler FILE statements are equated to the names of Data Files stored in the User and Fixed areas. FILES control records may also be used to define Data Files in Working Storage other than the master cartridge. All the User/Fixed Area files to be used by all the core loads in an execution must be defined in the FILES control records following the XEQ Monitor control record initiating the execution. All the files thus defined are available to each core load in the execution.

When Data Files are equated in a program stored in DCI, successful execution of this program requires that all cartridges on which these files are stored must be in the same condition and on the same logical drives as when the STORECI occurred. This is necessary since the Core Load Builder places an absolute sector address, including the drive code, into the file table for each equated file.

The format of the FILES control record is as follows.



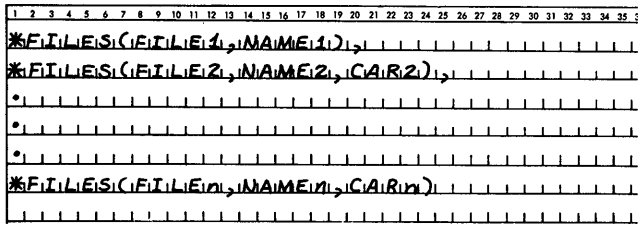
where

FILE1 through FILEn are the file numbers specified in the FORTRAN DEFINE FILE statements or Assembler FILE statements.

NAME1 through NAMEn are the names of Data Files already stored on disk. If the name is omitted (2 commas are required in the control record format), the file is placed in Working Storage on the specified cartridge.

CAR1 through CARn are the IDs of the cartridges on which the respective Data Files are found. If the cartridge ID is omitted, it is assumed that the corresponding Data File has been defined on the master cartridge.

Continuation of FILES control records may be indicated by a comma following the last file definition on the control record, as follows:



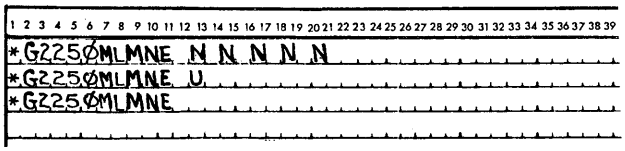
The continuation comma may appear only immediately after a right parenthesis.

No more than 159 files may be equated during an execution.

No embedded blanks are allowed in the FILES control record.

*G2250

G2250 is the name of the supervisor control record which is used to give the user graphic capabilities. The G2250 control record causes the Graphic Subroutine Package (GSP) communication module (GCOM) to be included in the core load immediately following the mainline program. (See IBM 1130/2250 Graphic Subroutine Package For Basic FORTRAN IV, Form C27-6934, for instructions on properly loading the mainline program.) Other supporting subroutines are also loaded into this area depending on the arguments described below. The format of the G2250 control record is:



where MLMNE is the mainline program to be executed. If the mainline program is executed from working storage, the mainline name must be omitted. The 1130/2250 user must enter a G2250 control record for each program using GSP to which he desires to link.

Card Column	Contents	Function
1-11	*G2250MLMNE	Specifies graphic support is required for the named mainline program. Loads GCOM immediately after the mainline program.
13	U	The character stroke subroutine containing upper case, numeric, and special characters is loaded.
	blank	The character stroke subroutine containing upper case, lower case, numeric, and special characters is loaded.
	N	No character stroke subroutine is loaded.
15	blank	The scissoring subroutine is loaded.
	N	The scissoring subroutine is not loaded.
17	blank	The ICA area expansion subroutine is loaded.
	N	The ICA area expansion subroutine is not loaded.
19	blank	The index controlled entity subroutine is loaded.
	N	The index controlled entity subroutine is not loaded.
21	blank	The level controlled direct entry subroutine is loaded.
	N	The level controlled direct entry subroutine is not loaded.

Information concerning the use of GSP subroutines as LOCALs and core storage layout requirements can be found in IBM 1130/2250 Graphic Subroutine Package For Basic FORTRAN IV, Form C27-6934.

*EQUAT

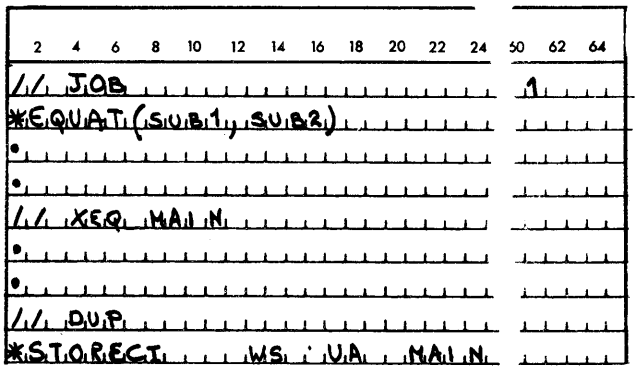
With the EQUAT control record the user specifies that subroutines will be substituted during the building of a core load. The format of the control record is as follows:

*EQUAT(SUB1,SUB2), ... (SUBN,SUBM)

where

SUB1 is the name of a subroutine which the Core Load Builder replaces with the subroutine SUB2 during the building of a Core Load.

Note that in the example given below the substitution of SUB2 for SUB1 is also accomplished in the *STORECI operation.



The EQUAT control record may also be used to substitute symbloic names in DSA statements. (Assembler programs only.) In this instance what has been said above concerning subroutine names is also applicable to symbolic names in DSA statements.

More than one control record can be used so long as the exact number of control records used is punched in the Job Monitor Control (See // Job).

In the Programming Tips and Techniques section, information is found on how the EQUAT function is used. (See USE OF THE EQUAT RECORD).

SUPERVISOR CORE DUMP PROGRAM

The DUMP program provides the user with a hexadecimal printout of the contents of core storage. The calling sequences for the DUMP and PDMP statements are contained in the Assembler language manual (Form C26-5927). FORTRAN programs access the DUMP Program through the FORTRAN statement CALL PDUMP (See FORTRAN language manual, Form C26-3715).

Terminal and Dynamic Dumps

The DUMP entry point ($DUMP) in the Skeleton Supervisor (and thus the DUMP program in the Supervisor) can be entered (1) by a BSI to the DUMP entry point, (2) by a manually executed transfer to the DUMP entry point plus 1, or (3) by a branch to location zero, which contains an MDX to $DUMP+1.

When the DUMP entry point is entered, a dump of the area of core storage bounded by the limit parameters is given in hexadecimal format. Execution of the core load in progress then resumes at the location following the last parameter of the call to the DUMP entry point.

When $DUMP+1 is entered, a dump of the entire contents of core storage is given in hexadecimal format. The DUMP program then executes a CALL EXIT, thereby terminating the execution of the core load in progress.

A portion of a core dump is printed below.

ACCUMULATOR 4000 EXTENSION 78D3 XR1 7FA0 XR2 78D3 XR3 0000 OVERFLOW OFF CARRY OFF

ADDR	***0	***1	***2	***3	***4	***5	***6	***7	***8	***9	***A	***B	***C	***D	***E	***F
0000	703F	FFFB	0000	0000	0FFA	0140	0080	0000	7AED	7C56	00B3	0000	00C4	0091	8C00	0000
0010	0000	5540	FFFF	C000	0327	0008	CC01	7FA0	D900	703F	4000	78D3	00F2	7400	00EE	70FD
0020	4DC0	C002	4400	C0F2	7400	C0EE	70FD	70F4	7B3F	3000	4C80	0028	003F	0150	0CC0	0000
0030	0000	0000	C001	0000	0000	0C00	C000	000C	7014	0CC0	1810	7012	0001	0004	FFFF	0000
0040	7400	0032	70FD	D8D6	69D2	C480	003F	D0D1	C8F3	4400	00F2	C0F0	7001	C0F0	D0C7	7400
0050	0032	70FD	C8BD	6580	0039	C101	18D0	C100	D8BB	6500	01DC	C0FE	1890	4400	00F2	7400
0060	00EE	70FD	4102	0000	C000	0000	0000	0000	0000	0C00	C000	0000	CC00	0000	C000	0000
0070	C000	0000	0000	0000	0000	0000	FFFF	0000	0802	0C01	0000	C000	0000	0000	0000	0000
0080	0000	0000	3000	4C80	0081	0000	3000	4C80	0085	CC00	3000	4C80	0089	0C00	3000	4C80
0090	008D	C000	3000	4CC0	C091	01D6	CC00	0000	0000	0C00	0140	0000	0000	0000	0C00	9C00
00A0	0000	C000	0000	CC00	0658	0658	0658	0000	0000	0C00	CC00	0000	C000	0000	0CC0	0000
00B0	0C00	0000	0000	0051	6906	6A07	2807	D80A	4400	0CF7	6500	7FA0	6600	78D3	2000	C802
00C0	4CC0	00B3	C001	9400	002A	D818	280E	690F	6A10	0816	1002	4C10	00D0	4480	002C	FFFE
00D0	6109	0810	1149	4580	7FE8	2C00	6500	7FA0	6600	78D3	C803	4CC0	00C4	4001	4000	78D3
00E0	0200	CFC0	0000	0300	0000	C000	C000	0000	0000	0CC0	C000	0000	0C00	0C00	0011	0000
00F0	00EF	FF6A	004B	7400	00EE	70FD	7002	00BA	7015	69CF	6A10	1008	D03C	18D0	D05B	6211
0100	6AED	C0F0	D0F4	7053	4C00	01BE	6908	081E	6500	7FA0	6600	78D3	4C80	00F7	6500	0004
0110	6600	00F2	0819	D0C9	485C	70EE	C80D	D900	74FF	00EE	703E	C812	C014	4293	1810	D480
0120	0198	70DD	0001	0140	0FFA	0140	C004	95C0	0004	95C0	0122	9600	9400	9781	0EBA	0141
0130	5002	5004	FEC0	0001	0080	C600	0008	5000	0FF8	01CC	0701	0007	000A	009F	FFFB	9680
0140	0400	0141	0000	FFFF	CC00	0000	1810	D0A6	74FF	0032	1000	70BC	C0E3	70CF	C0E8	4400
0150	0028	703A	C0D9	18D0	C101	1803	704D	7401	0032	6500	C004	C900	D8C7	D8D0	1810	1084
0160	DC0E	80DB	D01B	80DA	D033	80D6	8008	80C7	D006	62FD	698D	C101	E0CB	D101	9400	00A4
0170	4828	7006	C101	80C2	7401	C16F	7201	70F5	6600	00F2	C23D	E249	D250	C400	009F	EA4E
0180	D23A	EA43	D239	EA50	9247	D237	EA42	8247	D24D	EA48	D23B	CA3C	0A3A	D2EB	4828	70BC
0190	1C02	4828	70B8	100B	4828	70BC	C101	9400	009A	4818	7014	1893	180F	1002	EA3A	18D0
01A0	4810	7002	F251	8230	DA34	4213	CA38	DA34	4213	C231	D480	0198	9101	4C20	0116	CA3C
01B0	4808	7094	8A40	DA3C	4830	1810	824F	D100	CA36	DA34	C101	EA50	D101	4213	C24D	D235
01C0	C247	4820	4213	CA32	D900	C23C	4808	70E9	7500	0140	C900	DA32	CA3C	D900	708F	0000
01D0	0C00	C000	0000	C000	0000	0000	C000	0000	0000	0C00	C0A0	3333	016E	0100	03C0	001C
01E0	4480	7DB9	0006	6780	7FFC	18A0	DF00	0142	633C	6FCC	7925	6300	C193	4C28	01F1	6780
01F0	7F33	6F00	045A	6F00	02C0	C120	4C20	0307	C11D	4C28	02B3	C700	7F70	4C18	02E7	D400
0200	0395	4400	02A9	C302	1804	4C20	020C	CC00	0342	DC00	7DA4	7004	CC00	033E	DC00	7DA4
0210	4480	7DB8	4480	7DB5	4480	7DB5	4480	7DB5	4480	7DB5	4480	7DB5	C400	0386	4418	03E7
0220	4400	044C	6600	035A	6317	4079	4480	7DBB	7925	405A	6600	7925	6780	7FFC	C302	4480
0230	7DB0	C928	DA01	C303	4480	7DB0	C928	D206	18D0	D207	C305	4480	7DB0	C928	DA0D	C306
0240	4480	7DB0	C928	DA13	4480	7DBB	7925	4480	7DB5	4480	7DB5	4480	7DB5	4036	6600	0371
0250	4C3F	6600	C37B	403C	4480	7DB5	6215	6E00	039F	C400	0398	D400	039C	6680	7FFC	7206
0260	6A5D	7201	6A13	1810	D400	C39D	4400	03A7	1000	CC00	0390	D916	4480	7DB7	C4C0	039D
0270	4C18	02BD	4480	7DBB	7925	6600	7792	C202	8400	039C	D400	039C	7203	74FF	039F	7001
0280	703C	D400	0398	70DE	02EC	613C	C008	D500	7925	71FF	70FC	6500	7FA0	4C80	0284	4040
0290	0254	6105	630A	400B	740C	02A4	71FF	70FA	C003	D00A	4480	7DBB	7925	4C80	0290	02F7
02A0	6A01	C700	032F	D700	7925	73FF	70FA	4C80	029F	0203	6780	7FFC	CC00	0394	DB00	4480
02B0	7DB3	4C80	C2A9	C700	7F6B	1004	1804	4C18	02D9	C700	7F6B	4C00	01FF	C400	7788	6700
02C0	0004	4C18	02D9	C700	7F6B	1004	1804	9480	02BE	D400	0386	C600	00FB	8400	039B	D400
02D0	039B	C7C0	7F70	180C	100C	EC80	02BE	4C00	01FF	C193	4C10	02E7	7301	6F00	0336	74FC
02E0	0336	7C05	1810	D400	0386	4C00	01F1	4480	7DB5	4480	7DB5	4098	C11D	4C28	02FF	C84A
02F0	D845	C84A	D845	6600	032F	630A	40A8	4480	7DBB	7925	C845	DC00	7DA4	4480	7DBD	C83E
0300	D835	C83E	D835	66C0	032F	630A	70EF	6827	4400	044C	6700	7925	6680	7FB4	72C1	C202
0310	4C20	0315	C01C	72FD	70FA	C019	4C20	031B	C112	9202	D112	1810	D400	039D	C112	D400
0320	039B	4400	03A7	C40C	039D	4C18	02E7	4480	7DBB	7925	7203	C202	4C20	02E7	70F2	0000
0330	40C5	D5C4	40D6	C640	C4E4	D4D7	C6D3	C5E3	4040	4040	D3C5	E361	C6D3	C5E3	C6D3	C5E3
0340	4040	4040	D3C5	E340	7BC3	C9C4	D540	4040	5BC6	D7C1	C440	4040	7BC6	D7C1	C440	4040
0350	7BC3	C9C2	C140	4C40	7BE4	D3C5	E340	4040	7BC6	D3C5	E340	E2C3	E3D9	40D5	D64B	4040
≈																≈
7EB0	4480	7DB3	7A14	D8C8	280F	690C	6A09	6B06	6500	7FA0	4C80	7EB2	7E4F	6700	0000	6600
7EC0	7926	6500	7FA0	C8B8	2001	4C80	7EBC	435E	4480	7DBA	0100	C0FB	D002	C0B9	7005	435E
7ED0	4480	7DBA	0200	C0B4	D0AC	6BAD	C301	E0B3	D115	4820	C300	4C30	7EE0	4480	7DB4	005C
7EE0	80AC	18D0	1010	A8AA	8115	90A9	4C08	7EEB	4480	7DB4	005D	C896	4400	00F2	7400	00EE
7EF0	70FD	C091	D116	8300	D117	4480	7DB7	C12E	4C98	7ECF	1010	D12E	4F00	0002	70FD	C08D
7F00	D116	8300	D117	4480	7DB7	C12E	4C98	7EDD	1010	D12E	4F00	0002	0000	0000	0000	0000
7F10	0000	0000	0000	C000	0000	0000	0000	0000	0000	0CC0	0000	0000	0000	0000	0070	0001
7F20	0000	0CC0	0000	C000	C406	3040	0020	0000	0200	0000	C000	0000	0CC0	FFF6	0C00	0000
7F30	0000	0C00	0000	C000	0000	0000	0000	C000	C000	0C01	0001	010A	7800	0000	0000	0000
7F40	0000	CCC0	C000	1D52	CCC0	0000	0C00	0000	1D52	0C00	0000	0000	0000	01D6	0C00	0000
7F50	0000	C000	C000	2222	3333	000F	0ABB	3333	0000	0000	0000	0000	0140	0000	0000	0000
7F60	0000	0110	1110	0000	C000	CC00	FFFE	0000	0CC0	0C00	C000	0118	0000	0000	0000	C000
7F70	0150	0000	0000	0000	0000	0020	CC00	0000	0000	0000	000C	0009	0C00	0000	0C00	0000
7F80	0000	C000	C000	0000	C000	CC00	0000	0000	0000	0C00	0000	0000	0000	0000	0C00	0000
7F90	05A3	C00B	C56B	C010	03C0	0015	0461	0018	03C0	001C	05A3	001F	05A3	0024	0500	0029
7FA0	05F8	0030	023D	0035	023D	0037	C248	0039	0000	C000	0000	0000	0000	0000	0000	0000
7FB0	0C00	0000	0000	0000	0000	CC00	78D3	7DBB	0000	C000	CC00	0000	0000	0000	CCC0	0000
7FC0	0CC0	0000	C000	0000	0000	0000	0000	0000	0000	CCC0	C000	0000	0000	0000	0000	0000
7FD0	C000	0000	0000	0000	0000	C000	0C00	0000	0000	0C00	C004	056B	0461	7DFD	003F	0000
7FE0	7C64	0000	0000	0000	C000	0000	0000	0000	008D	008D	008D	008D	008D	7C56	7AEA	7AEA
7FF0	0000	0000	00A0	C0F2	78D3	7DBB	0802	78D3	11DE	7D91	7A06	0640	7782	7925	79B5	7963

DISK UTILITY PROGRAM (DUP)

The Disk Utility Program (DUP) provides the user with the ability to perform the following operations through the use of control records.

- Store Disk System Format (DSF) programs, Disk Core Image (DCI) programs, and Data Files on the disk
- Make the DSF programs, DCI programs, and Data Files on the disk available in printed, punched card, or punched paper tape output
- Remove DSF programs, DCI programs, and Data Files from the disk
- Determine the status of disk storage through a printed copy of LET/FLET
- Modify the system
- Perform other disk maintenance functions

DUP control records are described in the section of this manual entitled DUP Control Records. DUP error messages are listed in Appendix A.

GENERAL FLOW

DUP is called into operation when the Supervisor recognizes a DUP Monitor control record (// DUP). The control portion of DUP is brought into core to read the next record from the input stream, which should be a DUP control record (*...). The DUP control record is printed and analyzed. LET is searched for the program specified, and switches and indicators are set in accordance with the information obtained from the control record. The DUP program required to perform the requested operation is then read into core from the disk and given control.

The DUP program performs its assigned tasks, directed by the switches and indicators that were set according to the information on the DUP control record. Upon completion of its tasks, the DUP program prints a message and returns control to the control portion of DUP. The control portion indicates the completion of the DUP operation with a printed message and reads the next record from the input stream.

If the record read is a Monitor control record other than comments, control is returned to the Supervisor to process the record. If the record read is a DUP control record, DUP maintains control and reads the next record. Comments Monitor control records are simply printed; blank records are passed.

INFORMATION TRANSFER AND FORMAT CONVERSION

Table 2 summarizes the DUP operations that transfer information from one area or medium to another area or medium. In addition, the format conversions made during the transfers of information are shown. The acronyms for the various formats are described below. The formats are described in Appendix C.

Acronym	Format
DSF	Disk System Format
DDF	Disk Data Format
DCI	Disk Core Image Format
CDS	Card System Format
CDD	Card Data Format
CDC	Card Core Image Format
PTS	Paper Tape System Format
PTD	Paper Tape Data Format
PTC	Paper Tape Core Image Format
PRD	Printer Data Format

The user is advised to pay particular attention to Table 2 when performing save/restore operations, e.g., dumping to cards and later using the cards to store the information back on the disk. Note that there may be more than one way to dump and store data/programs, as in dumping a DCI program to cards and later storing it back to disk.

ALTERING LET/FLET

The two tables LET and FLET constitute a directory to the contents of the User and Fixed areas on disk. The allocation of disk storage and, correspondingly, the contents of LET/FLET can be altered by the user only through the use of DUP.

Before storing any DSF program, DCI program, or Data File, DUP searches LET/FLET to ensure that the name of the DSF program, DCI program, or Data File does not already appear in LET/FLET on the cartridge specified on the DUP control record. (If no cartridge is specified, the LET/FLET of every cartridge specified on the last JOB record is searched.) Disk storage is allocated to the DSF program, DCI program, or Data File and a corresponding entry is made in LET/FLET only if the name is not found.

When dumping or deleting a DSF program, DCI program, or Data File from the User/Fixed Area, the DSF program, DCI program, or Data File is located through LET/FLET using the name specified by the user in the DUP control record.

A LET/FLET printout and description is contained in Appendix G.

DUP CONTROL RECORDS

DUP control records call IBM-supplied programs that perform operations involving the disk such as storing, moving, deleting, and dumping data and/or programs.

DUP control records generally follow the format described below. Note that all fields in the control

Table 2. Summary of DUP Data Transfer Operations*

"FROM" Area Symbols, with Formats		"TO" Area Symbols, with Formats														
		UA			FX		WS			CD			PT			PR
		DSF	DDF	DCI	DDF	DCI	DSF	DDF	DCI	CDS	CDD	CDC	PTS	PTD	PTC	PRD
UA	DSF						DUMP	DUMPDATA		DUMP	DUMPDATA		DUMP	DUMPDATA		DUMP DUMPDATA
	DDF							DUMP DUMPDATA			DUMP DUMPDATA			DUMP DUMPDATA		DUMP DUMPDATA
	DCI							DUMPDATA	DUMP		DUMPDATA	DUMP		DUMPDATA	DUMP	DUMP DUMPDATA
FX	DDF							DUMP DUMPDATA			DUMP DUMPDATA			DUMP DUMPDATA		DUMP DUMPDATA
	DCI							DUMPDATA	DUMP		DUMPDATA	DUMP		DUMPDATA	DUMP	DUMP DUMPDATA
WS	DSF	STORE STOREMOD	STOREDATA	STORECI	STOREDATA	STORECI				DUMP	DUMPDATA		DUMP	DUMPDATA		DUMP DUMPDATA
	DDF		STOREMOD STOREDATA		STOREMOD STOREDATA						DUMP DUMPDATA			DUMP DUMPDATA		DUMP DUMPDATA
	DCI		STOREDATA	STOREMOD STOREDATAC	STOREDATA	STOREMOD STOREDATACI					DUMPDATA	DUMP		DUMPDATA	DUMP	DUMP DUMPDATA
CD	CDS	STORE	STOREDATA	STORECI	STOREDATA	STORECI	STORE	STOREDATA								
	CDD		STOREDATA	STOREDATACI	STOREDATA	STOREDATACI		STOREDATA	STOREDATACI							
	CDC		STOREDATA	STOREDATACI	STOREDATA	STOREDATACI		STOREDATA	STOREDATACI							
PT	PTS	STORE	STOREDATA	STORECI	STOREDATA	STORECI	STORE	STOREDATA								
	PTD		STOREDATA	STOREDATACI	STOREDATA	STOREDATACI		STOREDATA	STOREDATACI							
	PTC		STOREDATA	STOREDATACI	STOREDATA	STOREDATACI		STOREDATA	STOREDATACI							

* For this chart DUMPDATAE and STOREDATAE are the same as DUMPDATA and STOREDATA respectively.

record except the count field are always left-justified and that unless otherwise stated, all fields are required.

Column 1. Column 1 always contains an *(asterisk).

Operation Field. Columns 2 through 12 (21 in the case of the DEFINE operation) contain the name of the desired DUP operation. Columns 2 through 6 identify the basic operation (STOREDATACI); columns 7 through 12 (or 21) identify the extended operation (STOREDATACI). Where shown in the control record format, a blank character (b) is required within or following the operation name.

FROM and TO Fields. Columns 13 and 14 contain the "FROM" symbol, that is, the symbol specifying the disk area or I/O device from which information is to be obtained (the source). Columns 17 and 18 contain the "TO" symbol, that is, the symbol specifying the disk area or I/O device to which information is to be transferred (the destination). The symbols that must be used as the "FROM" and "TO" symbols are shown below.

Symbol	Disk Area or I/O Device
UA	User Area, Disk
FX	Fixed Area, Disk
WS	Working Storage, Disk
CD	Card I/O device. If the 1134 has been defined as the principal input device, CD is equivalent to PT.
PT	Paper Tape
PR	Principal print device

When used, the symbols UA, FX, and WS each specify an area on disk but do not identify the cartridge on which the area is found.

Name Field. Columns 21 through 25 contain the name of the DSF program, DCI program, or Data File involved in the specified DUP operation. The name may consist of up to five alphameric characters, and must be left-justified within the field. The first character must be alphabetic (A-Z, $), and no embedded blank characters are allowed.

When referencing a DSF program, DCI program, or Data File already stored on disk, the name must be an exact duplicate of the LET/FLET entry.

Count Field. Columns 27 through 30 contain the count. The count is always a right-justified decimal integer. The count field is defined in the individual control record formats for those operations that require it.

FROM and TO Cartridge ID Fields. Columns 31 through 34 contain the cartridge ID of the cartridge containing the disk area from which information is to be obtained, that is, the "FROM" (source) cartridge ID. Columns 37 through 40 contain the cartridge ID of the cartridge containing the disk area to which information is to be transferred, that is, the "TO" (destination) cartridge ID.

Either or both of these cartridge IDs may be omitted. If a cartridge ID is omitted, and the corresponding FROM or TO field is the User or Fixed Area, a search is made of the LET/FLET on each cartridge specified on the JOB record, starting with the cartridge on logical drive zero (the master cartridge) and continuing through logical drive four. If the corresponding FROM or TO field is Working Storage, then a default to System Working Storage is made. If a cartridge ID is specified, the LET/FLET on the specified cartridge only is searched, or System Working Storage is used.

Use of the "FROM" and "TO" cartridge IDs makes it possible for DUP (1) to transfer DSF programs, DCI programs, and Data Files from one cartridge to another without deleting them from the source cartridge, and (2) to operate on a DSF program, DCI program, or Data File even though the same name appears in the LET/FLET on more than one cartridge.

Unused Columns. All unused columns between columns 2 and 40 must be left blank. Columns 41 through 80 are ignored by DUP and are available for user's remarks.

DUP Operations and Control Record Formats

The following are descriptions of the various DUP operations. Each description consists of (1) a brief description of the processing performed, (2) a breakdown of the control record for the operation, and (3) a table of the transfers and format conversions possible in the operation.

*DUMP

The DUMP operation moves information from the User/Fixed Area on disk to Working Storage or makes information from the User/Fixed Area and Working Storage available as card, paper tape, or printed output. The print format is illustrated in Appendix C.

The movement of DSF programs from the User/Fixed Area to the output devices is accomplished in two phases; that is, the information is first moved to System Working Storage and then to the output device. Hence, information residing in Working Storage on the cartridge defined in the JOB Monitor control record by the Working Storage ID (see // JOB under Monitor Control Records) is destroyed during the DUMP operation. Data Files and DCI programs are moved directly from the User/Fixed Area to the output devices.

The number of disk blocks to be dumped is obtained from the LET/FLET entry, or, if the dump is from Working Storage, from the appropriate Working Storage Indicator in DCOM.

The format of the DUMP control record is as follows.

Card Column	Contents	Notes
1-6	*DUMPb	
7-12	Reserved	
13-14	"FROM" symbol	See chart below. If the dump is from Working Storage and the corresponding Working Storage Indicator is zero, an error message is printed (see DUP error messages, Appendix A).
15-16	Reserved	
17-18	"TO" symbol	See chart below. If the dump is to cards and if a 1442-6 or 1442-7 is utilized, each card is checked to see that it is blank before it is punched. If a non-blank card is read, the System will WAIT at $PRET with /100F displayed in the Accumulator after the appropriate error message has been printed (see DUP Error Messages, Appendix A).
19-20	Reserved	
21-25	Program name	The name is required except when the dump is from Working Storage to the printer.
26-30	Reserved	
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

The following chart is a summary of the information transfers and format conversions performed by DUMP.

Possible Sources, Including Formats	Possible Destinations, Including Formats
UA(DSF)	WS(DSF)
UA or WS (DSF)	CD(CDS) PT(PTS) PR(PRD)
UA or FX (DDF)	WS(DDF)
UA, FX, or WS (DDF)	CD(CDD) PT(PTD) PR(PRD)
UA or FX (DCI)	WS(DCI)
UA, FX, or WS (DCI)	CD(CDC) PT(PTC) PR(PRD)

*DUMPDATA

The DUMPDATA operation moves information from the User/Fixed Area on disk to Working Storage or makes information from the User/Fixed Area and Working Storage available in card, paper tape, or printed output. The print format is similar to that of DUMP (see Appendix C). The DUMPDATA operation differs from the DUMP operation in that the information, after transfer, is always in data format, and the amount of information transferred is dependent upon the count field of the DUMPDATA control record rather than the actual length of the program or data.

Information is moved directly from the User/Fixed Area or Working Storage to the output devices. The contents of Working Storage are not changed.

The count field (columns 27-30) in the DUMPDATA control record specifies the number of sectors to be dumped. This number of sectors is dumped regardless of the length of the DSF program, DCI program, or Data File, as indicated in the LET/FLET entry or in the Working Storage Indicator.

The format of the DUMPDATA control record is as follows.

Card Column	Contents	Notes
1-10	*DUMPDATAb	
11-12	Reserved	
13-14	"FROM" symbol	See chart below.
15-16	Reserved	
17-18	"TO" symbol	See chart below. If the dump is to cards, and if a 1442-6 or 1442-7 is utilized, each card is checked to see that it is blank before it is punched.
19-20	Reserved	
21-25	Program name	The name is required except when the dump is from Working Storage to the printer.
26	Reserved	
27-30	Count	The count (right justified, decimal) specifies the number of sectors to be dumped. The count overrides the contents of the Working Storage Indicator and the disk block count in the LET/FLET entry.
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

(continued)

The following chart is a summary of the information transfers and format conversions performed by DUMPDATA.

Possible Sources, Including Formats	Possible Destinations, Including Formats
UA(DSF)	WS(DDF)
UA or WS(DSF)	CD(CDD) PT(PTD) PR(PRD)
UA or FX (DDF)	WS(DDF)
UA, FX, or WS(DDF)	CD(CDD) PT(PTD) PR(PRD)
UA(DCI) or FX(DDF)	WS(DDF)
UA, FX, or WS(DCI)	CD(CDD) PT(PTD) PR(PRD)

*DUMPDATAbE

The DUMPDATAbE operation moves information from the User/Fixed Area on disk to Working Storage or makes information from the User/Fixed Area and Working Storage available in card or printed output. The DUMPDATAbE operation to an output device differs from the DUMPDATA operation in that the information on disk, which is assumed to be in packed EBCDIC form, 40 words per 80 card columns, is converted to card image format. Thus, the print-out is one line per source card, 80 positions, and the card output is an exact, full 80 column duplicate of the input cards in the corresponding STOREDATAE operation.

If the destination is Working Storage, no conversion takes place.

Information is moved directly from the User/Fixed Area or Working Storage to the output devices. The contents of Working Storage are not changed.

The count field (columns 27-30) in the DUMPDATAbE control record specifies the number of sectors to be dumped. This number of sectors is dumped regardless of the length of the Data File, as indicated in the LET/FLET entry or in the Working Storage Indicator.

The format of the DUMPDATAbE control record is the same as that of DUMPDATA except that col. 11 contains an E.

The following chart is a summary of the information transfers performed by DUMPDATAbE.

Possible Sources	Possible Destinations
UA or FX	WS
UA, FX, or WS	CD PR

*DUMPLET

The DUMPLET operation prints the contents of LET on the principal print device. In addition, the contents of FLET are also printed on the principal print device if a Fixed Area has been defined by the user.

If the name of a DSF program, DCI program, or Data File is specified in the DUMPLET control record, only the LET/FLET entry corresponding to that name is printed. If a cartridge ID is specified in the control record, the LET/FLET on only that cartridge is printed. If neither name nor cartridge ID are specified, the entire contents of both LET and FLET on each cartridge specified on the JOB record are printed. A sample LET/FLET dump and description appears in Appendix G.

The format of the DUMPLET control record is as follows.

Card Column	Contents	Notes
1-8	*DUMPLET	
9-20	Reserved	
21-25	Program name	Use of the name specifies that the LET/FLET entry for that name only is to be printed.
26-30	Reserved	
31-34	"FROM" cartridge ID	If an ID is specified, the LET/FLET on that cartridge only is printed.
35-80	Not used	

DUMPFLET

The DUMPFLET operation prints the contents of FLET on the principal print device.

If the name of a DCI program or Data File is specified in the DUMPFLET control record, only the FLET entry corresponding to that name is printed. If a cartridge ID is specified in the control record, the FLET on that cartridge only is printed. If neither name nor cartridge ID are specified, the entire contents of the FLET on each cartridge defined on the JOB record are printed. A sample LET/FLET dump and description appears in Appendix G.

The format of the DUMPFLET control record is as follows.

Card Column	Contents	Notes
1-10	*DUMPFLETb	
11-20	Reserved	
21-25	Program name	Use of the name specifies that the FLET entry for that name only is to be printed.
26-30	Reserved	
31-34	"FROM" cartridge ID	If an ID is specified, the FLET on that cartridge only is printed.
35-80	Not used	

*STORE

The STORE operation moves information from Working Storage to the User Area or accepts information from the input devices and moves it to Working Storage or the User Area.

All movement of information from the input devices to the User Area is accomplished in two phases; that is, the information is first moved to the System Working Storage and then to the User Area. Hence, information residing in Working Storage on the cartridge defined in the JOB Monitor control record by the Working Storage ID (see // JOB under Monitor Control Records) is destroyed during the STORE operation.

Since the User Area and Working Storage are adjacent areas, and since the User Area expands as needed into what had been Working Storage, DUP assumes that on any STORE operation to the User Area, the contents of that Working Storage are destroyed. Therefore, the appropriate Working Storage Indicator is reset to zero following the STORE operation to the User Area.

DUP makes the required LET entry (or entries) for each program stored. A LET entry is made for each entry point in the program. DUP supplies the disk block count required in the LET entry for the primary entry point.

The format of the STORE control record is as follows.

Card Column	Contents	Notes
1-6	*STORE	
7-10	Reserved	
11	Subtype (for type 3, 4, 5 and 7 subprograms only	See "System Overlays" under Core Load Builder.
12	Reserved	

(continued)

Card Column	Contents	Notes
13-14	"FROM" symbol	If the STORE operation is from Working Storage and the corresponding Working Storage Indicator is zero, an error message is printed (see DUP Error Messages, Appendix A).
15-16	Reserved	
17-18	"TO" symbol	See Chart below
19-20	Reserved	
21-25	Program Name	The name is required except when the STORE operation is to Working Storage.
26-30	Reserved	
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

The following chart is a summary of the information transfers and format conversions performed by STORE.

Possible Sources, Including Formats	Possible Destinations, Including Formats
WS(DSF)	UA(DSF)
CD(CDS) PT(PTS)	UA or WS(DSF)

*STOREDATA

The STOREDATA operation moves information from Working Storage to the User/Fixed Area or accepts information from the input devices and moves it to Working Storage or the User/Fixed Area. DUP assumes that the input to the STOREDATA operation is in data format; the output from the STOREDATA operation is always in data format.

Information is moved directly from the input devices to the User/Fixed Area. The contents of Working Storage are not changed except that when storing to the User Area, the contents of Working Storage on that drive are destroyed since the User Area and Working Storage are adjacent areas.

DUP makes the required LET/FLET entry. The name specified on the STOREDATA control record is the name used to generate the LET/FLET entry and is the name that must be used in all subsequent references to the Data File. DUP supplies the disk block count required in the LET/FLET entry if the source is cards or paper tape. If the source is Working Storage, the sector count specified in the STOREDATA control record is used.

The format of the STOREDATA control record is as follows.

Card Column	Contents	Notes
1-10	*STOREDATA	
11-12	Reserved	
13-14	"FROM" symbol	See chart below.
15-16	Reserved	
17-18	"TO" symbol	See chart below.
19-20	Reserved	
21-25	Program name	The name is not required when the STORE operation is from cards or paper tape to Working Storage.
26	Reserved	
27-30	Count	If the source is Working Storage, the count is the number (decimal) of sectors of data to be stored. This count overrides the contents of the Working Storage Indicator. If the source is cards, the count is the number (decimal) of cards to be read. If the source is paper tape, the count is the number (decimal) of paper tape records to be read.
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

The following chart is a summary of the information transfers and format conversions performed by STOREDATA.

Possible Sources, Including Formats	Possible Destinations, Including Formats
WS(DSF, DDF, DCI)	UA or FX(DDF)
CD(CDS, CDD, CDC)	UA, FX, or WS(DDF)
PT(PTS, PTD, PTC)	UA, FX, or WS(DDF)

*STOREDATAE

The STOREDATAE operation moves information from Working Storage to the User/Fixed Area or accepts information from the card reader and moves it to Working Storage or the User/Fixed Area. The source cards are converted to packed EBCDIC format, that is two columns per word or 8 cards per sector. Thus, the input is assumed to be in the 1130 character set, and in the card code.

When the source is Working Storage, no conversion takes place.

Information is moved directly from the input device to the User/Fixed Area. The contents of Working Storage are not changed except that when storing to the User Area, the contents of Working Storage on that drive are destroyed since the User Area and Working Storage are adjacent areas.

DUP makes the required LET/FLET entry. The name specified on the STOREDATAE control record is the name used to generate the LET/FLET entry and is the name that must be used in all subsequent references to the Data File. DUP supplies the disk block count required in the LET/FLET entry if the source is cards or paper tape. If the source is Working Storage, the sector count specified in the STOREDATAE control record is used.

Note that the corresponding dump operation, DUMPDATAbE, transfers a whole number of sectors to cards. To avoid not wanted output, the number of cards stored should consequently be a multiple of 8 (blank cards can be added for that purpose).

The format of the STOREDATAE control record is the same as that of STOREDATA except that col. 11 contains an E.

The following chart is a summary of the information transfers performed by STOREDATAE.

Possible Sources	Possible Destinations
WS	UA or FX
CD	UA, FX, or WS

*STOREDATACI

The STOREDATACI operation moves information from Working Storage to the User/Fixed Area on disk or accepts information from the input devices and moves it to Working Storage or to the User/Fixed Area. If the input is from cards or paper tape, the STOREDATACI operation assumes the input format to be card or paper tape core image format. If the input is from Working Storage (the information has been previously dumped to Working Storage or stored in Working Storage from an input device), the appropriate Format Indicator must indicate Disk Core Image format (DCI); otherwise, no STORE operation is performed. The output from the STOREDATACI operation is always in Disk Core Image format.

All movement of information from the input devices to the User/Fixed Area is done directly; that is, the transfer is not made via Working Storage. Hence, the contents of Working Storage are not changed by the STOREDATACI operation when storing information from an input device to the Fixed Area. Note, however, that when storing to the User Area, the contents of Working Storage on that drive are destroyed.

DUP makes the required LET/FLET entry. The name specified on the STOREDATACI control record is the name used to generate the LET/FLET entry and is the name that must be used in all subsequent references to the core image program. DUP computes the disk block count required in the LET/FLET entry from the count specified in the STOREDATACI control record.

The format of the STOREDATACI control record is as follows.

Card Column	Contents	Notes
1-12	*STOREDATACI	
13-14	"FROM" symbol	See chart below.
15-16	Reserved	
17-18	"TO" symbol	See chart below.
19-20	Reserved	
21-25	Program name	If the STORE operation is to Working Storage, the name is not required.
26	Reserved	
27-30	Count	The count (right justified, decimal) is the number of records in the core image input. The count is not required if the source is Working Storage.
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

The following chart is a summary of the information transfers and format conversions performed by STOREDATACI.

Possible Sources Including Formats	Possible Destinations, Including Formats
WS(DCI)	UA or FX(DCI)
CD(CDC, CDD)	UA, FX, or WS(DCI)
PT(PTC, PTD)	UA, FX, or WS(DCI)

*STORECI

The STORECI operation obtains an object program from Working Storage or from an input device, converts it into a core image program using the Core Load Builder, and stores the core image program into the User/Fixed Area.

The Core Load Builder is fetched to build a core image program for the STORECI operation as if execution were to follow; that is, that portion of the core load residing above core location 4096 is placed in the System CIB, and LOCALs and/or SOCALs are placed in System Working Storage. The STORECI operation stores all these portions of the core image program into the "TO" (destination) area.

The DCI program stored in the User/Fixed Area includes the Transfer Vector built by the Core Load Builder; however, neither the disk I/O subroutine nor any COMMON area is included. Figure 5 shows the layout of a DCI program as it is stored in the User/Fixed Area. No scale is intended in this illustration.

DUP makes the required LET/FLET entry for the core image program as it is stored. The name specified on the STORECI control record is the name used to generate the LET/FLET entry and is the name that must be used in all subsequent references to the DCI program. DUP obtains the disk block count required in the LET/FLET entry from the Core Load Builder.

The format of the STORECI control record is as follows.

Card Column	Contents	Notes
1-8	*STORECI	
9	Disk I/O subroutine indicator	This column specifies the disk I/O subroutine to be loaded into core by the Core Image Loader for use by the core load during execution Indicator — Disk Subroutine 0, 1 — DISK 1 N — DISK N blank or Z — DISK Z all others — An error message is printed (see DUP Error Messages, Appendix A)
10	Reserved	
11	LOCAL-can call LOCAL indicator	A punch in this column enables a LOCAL program to call another LOCAL. Without a punch, this is not possible.
12	Special ILS indicator	A punch in this column indicates that ILSs for this core load should be chosen from the special ILSs (see also //XEQ).
13-14	"FROM" symbol	See chart below. If the STORE operation is from Working Storage and the corresponding Working Storage Indicator is zero, an error message is printed (See DUP Error Message, Appendix A).
15-16	Reserved	
17-18	"TO" symbol	See chart below.
19-20	Reserved	
21-25	Program name	
26	Reserved	
27-30	Count	The count is the number (decimal) of FILES, NOCAL, LOCAL, and G2250 control records that follow the STORECI control record. These records are read by DUP for use by the Core Load Builder before the STORE operation is performed. Note that the mainline program name must not be used on the G2250 control records. Data files named in FILES record must be in Fixed Area.
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80		

NOTE: The LOCAL-calls-LOCAL option is described in "Programming Tips and Techniques".

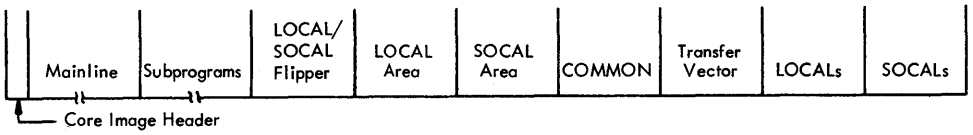


Figure 5. Layout of a Core Image Program Stored in the User/Fixed Area

The following chart is a summary of the information transfers and format conversions performed by STORECI.

Possible Sources, Including Formats	Possible Destinations, Including Formats
WS(DSF)	UA or FX(DCI)
CD(CDS)	UA or FX(DCI)
PT(PTS)	UA or FX(DCI)

The following chart is a summary of the information transfers and format conversions performed by STOREMOD.

Possible Sources, Including Formats	Possible Destinations, Including Formats
WS(DSF)	UA(DSF)
WS(DDF)	UA or FX(DDF)
WS(DCI)	UA or FX(DCI)

*STOREMOD

The STOREMOD operation moves information from Working Storage into the User/Fixed Area. If the name of the DSF program, DCI program, or Data File specified on the STOREMOD control record is identical to an entry in LET/FLET (that is, a DSF program, DCI program, or Data File of the same name already resides in the User/Fixed Area), the information in Working Storage overlays (replaces) that DSF program, DCI program, or Data File in the User/Fixed Area. The format of Working Storage must match the format of the LET/FLET entry which is to be replaced.

If the name on the STOREMOD control record does not match an entry in LET/FLET, a simple STORE operation is performed (see *STORE).

The STOREMOD operation permits the user to modify a DSF program, DCI program, or Data File in the User/Fixed Area without changing its name or relative position within the area. However, the length of the DSF program, DCI program, or Data File in Working Storage cannot be greater than the length of the DSF program, DCI program, or Data File that it replaces in the User/Fixed Area. No change is made to the LET/FLET entry as a result of this operation.

The format of the STOREMOD control record is as follows.

Card Column	Contents	Notes
1-10	*STOREMODb	
11-12	Reserved	
13-14	"FROM" symbol	The source is always Working Storage.
15-16	Reserved	
17-18	"TO" symbol	See chart below.
19-20	Reserved	
21-25	Program name	
26-30	Reserved	
31-34	"FROM" cartridge ID	
35-36	Reserved	
37-40	"TO" cartridge ID	
41-80	Not used	

*DELETE

The DELETE operation removes a specified DSF program, DCI program, or Data File from the User/Fixed Area. The deletion is accomplished by the removal of the LET/FLET entry (or entries) for the DSF program, DCI program, or Data File, including the dummy entry for associated padding, if any.

If a DSF program, DCI program, or Data File is deleted from the User Area, that area is packed so that (1) the areas represented by LET entries are contiguous, and (2) Working Storage can be increased by the amount of disk storage formerly occupied by the deleted DSF program, DCI program, or Data File.

If a DCI program or Data File is deleted from the Fixed Area, no packing of that area occurs. The FLET entry for the deleted DCI program or Data File, including the dummy entry for associated padding, if any, is replaced by a single dummy entry (1DUMY) representing the area formerly occupied by the deleted DCI program or Data File and its padding. DUP store operations may be used to place new entries in the Fixed Area.

The contents of Working Storage are not destroyed by the DELETE operation.

The format of the DELETE control record is as follows.

Card Column	Contents	Notes
1-8	*DELETEb	
9-20	Reserved	
21-25	Program name	
26-30	Reserved	
31-34	"FROM" cartridge ID	The deletion is performed on the specified cartridge only. If no cartridge ID is specified, and the program or data file name (21-25) is present in LET/FLET of more than one cartridge specified for this JOB, the deletion will be from the first logical drive on which the name is found.
35-80	Not used	

*DEFINE

The DEFINE operation (1) initially establishes the size of the Fixed Area, (2) increases or decreases the size of the Fixed Area, (3) deletes the Assembler or FORTRAN Compiler, or both, from the System Area. If the Assembler and/or FORTRAN Compiler is to be deleted, this deletion must be performed prior to defining the Fixed Area, (which is restricted to the master cartridge), or after completely removing a defined Fixed Area.

Definition of a Fixed Area on disk allows the user to store DCI programs and Data Files in fixed locations, which can subsequently be referred to by sector address. The Fixed Area is defined in cylinder increments (one cylinder minimum). When a FIXED AREA is defined, one cylinder is always reserved for FLET, i.e., the initial definition of the Fixed Area must be two cylinders.

Increases and decreases in the size of the Fixed Area must also be made in cylinder units; however, the Fixed Area cannot be decreased by a number greater than the number of unused cylinders at the end of the last program or data file in the Fixed Area. If all DCI programs and Data Files have been deleted from the Fixed Area (1DUMY entries) and the Fixed Area is decreased to less than two cylinders by a DEFINE FIXED AREA control record, the remaining Fixed Area, as well as FLET, is deleted. The Fixed Area and FLET will likewise be deleted if the DEFINE FIXED AREA control record specifies a decrease that exceeds the number of cylinders of Fixed Area on the cartridge.

The control record format for definition of the Fixed Area is described below.

Card Column	Contents	Notes
1-8	*DEFINEb	
9-18	FIXEDbAREA	
19-26	Reserved	
27-30	Count	In initial definition of the Fixed Area, the count is the number (decimal) of cylinders to be allocated as the Fixed Area which must INCLUDE one cylinder for FLET, thus a minimum of two cylinders must be specified. After initial definition, the count is the number of cylinders by which the Fixed Area is to be increased or decreased.
31	Sign	If the Fixed Area is being decreased, this column contains a minus sign; otherwise, it is blank.
32-36	Reserved	
37-40	Cartridge ID	This ID specifies the cartridge which is to be altered.
41-80	Not used	

Deletion of the Assembler and/or FORTRAN Compiler causes the specified Monitor programs to be removed from the IBM System Area on the master cartridge. The IBM System Area is then packed so that following programs and areas occupy the areas formerly occupied by the deleted Monitor programs. SLET entries are updated to reflect the new disk storage allocation for the Monitor programs. The reload table is used to make adjustments in the programs which use disk storage addresses from SLET. If the Assembler and/or FORTRAN Compiler is to be deleted, the user must perform this deletion before defining the Fixed Area on the master cartridge, or after completely removing the Fixed Area. After the Assembler and/or FORTRAN Compiler have been deleted, neither can be restored without performing an initial load.

The control record format for deletion of the Assembler and/or FORTRAN Compiler is described below.

Card Column	Contents	Notes
1-8	*DEFINEb	
9-13	VOIDb	
14-22	ASSEMBLER or FORTRANbb	
23-80	Not used	

*DWADR

The DWADR control record causes a sector address to be written on every sector of Working Storage on the cartridge specified by the DWADR control record, or if no ID is specified, on the System Working Storage. The operation restores correct disk sector addresses in Working Storage if they have been modified during execution of a user's program.

The contents of Working Storage prior to the operation are destroyed.

Following the sector address word (word 0), the first 240 words of each sector contain the sector address of that sector, including the drive code. The remaining 80 words of each sector contain zeros.

A dummy //DUP record is printed on the principal printer following the printing of the *DWADR control record and the DUP exit message.

The format of DWADR control record is as follows.

Card Column	Contents	Notes
1-6	*DWADR	
7-36	Reserved	
37-40	Cartridge ID	This ID specifies the cartridge on which the Working Storage sector addresses are to be rewritten.
41-80	Not used	

ASSEMBLER

The basic language for the Assembler in the Monitor system is described in the publication IBM 1130 Assembler Language (Form C26-5927). Therefore, this section contains only a general description of the Assembler program and its operation. Assembler control records are described in the section Assembler Control Records; Assembler messages, error messages, and error detection codes are listed in Appendix A.

The 1130 Monitor Assembler cannot be operated independently of the Monitor system; however, the Assembler can be deleted from the Monitor system if desired (see *DEFINE under DUP Control Records).

An ASM Monitor control record is used to call the Assembler into operation. The Assembler reads the source program, including control records, from the principal input device. After assembly, the object program resides in System Working Storage. The object program can now be (1) called for execution with an XEQ Monitor control record, (2) stored in the User/Fixed Area with a STORE or STORECI operation (see DUP Control Records), or (3) punched as a binary deck or tape with a DUMP operation (see DUP Control Records).

If symbol table overflow exceeds the number of sectors allocated for overflow by the OVERFLOW SECTORS control record (a maximum of 32 sectors is allowed), an Assembler error message is printed. The approximate maximum size of the symbol table (including overflow) and, hence, the maximum number of symbols that can be defined in a program, is determined by the size of core storage as indicated below:

Size of Core Storage (Words)	4096	8192	16384	32768
Symbol Table Size	3500	4865	7595	13055

CARD OPERATION

The source deck (including Assembler control cards) can be assembled either as part of a job or as a separate job. In either case, the source deck must be preceded by an ASM Monitor control record.

One-Pass Mode

In most cases, the source deck is passed through the 1442 Card Read Punch or 2501 Card Reader only once. If the assembly is part of a stacked job, the assembly proceeds without operator intervention. If the END card of the source deck is the last card in the hopper, press reader START when the reader goes not-ready.

The assembly of a program may start in one-pass mode and then change to two-pass mode. This condition occurs when the intermediate output of pass 1 exceeds the capacity of Working Storage less the number of overflow sectors specified. The system WAITs at the preoperative error trap ($PRET) with /100E (1442 input) or /400E (2501 input) displayed in the Accumulator (see Assembler error messages, Appendix A). If this assembly is part of a stacked job, operator intervention is necessary to prevent the Assembler from reading the Monitor control card following the END card of the source deck. Remove the stacked input behind the END card and press PROGRAM START. The assembly will continue in two-pass mode

Two-Pass Mode

In some cases it may be known in advance that it is necessary to assemble in two-pass mode, that is, pass the source deck through the 1442 Card Read Punch or the 2501 Card Reader twice. If a copy of the source deck, including all Assembler control records, is placed behind the original, the source deck will be read twice, and a stacked job is again possible even when in two-pass mode. Two-pass mode is not allowed with 1134 or Keyboard input.

It is important to note that when a deck is being assembled in two-pass mode, the Assembler is ready to read another card as soon as pass 1 processing of the END card is completed. Therefore, a Monitor control record must not follow the END card the first time (or the first END card if the deck has been copied), or the Assembler will trap this record and execute a CALL EXIT.

If the deck has not been copied, the END card should be the last card in the hopper. Press reader START to process the last card and complete pass 1. The Assembler will then try to read cards for pass 2; therefore, the source deck (with its control cards) should be removed from the stacker and placed in the hopper. Press reader START to begin pass 2 of the assembly. Operation is continuous if the source deck is taken from the stacker during pass 1 and placed in the hopper behind the END card. If the END card is the last card in the hopper, press reader START to complete the assembly.

Punch Symbol Table Option

If the *PUNCH SYMBOL TABLE Assembler control card is used and the principal input device is the 1442 Card Read Punch, sufficient blank cards must be placed after the END card and before the next Monitor control record in the stacked job input. (If a non-blank card is read when punching on the 1442-6, 7 the

Assembler will WAIT at the preoperative error trap ($PRET) with /100F displayed in the accumulator). In estimating the number of blank cards required, allow one card for each symbol used in the source deck. Unnecessary blank cards will be passed until the next Monitor control record is read.

If the system configuration is 2501/1442, place blank cards in the 1442 hopper and press 1442 START before beginning the assembly.

Note: Do not place non-blank cards in the 1442-5. The punch may be damaged if an attempt is made to punch a hole where a hole exists. No error is detected.

KEYBOARD/PAPER TAPE OPERATION

Most of the procedures for card input are also applicable to keyboard/paper tape input. The LIST DECK, LIST DECK E, PUNCH SYMBOL TABLE, and TWO PASS MODE options are not allowed with keyboard/paper tape input.

Note: The paper tape input to the Assembler is punched in PTTC/8 code, one frame per character. The format of the keyboard/paper tape control records is the same as the card format. The format of the symbolic program keyboard/paper tape records is the same as card format except for the following:

- The record does not contain leading blanks corresponding to card columns 1-20.
- The record does not contain blanks or data corresponding to card columns 72-80.
- Trailing blanks need not be used. Therefore, up to 51 characters (corresponding to card columns 21-71) can appear in the record.

The assembly is continuous, and at the end of the assembly control is returned to the Supervisor, which will then pass any delete codes between the Assembler and the next Monitor control record. The assembler will also pass any codes that may occur between paper tape records of the source program.

The first record processed by the Assembler is checked for an asterisk in column one. If an asterisk is present in column one, this record is treated as an Assembler control record. This procedure continues until the first non-asterisk character is detected in column one. For this record, and all records following (up to and including the END statement), column one is treated as if it were column twenty-one; therefore, the first non-control record should not be an * comments record.

ORIGIN OF MAINLINES

The origin of a relocatable program is always set at zero unless otherwise specified in the source program.

The origin of an absolute mainline program, if not otherwise specified in an ORG statement, is set to the end of DISKN plus 30 (the core image header record is 30 words long).

If the program requires DISKZ, DISK1, or DISKN, the origin may be set to the end of the requested disk I/O subroutine plus 30.

If no disk I/O subroutine is used by the program, the origin may be set as low as the end of DISKZ plus 30.

Note that if DISKZ is in core during execution (required or not), the ORG statement for the program being executed must specify an even core address greater than or equal to the end of DISKZ plus 30. An ORG to the end of DISKZ plus 30, followed by a BSS or a BES of an odd number of locations is not allowed. This sequence has the same effect as an ORG to an odd location.

ASSEMBLER CONTROL RECORDS

Assembler control records are used to specify options affecting an assembly and its output. These control records must precede the source program and can be in any order (see Figure 6). Assembler control records

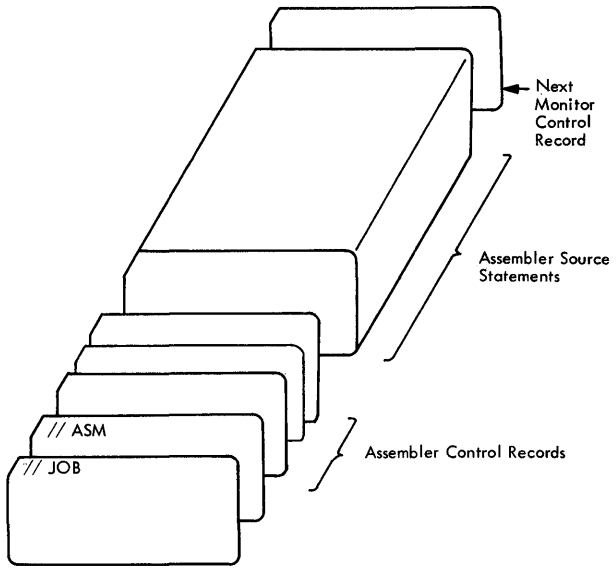


Figure 6. Layout of an Assembler Input Deck

can be entered in card or paper tape form along with the source program deck or tape or, unless otherwise noted, may be entered from the Keyboard along with the source statements (see // TYP under Monitor Control Records).

All Assembler control records have the following format:

Column 1: * (asterisk)
2-71: Option

If an Assembler control record contains an asterisk in column 1, but the option does not agree, character for character, with its valid format, as described below, the asterisk is replaced by a minus sign on the control record listing. The erroneous control record is ignored and no other error occurs.

Assembler control records can be written in free form; that is, any number of blanks may occur between the characters of the option. However, only one blank must separate the last character in the option, and the first character of any required numeric field. Remarks may be included in the control record following the option or numeric field; however, at least one blank must separate the last character of the option or numeric field and the remarks.

*TWO PASS MODE

This control record causes the Assembler to read the source deck twice. TWO PASS MODE must be specified when:

- The user desires a list deck to be punched on the 1442 Card Read Punch, model 6 or 7 (see LIST DECK and LIST DECK E).
- One-pass operation cannot be performed because the intermediate output (source records) exceeds the capacity of Working Storage.

This control record is ignored if source statements are entered from the Keyboard or the 1134 Paper Tape Reader.

The format of the TWO PASS MODE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	TWO PASS MODE	
72-80	Not used	

*LIST

This control record causes the Assembler to provide a printed listing on the principal print device (1403 Printer, 1132 Printer, or Console Printer). The format of the printed listing corresponds to that of the list deck (see Figure 7). If the LIST control record is not used, only those statements in which assembly errors are detected will be listed. All BSS, BES, ORG, and EQU statements in which errors are detected will be unconditionally listed in Pass 1 of the assembly.

A sample program listing appears in Appendix J.

The format of the LIST control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	LIST	
72-80	Not used	

*LIST DECK

This control record causes the Assembler to punch a list deck if the principal I/O device is a 1442 model 6 or 7 Card Read Punch. This option requires two passes of the source deck (TWO PASS MODE). The list deck format is shown in Figure 7. Object information is punched into columns 1-19 of the source deck during pass 2.

This control record is ignored if entered from the 2501 Card Reader, the 1134 Paper Tape Reader, or the Keyboard.

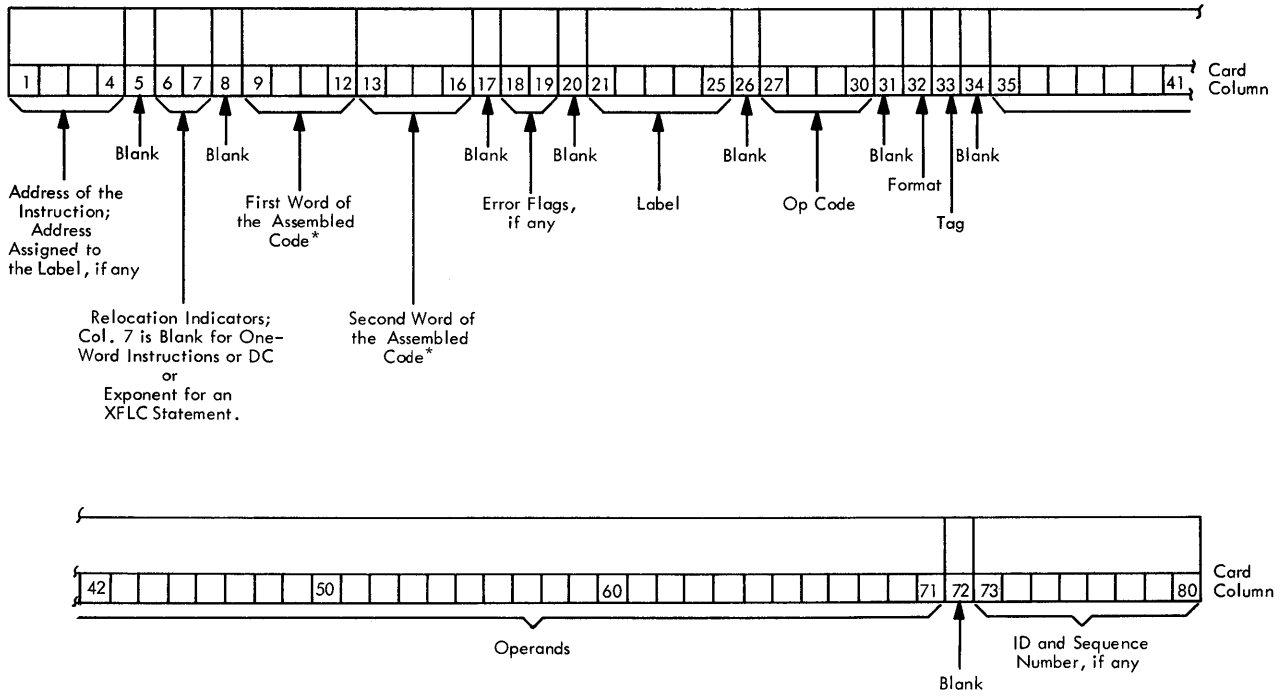
The format of the LIST DECK control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	LIST DECK	
72-80	Not used	

*LIST DECK E

This control record causes the Assembler to punch assembly error codes only (columns 18-19) in the list deck output (see LIST DECK). The principal I/O device must be a 1442 model 6 or 7 Card Read Punch. The Assembler error detection codes are listed in Appendix A.

This control record is ignored if entered from the 2501 Card Reader, the 1134 Paper Tape Reader, or the Keyboard.



*For EBC statements, columns 9-12 contain the number of EBC characters.
For BSS and BES statements, columns 9-12 contain the number of words reserved for the block.
For ENT, ILS, and ISS statements, columns 9-16 contain the entry label in packed EBCDIC code.

Figure 7. List Deck Format

The format of the LIST DECK E control record is as follows:

Card Column	Contents	Notes
1	*	Asterisk
2-71	LIST DECK E	
72-80	Not used	

*PRINT SYMBOL TABLE

This control record causes the Assembler to provide a printed listing of the symbol table on the principal print device. Symbols are grouped five per line. Multiply-defined symbols are preceded by the letter M; symbols with absolute values in a relocatable program are preceded by the letter A. The M and A flags, however, are not counted as assembly errors.

The format of the PRINT SYMBOL TABLE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	PRINT SYMBOL TABLE	
72-80	Not used	

*PUNCH SYMBOL TABLE

This control record causes the Assembler to punch the symbol table as a series of EQU source cards. Each source card contains one symbol. These cards can be used as source input to the System Symbol Table when the SAVE SYMBOL TABLE control record is used with an assembly in which they are included:

This control record is ignored if entered from the 1134 Paper Tape Reader or the Keyboard.

The format of the PUNCH SYMBOL TABLE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	PUNCH SYMBOL TABLE	
72-80	Not used	

*SAVE SYMBOL TABLE

This control record causes the Assembler to save the symbol table generated in this assembly on the disk as a System Symbol Table. This System Symbol Table is saved until the next assembly containing a SAVE SYMBOL TABLE control record causes a new assembly-generated symbol table to replace it. This control record is also used with the SYSTEM SYMBOL TABLE control record to add symbols to the System Symbol Table. The SAVE SYMBOL TABLE option requires that this assembly be absolute. If any assembly errors are detected, or if the symbol table exceeds 100 symbols, the symbol table is not saved as a System Symbol Table, and an assembly error message is printed (see Assembler Error Messages, Appendix A).

The format of the SAVE SYMBOL TABLE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	SAVE SYMBOL TABLE	
72-80	Not used	

*SYSTEM SYMBOL TABLE

This control record causes the Assembler to add the System Symbol Table (previously built by a SAVE SYMBOL TABLE assembly) to the symbol table for this assembly as the assembly begins. This control record is used when it is desired to refer to symbols in the System Symbol Table without redefining those symbols in the source program, or it is used together with the SAVE SYMBOL TABLE control record when it is desired to add symbols to the System Symbol Table. All symbols in the System Symbol Table have absolute values.

The format of the SYSTEM SYMBOL TABLE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	SYSTEM SYMBOL TABLE	
72-80	Not used	

*LEVEL

This control record specifies the interrupt levels serviced by an ISS and, hence, the associated ILS subroutines. It is required for the assembly of an ISS subroutine. The interrupt level number is a decimal number in the range 0-5. If the device operates on more than one interrupt level (for example, the 1442 Card Read Punch), one LEVEL control record is required for each interrupt level on which the device operates. At least one blank must separate the word LEVEL and the interrupt level number.

If a LEVEL control record is not used when assembling an ISS subroutine, an Error Message is printed at the end of the assembly (see Assembler Error Messages, Appendix A).

The format of the LEVEL control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	LEVELbn	n is an interrupt level number
72-80	Not used	

*OVERFLOW SECTORS

This control record specifies the number of sectors of Working Storage to be used by the Assembler for symbol table overflow. The number of overflow sectors (nn) is a decimal number between 1 and 32. If the entry is zero or blank, no overflow sectors are allowed. If the entry is greater than 32, only 32 overflow sectors are allowed. If this control record is not used, no overflow sectors are allowed; if it is used, the Assembler actually allocates one more sector than the number specified. This additional sector is used as a working sector when the Assembler is handling symbol table overflow.

The format of the OVERFLOW SECTORS control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	OVERFLOW SECTORSbnn	nn is the number of sectors assigned to symbol table overflow.
72-80	Not used	

*COMMON

This control record specifies the length (in words) of COMMON as defined by a FORTRAN core load that is to be executed prior to the execution of the program being assembled. Use of this control record provides for a COMMON area to be saved in linking between FORTRAN mainlines and Assembler mainlines.

The format of the COMMON control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-71	COMMONbnnnnn	nnnnn is the number of words of COMMON (decimal) to be saved between links.
72-80	Not used	

FORTRAN COMPILER

The basic language for the FORTRAN Compiler in the Monitor system is described in the publication IBM 1130/1800 Basic FORTRAN IV Language (Form C26-3715); therefore, this section contains only a general description of the Compiler and its operation. The FORTRAN Compiler control records are described in the section FORTRAN Control Records; FORTRAN messages and error messages are listed in Appendix A.

The FORTRAN Compiler cannot be operated independently of the Monitor system; however, it can be deleted from the Monitor system if desired (see *DEFINE under DUP Control Records).

A FOR Monitor control record is used to call the FORTRAN Compiler into operation. The Compiler reads the source program, including control records, from the principal input device. After compilation, the object program resides in System Working Storage and can be (1) called for execution with an XEQ Monitor control record, (2) stored in the User/Fixed Area with a STORE or STORECI operation (see DUP Control Records), or (3) punched as a binary deck or tape with a DUMP operation (see DUP Control Records).

The 1130 FORTRAN I/O logical unit numbers and record sizes are listed in Table 3.

//b RECORDS READ DURING THE EXECUTION OF A FORTRAN PROGRAM

During the execution of a FORTRAN program, any //b record encountered by CARDZ, READZ, or PAPTZ will cause an immediate CALL EXIT. The Supervisor will then search for the next valid Monitor control record entered from the reader. Only the //b characters on the record trapped by CARDZ, READZ, or PAPTZ are recognized. Any other data entered in this record is not available to programs in the Monitor system. The record is not listed. For off-line listing purposes, however, this record can contain comments (e.g., // END OF DATA).

FORTRAN CONTROL RECORDS

Before a FORTRAN program is compiled, the user can specify certain options affecting both the compilation and execution of the program by means of control records. These control records must precede the source program and can be in any order (see Figure 8).

FORTRAN control records can be entered in card or paper tape form along with the source program deck or tape, or they may be entered from the Keyboard along with the source statements (see // TYP under Monitor Control Records). The IOCS, NAME and ORIGIN control records can be used only in mainline programs; the others can be used in both mainline programs and subprograms.

All FORTRAN control records have the following format:

Column 1: *(asterisk)
2-72: Option

Table 3. FORTRAN I/O Logical Unit Designations and Record Sizes

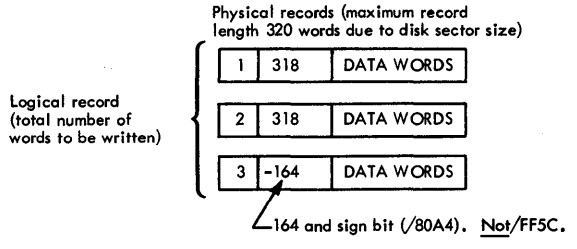
Logical Unit Number	Device	Kind of Transmission	Record Size Allowed
1	Console Printer	Output only	120
2	1442 Card Read Punch	Input/output	80
3	1132 Printer	Output only	1 carriage control + 120
4	1134/1055 Paper Tape Reader Punch	Input/output	80, plus max. of 80 case shifts for PTTC/8 code, plus NL code.
5	1403 Printer	Output only	1 carriage control + 120
6	Keyboard	Input only	80
7	1627 Plotter	Output only	120
8	2501 Card Reader	Input only	80
9	1442 Card Punch	Output only	80
10	UDISK	Unformatted input/output without data conversion	320*

*Unformatted disk I/O comprises 320 word records (including a two-word header). The first word of the header must contain the count of the physical record within the logical record (see example following). The second word of the header must contain the number of effective words in the individual physical record. The second word of the header of the last physical record within a logical record must have the sign bit (-) on. Unformatted disk characters are stored in as they appear in core storage.

Example:

DIMENSION A (400) 800 words

WRITE (10) A



An end-of-file record occupies one sector. Word one of the header must be 1 and word two must be a negative zero (/8000).

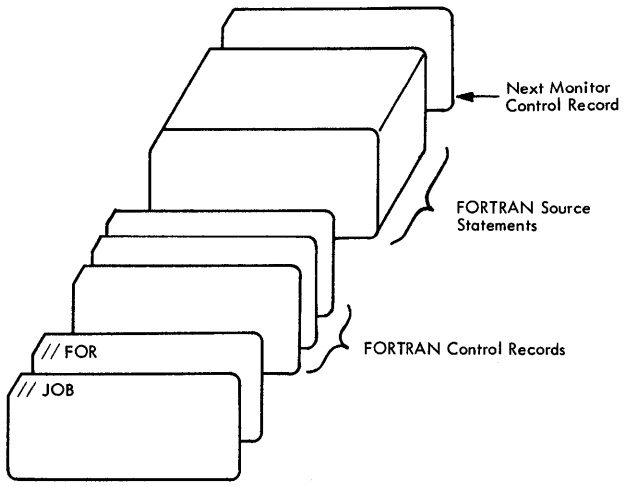


Figure 8. Layout of a FORTRAN Compiler Input Deck

If a FORTRAN control record contains an asterisk in column 1, but the option does not agree, character for character, with its valid format, as described below, the asterisk is replaced by a minus sign on the control record listing. The erroneous control record is ignored in the compilation and the option is not performed; however, no error results.

The same action is taken if in an ORIGIN record the address is not between 0 and 32767 (decimal) or 0000 and 7FFF (hexadecimal).

FORTRAN control records can be written in free form; that is, any number of blanks may occur between the characters of the option. No remarks are allowed.

*IOCS(...)

This control record is required to specify any I/O device that is to be used during execution of the program; however, only the devices required should be included. Because the IOCS control record may appear only in the mainline program, it must include all the I/O devices used by all FORTRAN subprograms that are called. The device names must be in parentheses with a comma between each name. The valid names and the devices to which they correspond are listed below:

Name	Device
CARD	1442 Card Read Punch, Model 6 or 7
2501 READER	2501 Card Reader
1442 PUNCH	1442 Card Punch, Model 5 (1442 Model 6 or 7 if used as a punch only)
TYPEWRITER	Console Printer
KEYBOARD	Keyboard
1132 PRINTER	1132 Printer
1403 PRINTER	1403 Printer
PAPER TAPE	1134/1055 Paper Tape Reader/Punch
PLOTTER	1627 Plotter
DISK	Disk
UDISK	Disk (unformatted disk I/O)

Note that CARD is used for the 1442 Card Read Punch, Model 6 or 7 and that 1442 PUNCH is used for the 1442 Card Punch, Model 5 (1442 PUNCH may be used with a 1442 Model 6 or 7 if the function is punch only; 1442 PUNCH uses less core). These two names are mutually exclusive; therefore, the use of both the CARD and 1442 PUNCH IOCS Control Records in the same compilation is not allowed.

Subprograms that are a part of a FORTRAN core load but are written in Assembler language can use any I/O subroutines for any device that is not specified on the IOCS control record. Otherwise they must use the same I/O subroutine as the FORTRAN subprogram.

Any number of IOCS control records can be used to specify the required device names.

The format of the IOCS control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	IOCS (d, d, ..., d)	d is a valid device name selected from the above list.
73-80	Not used	

*LIST SOURCE PROGRAM

This control record causes the Compiler to list the source program on the principal print device as it is read in.

The format of the LIST SOURCE PROGRAM control record is as follows:

Card Column	Contents	Notes
1	*	Asterisk
2-72	LIST SOURCE PROGRAM	
73-80	Not used	

*LIST SUBPROGRAM NAMES

This control record causes the Compiler to list on the principal print device the names of all subprograms (including EXTERNAL subprograms) called directly by the compiled program.

The format of the LIST SUBPROGRAM NAMES control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	LIST SUBPROGRAM NAMES	
73-80	Not used	

*LIST SYMBOL TABLE

This control record causes the Compiler to list the following items on the principal print device:

- Variable names and their absolute or relative addresses
- Statement numbers and their absolute or relative addresses
- Statement function names and their absolute or relative addresses
- Constants and their addresses

The format of the LIST SYMBOL TABLE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	LIST SYMBOL TABLE	
73-80	Not used	

*LIST ALL

This control record causes the Compiler to list the source program, subprogram names, and the symbol table on the principal print device. If this control record is used, the other LIST control records are not required.

The format of the LIST ALL control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	LIST ALL	
73-80	Not used	

*EXTENDED PRECISION

This control record causes the Compiler to store variables and real constants in three words instead of two and to generate linkage to extended precision subprograms.

The format of the EXTENDED PRECISION control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	EXTENDED PRECISION	
73-80	Not used	

*ONE WORD INTEGERS

This control record causes the Compiler to allocate one word of storage for integer variables rather than the same allocation (two or three words) used for real variables. Whether this control record is used or not, integer constants are always contained in one word. When this control record is used, the program does not conform to the USASI Basic FORTRAN standard for data storage and may require modification in order to be used with other FORTRAN systems.

The format of the ONE WORD INTEGERS control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	ONE WORD INTEGERS	
73-80	Not used	

*NAME

This control record causes the Compiler to print the specified program name at the end of the listing. The name is five consecutive characters (including blanks) starting at the first non-blank column following NAME. At least one blank must separate the word NAME and the mainline program name.

The format of the NAME control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	NAMEbxxxxx	xxxxx is the name of the mainline object program.
73-80	Not used	

**(Header Information)

This column record causes the Compiler to print the information in columns 3-72 at the top of each page of compilation printout when a 1403 Printer or 1132 Printer is the principal print device. It initially causes a skip to channel 1 when the first statement of the program is read.

The format of the header control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2	*	Asterisk
3-72	Any string of characters	
73-80	Not used	

*ARITHMETIC TRACE

This control record causes the Compiler to generate linkage to the trace subprograms, which are executed whenever a value is assigned to a variable on the left of an equal sign. If console entry switch 15 is on during execution and program logic (see Optional Tracing) does not prevent tracing, the value of the assigned variable is printed as it is calculated.

If tracing is requested, an IOCS control record must also be present to indicate that either the typewriter (that is, the Console Printer), 1132 Printer, or 1403 Printer is needed. If more than one print device is specified in the IOCS control record, the fastest device is used for tracing.

The traced value for a variable to the left of an equal sign of an arithmetic statement is printed with one leading asterisk.

The format of the ARITHMETIC TRACE control record is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	ARITHMETIC TRACE	
73-80	Not used	

*TRANSFER TRACE

This control record causes the Compiler to generate linkage to the trace subprograms, which are executed whenever an IF statement or computed GO TO statement is encountered. If console entry switch 15 is on during execution and program logic (see Optional Tracing) does not prevent tracing, the value of the IF expression or the value of the computed GO TO index is printed.

If tracing is requested, an IOCS control record must also be present to indicate that either the typewriter (that is, the Console Printer), 1132 Printer, or 1403 Printer is needed. If more than one print device is specified in the IOCS control record, the fastest device is used for tracing.

The traced value for the expression in an IF statement is printed with two leading asterisks. The traced value for the index of a computed GO TO statement is printed with three leading asterisks.

The format of the TRANSFER TRACE control records is as follows.

Card Column	Contents	Notes
1	*	Asterisk
2-72	TRANSFER TRACE	
73-80	Not used	

Optional Tracing

The user can elect to trace only selected parts of the program by placing statements in the source program logic flow to start and stop tracing. This is done by executing a CALL TSTOP to stop tracing or a CALL TSTRT to start tracing. Thus, tracing occurs only if:

- Console entry switch 15 is on (can be turned off at any time)
- The trace control records were compiled with the source program
- A CALL TSTOP has not been executed, or a CALL TSTRT has been executed since the last CALL TSTOP.

*ORIGIN ddddd or *ORIGIN/xxxx

This control record causes the compiler to output absolute object code starting at the address specified, The address should consist of 1-5 decimal digits or 1-4 hexadecimal digits preceded by a slash. Furthermore the address must be in the range 0-32767 (decimal), i.e. 0000-7FFF (hexadecimal).

The ORIGIN dddd control record is as follows:

Card Column	Contents	Notes
1	*	Asterisk
2-72	ORIGIN ddddd	ddddd is the decimal address as specified above.
13-80	Not Used	

The ORIGIN/xxxx control record is as follows:

Card Column	Contents	Notes
1	*	Asterisk
2-72	ORIGIN/xxxx	xxxx is the hexadecimal address as specified above.

Operating Notes

A constant in a STOP or PAUSE statement is treated as a hexadecimal number. This hexadecimal number and its decimal equivalent appear in the list of constants. The hexadecimal number is also displayed in the accumulator when the system waits at $PRET during the execution of the PAUSE or STOP statement.

Variables and constants that require more than one word of storage have the address of the word nearest the zero address of the machine. In the case of arrays, the given address refers to the addressed word of the first element. In the case of a two- or three-word integer, the integer value is contained in the addressed word. The first variable listed might not be addressed at 0000 because space may be required for generated temporary storage locations.

The relative address for variables not in COMMON would be the actual address if the program started at storage location zero. The relative address for variables in COMMON would be the actual address if the machine had 32K storage. Variables in COMMON reside in the high-order core location of the machine being used (e.g., first COMMON variable will be loaded to /1FFF on an 8K machine).

Any of the three versions of the disk I/O subroutines may be used with a FORTRAN core load. However, under normal circumstances no advantage in speed may be gained, because the FORTRAN disk formatting subroutine operates with one sector at a time. SOCALs may operate faster if DISKN is used.

KEYBOARD INPUT OF DATA RECORDS

Data records of up to 80 characters can be read from the keyboard by a FORTRAN READ statement. Data values must be right-justified in their respective fields.

Keyboard Operation

If it is desirable to key in less than 80 characters, the EOF key can be pressed to stop transmittal. Also, the ERASE FIELD or BACKSPACE key can be pressed to restart the record transmittal if an error is detected while entering data. If the keyboard appears to be locked up, press REST KB to restore the keyboard. The correct case shift must be selected before data is entered.

Buffer Status After Keyboard Input

Before entering each data record the buffer is filled with blanks. Therefore, when the EOF key is pressed prior to completing a full buffer load of 80 characters, the rest of the buffer remains blank. If more data is necessary to satisfy the list items, the remaining numeric fields (I, E, or F) are stored in core as zeros and remaining alphameric fields (A or H) are stored as blanks. Processing is continous and no errors result from the above condition.

Note: For information about buffer status after pressing the ERASE FIELD or BACKSPACE key, SUBROUTINE FUNCTIONS, Re-entry concerning TYPEZ.

OBJECT PROGRAM PAPER TAPE DATA RECORD FORMAT

Data records of up to 80 EBCDIC characters in PTTC/8 code can be read or written by the FORTRAN object programs. The delete and new-line codes are recognized. Delete codes and case shifts are not included in the count of characters. If a new-line code is enountered before the 80th character is read, the record is terminated. If the 80th character is not a new-line code, the 81st character is read and assumed to be a new-line code. A new-line code is punched at the end of each output record.

A-CONVERSION

Spacing, tabulating, and shifting on the Console Printer can be controlled by outputting a unique value for the operation desired. These values must be assigned as integer constants and outputted through A-Conversion.

The operations that can be performed and the unique values assigned to them are:

OPERATION	VALUE
Backspace	5696
Carrier Return	5440
Line Feed	9536
Shift to print black	5184
Shift to print red	13632
Space	16448
Tabulate	1344

As an example of Console Printer control, assume that a variable, X, is to be printed in the existing black ribbon shift and that another variable, Y, is to be shifted back to black. This can be accomplished as follows:

```
      I=1344
      J=13632
      K=5184
      L=1
      WRITE (L,3)X, I, J, Y, K
    3 FORMAT (F12.6, 2A1, F12.6, A1)
```

FORTRAN logical unit 1, as specified in the WRITE statement, is the Console Printer. The sequence of operations to be performed are: print the variable X, tabulate, shift to print red, print the variable Y, shift to print black.

Each control variable counts as one character and must be included in the count of the maximum line length.

FORTRAN I/O ERRORS

If input/output errors are detected during execution, the program stops and execution should not be continued. The error is indicated by a display in the accumulator. The error displays and meanings are listed in Appendix A, Table 12.

When the output field is too small to contain the number, the field is filled with asterisks and execution is continued.

The input/output routines used by FORTRAN (PAPTZ, CARDZ, PRNTZ, WRTYZ, TYPEZ, PNCHZ, READZ, PRNZ) wait on any I/O device error or device not in a ready condition. When the devices are ready, press PROGRAM START to execute the I/O operation.

Error detection in functional and arithmetic subroutines is possible by the use of source program statements. Refer to "FORTRAN Machine and Program Indicator Tests" in the manual, IBM 1130/1800 Basic FORTRAN IV Language (Form C26-3715).

RPG

The RPG specifications are described in the publication IBM 1130 RPG Specifications, Form C21-5002; therefore, this section contains only a general description of the RPG program and its operation. The RPG control and End of File cards are described under the heading RPG Compiler Control. RPG error messages and error notes are described in Appendix A.

The RPG Compiler cannot be operated independently of the Monitor system; however, it can be deleted from the Monitor system if desired (see *DEFINE under DUP Control Records).

An RPG Monitor control record (// RPG) is used to call the RPG Compiler into operation. The compiler reads the source program, including the RPG control card and End of File card, from the principal input device. After compilation, the object program resides on disk Working Storage in Disk System Format. The object program can then be (1) called for execution with the XEQ Monitor control record, (2) stored in the User/Fixed Area with a STORE or STORECI operation (see DUP Control Records) or (3) punched as a binary deck with a DUMP operation (see DUP Control Records).

RPG COMPILER CONTROL

The RPG Compiler uses two special cards in its operation. The first, the RPG control card, acts as a header for the source deck and supplies operating parameters to the compiler. The second, the RPG End of File card, acts as a delimiter, and is required at the end of any input to the RPG compiler or to an RPG data file.

RPG Control Card

The first card of an RPG source deck must be the RPG control card. The layout of this card is included on the RPG Control Card and File Description Specifications, form number X24-3347. A detailed description of all entries on this card appears in the 1130 RPG Specifications manual.

For RPG Compiler operation, the entries in column 6 and column 11 of the RPG control card are basic.

- Column 6 of the RPG control card must contain an H.
- Column 11 of the RPG control card indicates the type of run required.
  - blank - Compilation with listing
  - B - Compilation only
  - D - Listing only

All other entries on the RPG control card are optional.

End of File Card

The last card of an RPG source deck must be an End of File card. The End of File card is also required as the last card of a data file.

The format of the End of File card is as follows.

/* (slash in column 1; * in column 2)

Columns 3-80 of the End of File card are not used.

RPG PROGRAM OPERATION

Figure 8.1 illustrates the stacked input required to compile an RPG source program, store the object program in the Users Area and execute the object program. If the // DUP and *STORE card were omitted from the Monitor input, the program would be executed from Working Storage; however, the program would not be available for future execution since it was not saved.

If the program being compiled is not executed often, it may be advisable to store it in cards rather than on disk. Figure 8.2 shows the input required to compile an RPG program and punch an object deck. Figure 8.3 lists the input required to execute the object program from cards.

Most RPG programs require data input during program execution. This data can be input on data cards at execution time or it can be stored on a predefined data file on disk at any time before execution. Figure 8.4 shows how a data file may be built for use with RPG. RPG files may be sequential or indexed-sequential (ISAM). See RPG File Organization in the section Programming Tips and Techniques for detailed information on RPG disk files.

The compiler will print out addresses for various routines in the Key Addresses of Object Program Table. For example, the "Close Files" routine (which is approximately at the end of the mainline program) is included in this table. This routine may require from 2 to 16 additional words (hexadecimal) depending on the type and number of files to be closed. The address of this routine can be helpful when dealing with programs which exceed the available core storage; by adding the number of additional words to the address of the "Close Files" routine, the size of the generated mainline program can be determined.

On an ISAM load function, the compiler prints the following information:

Filename

Number of sectors required if no overflow is desired.

Figure 8.1 Stacked Input to Compiler, Store and Execute an RPG Program.

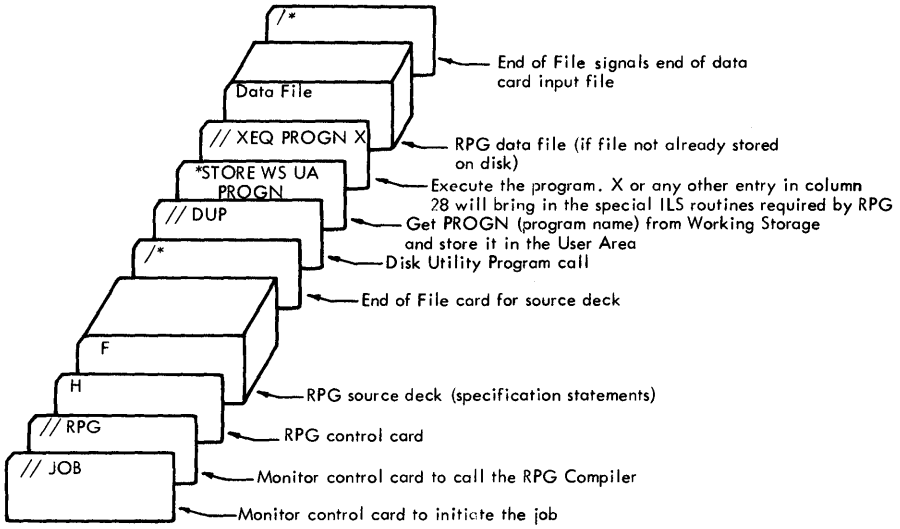


Figure 8.2 Stacked Input to Compile an RPG Program and Punch an Object Deck.

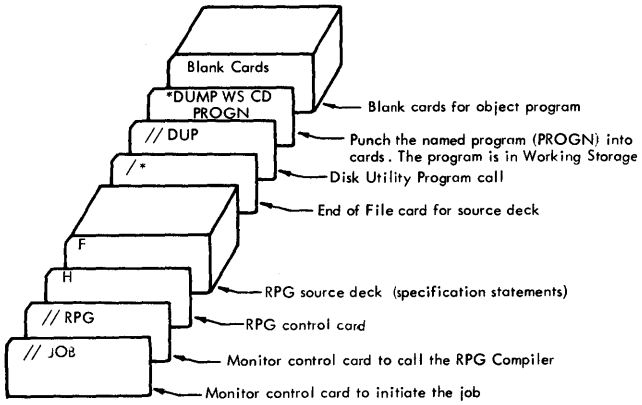


Figure 8.3 Stacked Input to execute an RPG Object Program from Cards.

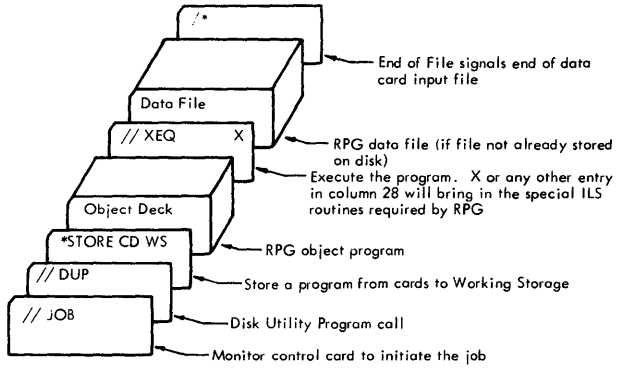
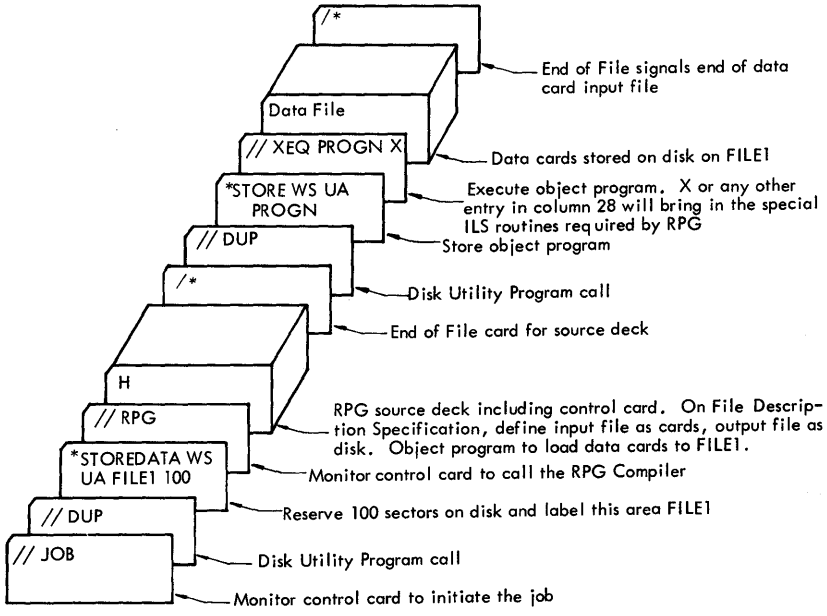


Figure 8.4 Reserving Space for and Storing an RPG File on Disk.



Number of sectors required if ten percent overflow is desired. This information can be used to reserve file space for ISAM records.

The number of sectors needed for a sequential file can be calculated by the following formula

Number of sectors required = Number of records/ /(640/record length)

RPG OBJECT PROGRAM CONSIDERATIONS

The RPG object program requires a special set of ILS subroutines. The user must punch a non-blank character in column 28 of the XEQ card and in column 12 of the STORECI card to assure that they will be loaded. If the program is stored in core image, the ILS subroutines are stored with the program on disk.

The storing of object programs in Disk Core Image format in the User of Fixed Area on disk is not recommended (see Disadvantages of Storing a Program in Disk Core Image Format (DCI)).

CORE LOAD BUILDER

The Core Load Builder builds a specified mainline program into a core image program. The mainline program, with its required programs (LOCALs and SOCALs included), is converted from Disk System format to Disk Core Image format. During the conversion, the Core Load Builder also builds the Core Image Header record and the Transfer Vector. The resultant core image program is suitable for immediate execution or for storing on the disk in Disk Image format for future execution. The Core Load Builder can build a core load that references up to approximately 375 different LIBF and CALL entry points, e.g., 80 LIBFs plus 295 CALLs (the maximum number of LIBFs allowable is 83 due to the size of the LIBF Transfer Vector).

If the core load is built on an 1130 system with core size 4K, the maximum number of different LIBF and CALL entry points is approximately 110.

The Core Load Builder is called by:

- The Supervisor. After the Supervisor has detected the XEQ Monitor control record in the input stream and has read the Supervisor control records, if any, and written them in the Supervisor Control Record Area (SCRA) on disk, the Supervisor dummys up a CALL LINK to the program specified on the XEQ record unless the program resides in Working Storage, in which case the Supervisor calls the Core Load Builder directly. The Core Load Builder then builds the core load and returns control to the Core Image Loader to fetch the core load and transfer control to it.

- DUP. After DUP has detected the STORECI control record, it reads the Supervisor control records, if any, and writes them in the Supervisor Control Record Area (SCRA) on disk. Unless the program is already in Working Storage, DUP fetches the program, converts it to Disk System format, if necessary, and stores it in Working Storage. Next, the Core Load Builder is fetched to construct the core image program (see Core Load Construction). After the core image program has been built, the Core Load Builder returns control to DUP to store the core image program in the User or Fixed Area.

- The Core Image Loader. When the Resident Monitor is entered at the LINK entry point, the Core Image Loader is called to transfer control to the next link. The Core Image Loader determines the format of the link from the LET/FLET entry and, if the program to be executed is in Disk System format, calls the Core Load Builder to construct the core image program (see Core Load Construction). After the core image program has been built, the Core Load Builder returns control to the Core Image Loader to fetch the core load and transfer control to it.

CORE LOAD CONSTRUCTION

The following paragraphs describe the functions of the Core Load Builder during the construction of a core image program. These functions are not necessarily performed in the order in which they appear.

Figure 9 shows a core image program being built.

Figure 5 (see *STORECI under DUP Control Records) shows a core image program stored on disk.

Figure 11 (see Fetching a Link under Core Image Loader) shows a core load ready for execution.

Processing the Contents of the SCRA

The LOCAL, NOCAL, FILES, G2250, and EQUAT control records are read from the Supervisor Control Record Area (SCRA) on disk and analyzed. Tables are built from the information obtained from the respective control record types. These tables are used in later phases of the construction of the core image program.

Conversion of the Mainline Program

The mainline program is converted from Disk System format to Disk Core Image format. The mainline is always converted before any other part of the core load.

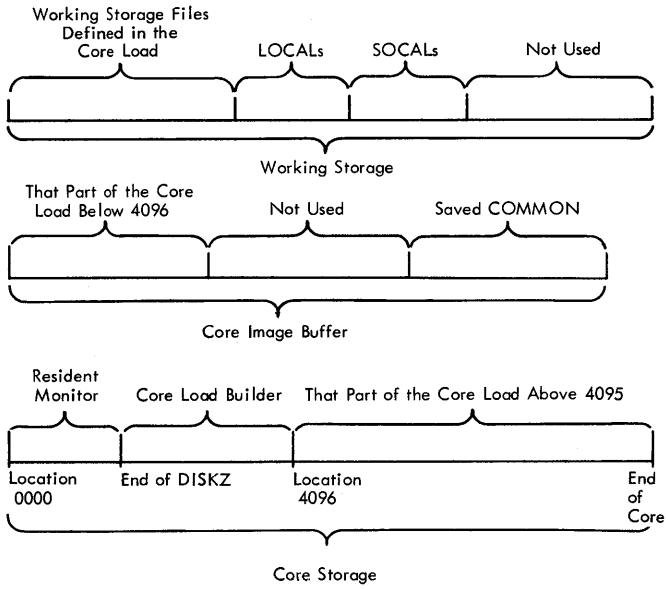


Figure 9. Distribution of a Core Image Program being Built

Incorporation of Subprograms

All the subprograms called by the mailine program and by other subprograms are included in the core load, except for (1) the disk I/O subroutine, (2) any LOCAL subprograms specified, and (3) SOCALs (see System Overlays).

If LOCALs have been specified or if SOCALs are employed by the Core Load Builder, the LOCAL/SOCAL flipper (FLIPR) is included in the core load. The order of conversion is generally NOCALs, followed by the subprograms in the order they are called. The order of processing when either LOCALs or SOCALs are included is more complicated and will not be discussed here.

By means of the function of the EQUAT control record (see SUPERVISOR CONTROL RECORDS) a subroutine, called in the core load that is being built, can be replaced by another subroutine. Furthermore, a symbolic name in a DSA statement can be replaced by another symbolic name.

Provision for LOCALs and SOCALs

If LOCALs have been specified, a LOCAL Area as large as the largest LOCAL is reserved in the core load, into which the LOCAL subprograms are read by the LOCAL/SOCAL flipper. In addition, the subprograms specified on the LOCAL control records are written in Working Storage following any files defined in Working Storage. If the core load is executed immediately, each LOCAL is read, as it is called, from Working Storage into the LOCAL Area by the LOCAL/SOCAL flipper. If the core load is stored in Disk Core Image format before it is executed, the LOCALs are stored following the core load. During execution, the LOCAL/SOCAL flipper fetches them from the User/Fixed Area.

If SOCALs are employed by the Core Load Builder, a SOCAL Area as large as the largest SOCAL (usually SOCAL 2) is reserved in the core load, into which the SOCALs are read by the LOCAL/SOCAL flipper. In addition, the subprograms comprising the SOCALs are written in Working Storage following any files defined in Working Storage and any LOCALs stored there. If the core load is executed immediately, each SOCAL is read from Working Storage into the SOCAL Area by the LOCAL/SOCAL flipper as it is called. If the core load is stored in Disk Core Image format before it is executed, the SOCALs are stored following the core load and the LOCALs, if any. During execution, the LOCAL/SOCAL flipper fetches the SOCALs from the User/Fixed Area.

Construction of the Core Image Header

During the construction of the Core Image program, the Core Load Builder also constructs the Core Image Header, which contains the information required b the Core Image Loader to initialize the core load for execution. This header becomes a part of the core image program and resides in core along with the rest of the core load during execution. Since FORTRAN subroutines access this information during execution, the header is not to be considered a work area.

Processing Defined Files

The Core Load Builder uses the information in the FILES control record to equate files defined in the mainline program (by the FORTRAN DEFINE FILE statement or by the Assembler FILE statement). to Data Files on disk. The processing consists of comparing the file number in a 7-word DEFINE FILE table entry with each of the file numbers from the FILES control records, which have been stored in the SCRA by the Supervisor or DUP. If a match occurs, the name of the disk area associated with the file number on the FILES control record is found in LET/FLET, and the sector address of that disk area (including the logical drive code) is placed in word 5 of the DEFINE FILE table entry. If none of the file numbers from the FILES control records match the number in the DEFINE FILE table entry or if no name is specified on the FILES control record, the Core Load Builder assigns an area in Working Storage for the Data File. The sector address of the Data File, relative to the start of Working Storage, is placed in word 5 of the DEFINE FILE table entry. This procedure is repeated for each 7-word DEFINE FILE table entry in the mainline program.

Use of the Core Image Buffer (CIB) and Working Storage

The Core Load Builder places in the CIB any parts of the core load which, when loaded, are to reside below location 4096. Any parts of the core load that are to reside above location 4095 are placed directly into core storage.

Enough Working Storage is reserved by the Core Load Builder to contain any Data Files assigned by the Core Load Builder to Working Storage. All the LOCAL subprograms and SOCALs, respectively, are stored in Working Storage following any files defined there. Figure 9 shows the distribution of a core image program between core storage, the CIB, and Working Storage. These diagrams depict a core image program just after it has been built but before it has been stored (STORECI).

Assignment of the Core Load Origin

The Core Load Builder origins core loads built from relocatable mainline programs at the next higher-addressed word above the end of the disk I/O subroutine to be used by the core load plus 30.

Disk I/O Subroutine in Core	Core Load Origin Decimal	Hexadecimal
DISKZ	510	/01FE
DISK1	690	/02B2
DISKN	960	/03C0

The origins for core loads built from absolute mainline programs are not controlled by the Core Load Builder. Therefore, the user must origin absolute mainline programs at 30 or more words above the end of the disk I/O subroutine to be used by the core load (these 30 words are required for the Core Image Header).

TRANSFER VECTOR

The Transfer Vector is a table included in each core load that provides the linkage to the subprograms. It is composed of the LIBF TV, the Transfer Vector for subprograms referenced by LIBF statements, and the CALL TV, the Transfer Vector for subprograms referenced by CALL statements.

Each CALL TV entry is a single word containing the absolute address of an entry point in a subprogram included in the core load that is referenced by a CALL statement. In the case of a subprogram referenced by a CALL statement but specified as a LOCAL, the CALL TV entry contains the address of the special LOCAL linkage instead of the subprogram entry point address. If SOCALs are required, the CALL TV entries for function subprograms contain the address of the special SOCAL linkage instead of the subprogram entry point address.

Each LIBF TV entry consists of three words. Word 1 is the link work in which the return address is stored. Words 2 and 3 contain a branch to the subprogram entry point. In the case of a subprogram referenced by a LIBF statement but specified as a LOCAL, the LIBF TV entry for its entry point contains a branch to the special LOCAL linkage instead of to the subprogram entry point address. The Core load Builder inserts the address of word 1 of the T. V. entry (link word) into Entry point +2 of the associated LIBF subroutine. If SOCALs are required, the LIBF TV entry for a SOCAL subprogram contains a branch to a special entry in the LIBF TV for the SOCAL of which the subprogram is a part. This special entry provides the linkage to the desired SOCAL subprogram.

SYSTEM OVERLAYS

SOCALs (system-overlays-to-be-loaded-on-call) are subprogram groups (by type and subtype) that are made into overlays by the Core Load Builder. They make it possible for many FORTRAN core loads that would otherwise not fit into core to be loaded and executed.

If, in constructing a core image program from a FORTRAN mainline program, the Core Load Builder determines that the core load will not fit into core, SOCALs are created by the Core Load Builder for the core load. In addition, the LOCAL/SOCAL flipper, which fetches the SOCALs when they are required during execution, is included in the core load along with the area into which the SOCALs are loaded (the SOCAL Area).

The SOCALs are created by subprogram type and subtype (see the description of program type and subtype under Disk System Format in Appendix C). The following table describes the SOCALs.

Subprogram Class	Type	Subtype	Overlay (SOCAL Number)
Arithmetic	3	2	1
Function	4	8	1
Non-disk FORTRAN I/O and "Z" conversion subroutines	3	3	2
"Z" device subroutines	5	3	2
Disk FORTRAN I/O	3	1	3

There are two SOCAL options. The Core Load Builder first attempts to make the core load fit into core by using SOCALs 1 and 2 only (option 1). If the core load still will not fit into core, SOCALs 1 and 2 and 3 are used (option 2). If the use of option 2 still does not make it possible for the core load to fit into core, an error message is printed (see Core Load Builder Error Messages, Appendix A).

Option 1 reduces the core requirement of the core load by an amount equal to the size of the smaller of the two SOCALs used, minus approximately 15 additional words required for the special SOCAL linkage. Option 2 reduces the core requirement by an amount equal to the sum of the sizes of the two smallest SOCALs minus approximately 20 additional words required for the special SOCAL linkage. SOCAL 2 is usually the largest SOCAL.

Each SOCAL does not contain all the available subprograms of the specified types and subtypes; only those subprograms of the specified types and subtypes required by the core load are contained in the SOCAL.

If a subprogram that would otherwise be included in a SOCAL is specified as a LOCAL subprogram, that subprogram is made a LOCAL and is not included in the SOCAL in which it would ordinarily be found.

SOCALs are never built for core loads in which the mainline program is written in Assembler or RPG language.

LOCAL/SOCAL FLIPPER (FLIPR)

The LOCAL/SOCAL flipper is included in each core load in which LOCAL usbprograms have been specified and/or in which SOCALs have been employed. If execution of the core load immediately follows the building of the core image program, this subroutine reads a LOCAL/SOCAL from Working Storage into the LOCAL/SOCAL Area as it is called during execution. If the core image program was stored in the User or Fixed Area in Disk Core Image format prior to execution, the flipper reads each LOCAL/SOCAL as it is called during execution from the User or Fixed Area (where it was stored following the core load) into the LOCAL/SOCAL Area.

The flipper is entered via the special LOCAL/SOCAL linkage. A check is made to determine if the required LOCAL/SOCAL is already in core. If it is not in core, the flipper reads the required LOCAL/SOCAL into the LOCAL/SOCAL Area, and transfers the LOCAL/SOCAL subprogram via the special linkage.

CORE IMAGE LOADER

The Core Image Loader serves both as a loader for core loads and as an interface for some parts of the Monitor system.

On any entry to the Skeleton Supervisor, the Core Image Loader is fetched and control is transferred to it. The Core Image Loader determines where the Skeleton Supervisor was entered, i.e., at $EXIT, $DUMP, or $LINK.

FETCHING THE SUPERVISOR

If an entry was made to the Skeleton Supervisor at the $EXIT entry point, the Core Image Loader first fetches the disk I/O subroutine used by the Monitor programs (DISKZ), if it is not already in core. It then fetches and transfers control to the Monitor Control Record Analyzer to read Monitor control records from the input stream.

If an entry was made to the Skeleton Supervisor at the $DUMP entry point, the Core Image Loader first saves words 6-4095 on the CIB and then fetches and transfers control to the DUMP program to perform the core dump according to the parameters specified. At the completion of the dump, the DUMP program either restores core from the CIB and transfers control back to the core load, or it terminates the execution with a CALL EXIT (see Terminal and Dynamic Dumps under Supervisor).

FETCHING A LINK

If an entry was made to the Skeleton Supervisor at the $LINK entry point, the Core Image Loader first saves low COMMON (locations 1536-1855 if DISKN is in core, locations 1216-1535 if DISK1 is in core, or locations 896-1215 if DISKZ is in core). It then determines from COMMA the lowest-addressed word of COMMON, if any, defined by the core load just executed. Any COMMON below location 4096 is saved in the CIB by the Core Image Loader.

Figure 10 illustrates the scheme used in saving COMMON between links.

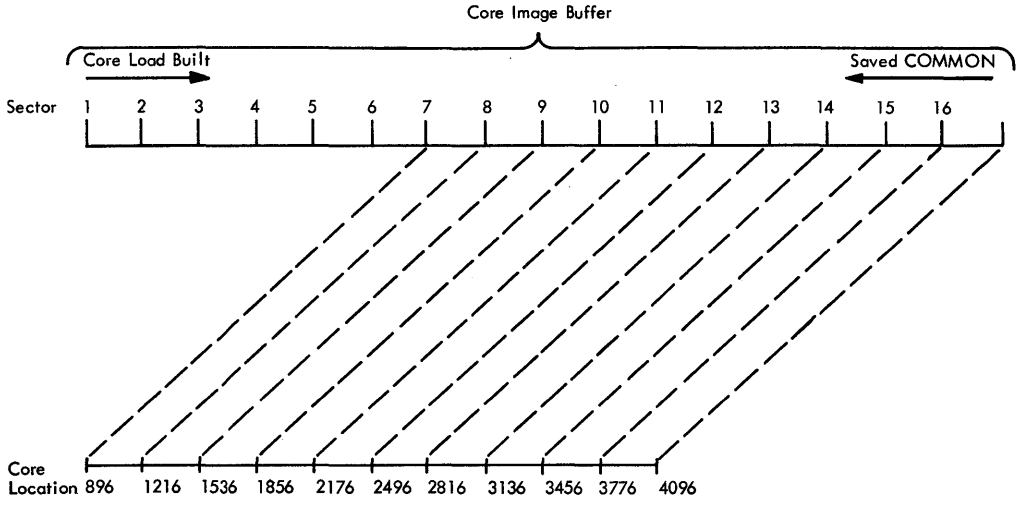


Figure 10. Scheme for Saving COMMON between Links

The LET/FLET entry for the link to be fetched is then located, and the Core Image Loader determines from it whether the link is in Disk Core Image format or Disk System format. If the link is in Disk Core Image format, the Core Image Loader fetches the disk I/O subroutine required by the core load, if it is not already in core. It next restores low COMMON if it lies within the COMMON defined by the core load just executed. The core load is then fetched and control is transferred to it.

If the link is in Disk System format, the Core Image Loader calls the Core Load Builder to construct a core image program from the mainline program. After the core image program has been built, the Core Load Builder returns control to the Core Image Loader, which then fetches the core load, as described above, and transfers control to it.

Figure 11 shows the layout of a core load loaded into core, ready for execution.

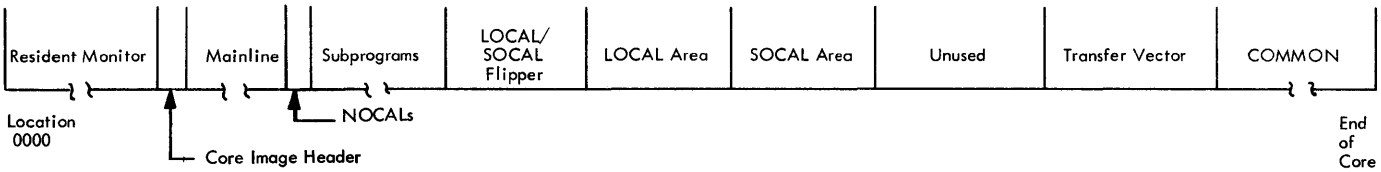


Figure 11. Layout of a Core Load Loaded for Execution

The information presented in this section should assist the user in achiever maximum utilization of the Monitor system.

STACKED INPUT ARRANGEMENT

Input to the Monitor System consists of control records, source programs, object programs, and data arranged logically by job.

The following points must be considered when arranging the input for any job.

1. Any number of comments records can be inserted in front of (but not immediately following) ASM, FOR, RPG, or XEQ monitor control records, and in front of an following JOB or DUP monitor control records.

2. Any records other than monitor control records which remain after the execution of an ASM, FOR, RPG, or XEQ subjob are passed until the next monitor control record is read. After a DUP operation , records are passed until either a monitor control record or another DUP control record is read.

3. If an error is detected in an assembly, FORTRAN compilation, RPG compilation or during loading From Disk System format, the resulting object program or any programs that follow within the job cannot be executed. Also, if an error is detected in an assembly, FORTRAN sompilation, RPG compilation, or during a loading from Disk System format during a STORECI function, all DUP functions are bypassed until the next valid ASM, FOR, RPG, or JOB record is read.

4. If the FORTRAN compiler, RPG compiler, or the assembler encounters a monitor control record, control will be transferred to the Supervisor, i.e., the monitor control record will be trapped. The Supervisor will correctly analyze the record after the compilation or assembly has been abandoned. DUP will not trap a monitor control record during a DUP operation (refer to DUP Control Records).

The stacked input arrangement shown in Figure 11.1 will assemble/compile, store, and execute both Programs A and C, providing there are no source program errors, and there is sufficient room in the Working Storage Area (refer to Working Storage Area). A source program error causes the DUP STORE operation (refer to DUP Control Records) to be bypassed for that program, and all following XEQ requests preceding the next JOB record are disregarded. Thus, if the successful execution of one program depends upon the successful completion of the previous program, both programs should be considered as one job and the XEQ control records should not be separated by a JOB record.

Job B calls in the Disk Utility Program, and stores object program B on disk.

USING THE DISK I/O SUBROUTINES

All core loads, whether they use disk I/O or not require one of the three disk I/O subroutines. As a minimum, this disk subroutine is used to read the core load into core and execute CALL EXIT, CALL LINK, CALL DUMP, and/or CALL PDUMP. Generally, DISKZ is used by FORTRAN and RPG core loads and DISK1 or DISKN by Assembler-Language core loads. DISKN provides faster operation than DISK1 for operations involving more than 320 words, as well as the simultaneous operation of disk drives. DISKZ is intended for use only in an error-free environment , because it does no preoperative parameter checking, whereas DISK1 and DISKN do. DISKZ also has a special calling sequence; DISK1 and DISKN have the LIBF calling sequence. Bear in mind that all three disk subroutines are assembled as mainlines and are thus not the same as programs stored in the System Library , even though DISK1 and DISKN (but not DISKZ) may be referenced with the LIBF statement. They are described with library subroutines because they are similar in some respects to library subroutines. Actually, they are neither incorporated into the core load like library subroutines nor are they stored in the System Library. A switch is set in COMMA to indicate which version of disk I/O is requested on the XEQ record. The setting of this switch is not altered

until 1) a Monitor control record is read or 2) a link that is stored in DCI is called. In the first case the switch is set to indicate DISKZ, unless the record was XEQ, in which case the switch is set to indicate whatever version is requested. In the second case the switch is set to indicate the version of disk I/O required by the link. In short, each DSF link except the first in an execution must utilize the same version of disk I/O as the preceding link. The first link must, of course, utilize the disk I/O specified on the XEQ record.

In order to save core in Monitor programs, all of which utilize DISKZ, DISKZ has been pared to a minimum. The following is a list of functions that are not available in DISKZ but are available in DISK1 and/or DISKN.

- No validity checking of the word count and sector address
- No file protection
- No LIBF type calling sequence
- No validity checking of the function indicator
- No write without readback check option
- No write immediate function
- Word count may not be on an odd boundary
- No simultaneous disk operations
- Does not "make" the sector gap when reading or writing more than 320 words

USING LINKS TO AVOID OVERPRINTING

To prevent overprinting in a link to another program, at least one space should be given prior to the linking. This is due to the fact that the Core Load Builder assumes that a space before printing is not necessary, since all monitor programs have a space after print.

THE USE OF SOCALs

Restrictions on Subroutines in SOCALs

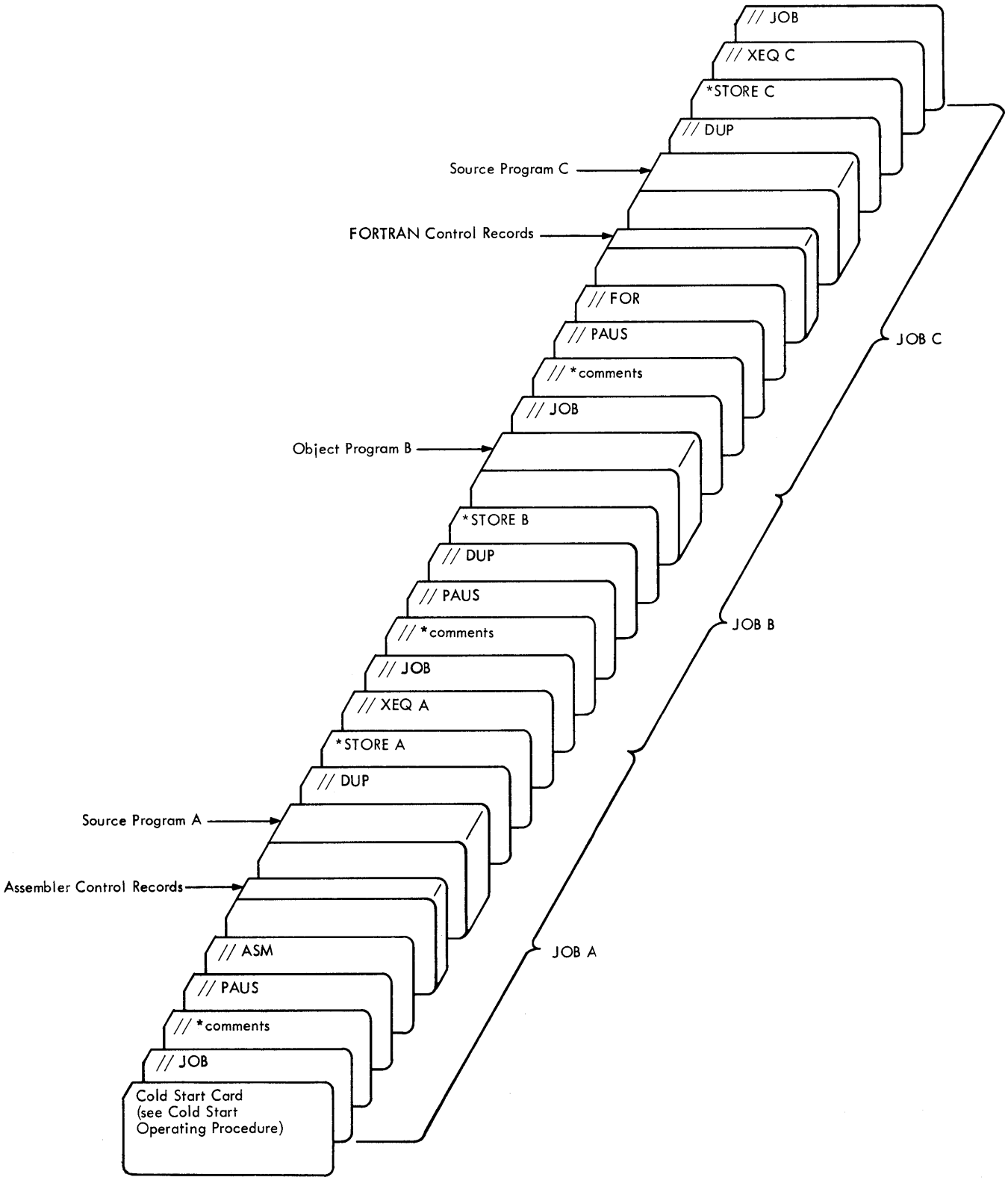
A rule of prime importance regarding subroutines in the SOCAL scheme is that none must cut across SOCALs. That is, a given subroutine that is in one SOCAL may not call a subroutine that is in another SOCAL or cause another SOCAL to be brought into core before the execution of the given subroutine is completed. This is due to the fact that the IBM-supplied 1130 subroutines that go into the SOCAL scheme are not re-enterable. It should also be noted that disk I/O is used every time a SOCAL is brought into core. This means that disk I/O will sometimes be entered without the user's direct knowledge.

When the 1627 Plotter is used in a program, the following subroutines must not be in a SOCAL for that program: EADD, FADD, FMPY, EMPY, XMD, XMDS and FARC. They must instead be in-core subroutines. This can be achieved by:

1. DUMP the programs to cards
2. DELETE the programs
3. STORE the programs with sub type zero

Decreasing Program Execution Time

When writing or modifying a program that is known to require SOCALs, planning is required to minimize the flipping of the various SOCALs in and out of core during execution. Ideally the program should be written in sections, each of which employs a single SOCAL, e.g., input, computation, and output. Even input and output should be carefully planned so as to separate disk and non-disk operations whenever possible.



● Figure 11.1 Example of Stacked Input (Three Jobs)

LOCAL CALLS A LOCAL

For the Assembler language programmer, it is possible to execute DSF core loads in which a LOCAL calls another LOCAL. This may be effected by punching column 26 of the XEQ record; this will cause all DSF core loads for that execution to allow LOCALs to call LOCALs. The user must make provision in all CALL LOCALs (Type 4 or 6 subroutines) in a given LOCAL-Calls -LOCAL chain to pass along the link word, which implies that all such subroutines must be written in Assembler language. This is necessary in order to return from the last LOCAL in the chain to the place from which the first LOCAL was called. There is no way to pass along the link word in a FORTRAN-written CALL subroutine, thus making this restriction necessary.

DISADVANTAGES OF STORING A PROGRAM IN DISK CORE IMAGE FORMAT (DCI)

Before deciding to convert a program to DCI, one should weigh the advantages gained in loading time against some disadvantages, one of the most important of which involves maintenance. Suppose, for example, that a DCI program contained a subroutine from the IBM-supplied System Library that contained an error that was fixed after the core load was built and stored. The correction would be sent out, but it would be applied only to the subroutine itself; it would not be applied to the DCI programs that have had the subroutine already built into themselves. Such programs, to acquire the fix, would have to be deleted and rebuilt (STORECI) after the maintenance mod was installed.

Another important consideration concerns core loads that contain references to non-Working Storage disk files. Of course, the system disallows STORECI if the core load references a file in the User Area, because the location (sector address) of that file may change (because of deleted programs). Any DCI core loads that reference such a file will do so by the old sector address, and the results are then unpredictable.

To a lesser extent the same danger exists if the DCI program references a file in the Fixed Area, even though that operation is allowed. The file may be deleted after the DCI program is stored, for example, and a new file or program stored in its place. This is complicated by the fact that not only are the sector addresses built into the DCI program, but also the logical drive codes, which implies that every time such a program is executed the user must be certain that all disk cartridges required are mounted on the same logical drives as when the program was originally stored in DCI.

TIPS ON MONITOR CONTROL

Temporary JOB Mode

In many cases DUP delete functions must be performed to clear the User Area of old programs before newly assembled or compiled programs may be stored. The necessity for such deletions is avoided by using the temporary mode when running jobs that contain programs that are likely to be replaced at a later time. In the Temporary mode all programs stored to the User Area are automatically deleted when the next JOB record is processed. This assures the user that his new program is the one stored in the User Area and is particularly useful while debugging.

EJECT Monitor Control Record

The EJECT record is used to control the beginning of a new page on the principal print device during a job. For example, messages to the operator of the Monitor Comments control record type may be placed in a more readable position if followed by an EJECT record.

```
// JOB
.
.
.
//* (message to operator)
// EJECT
// PAUSE
```

SYSUP

Changing cartridges in the middle of a job is permissible; however, great care must be exercised. A system update operation must always be performed. This function is provided by the System Library subroutine SYSUP. The subroutine should be called immediately following the loading of the new cartridge or cartridges as follows:

```
Label     Operation   F T  Operands & Remarks
          CALL        SYSUP   CALL DCOM UPDATE
          DC          ADDR
          .
          .
          .
ADDR      DC          /1wwwww (CART ID OF LOGICAL 0)
          DC          /1WWWWW (CART ID OF LOGICAL 1)
          DC          /1xxxxx (CART ID OF LOGICAL 2)
          DC          /1yyyyy (CART ID OF LOGICAL 3)
          DC          /1zzzzz (CART ID OF LOGICAL 4)
```

The IDs of cartridges to be used must be specified, even those that were previously specified except the master (logical 0), which if unspecified will be the previous master. Continuation of the job must be delayed until all the newly loaded cartridges are ready. Rules governing the specification of cartridges are the same as those for the JOB Monitor control record. record.

For a tip on using SYSUP see REELING, section Programming, Tips and Techniques.

The FORTRAN calling sequence for SYSUP is described in the section of this manual entitled System Library Utility Subroutines.

MAXIMUM PERFORMANCE OF HIGH SPEED DEVICES

Double Buffering

The 2501 Card Reader model A2, rated at 1000 cards per minute presents a special problem to the programmer who desires maximum performance from his card I/O operations. If any conversion of the card data is required, the reading speed is likely to drop to 500 cards per minute, unless double-buffering is used.

The principle involved, is to read into one buffer while converting and processing the data from another buffer. This scheme does cost core for the extra buffer and additional programming involved, but in most cases it should allow the card throughput for the 2501 to remain at 1000 cards per minute. The coding illustrated below shows the double-buffering technique used for reading cards from the 2501, and converting them to EBCDIC.

```
Label     Operation   F T  Operands & Remarks
          LIBF        READ0   PRIME DOUBLE-BUFFERED
          DC          /1000   *CARD READING. THIS READ
          DC          BUF1    *PERFORMED ONCE ONLY.
*
READ      LIBF        READ0   THIS READ WILL NOT START
          DC          /1000   *UNTIL PREVIOUS READ
SET1      DC          BUF2    *IS COMPLETED
*
          LIBF        ZIPCO   BR TO EXECUTE FAST CONVERT
          DC          /1101   IBM CARD CODE TO EBCDIC
SET2      DC          BUF1+1  INPUT AREA ADDRESS
SET3      DC          BUF1+1  OUTPUT AREA ADDRESS
          DC          B0      NO. OF COLUMNS TO CONVERT
*
          CALL        HLEBC   CONVERSION TABLE FOR ZIPCO
*
          LDD         BFADR
          STO         SET1    CHANGE READ BUFFER ADDRESS
          RTE         16      EXCHANGE BUFFER ADDRESSES
          STO         BFADR   *FOR NEXT TIME THRU LOOP.
          A           ONE
          STO         SET2    CHANGE INPUT AND OUTPUT BFR
          STO         SET3    *ADDRESSES FOR CONVERSION.
*
*
*  CODING FOR REQUIRED PROCESSING SHOULD FOLLOW
*
*
*
*
*
*
*
*
*
          B           READ
*
*
*  CONSTANTS AND WORK AREAS
*
*
*
ONE       DC          1       CONSTAINT VALUE OF 1
BUF1      DC          B0      WORD COUNT FOR CARD BFR 1
          BSS         B0      CARD BUFFER 1
BUF2      DC          B0      WORD COUNT FOR CARD BFR 2
          BSS         B0      CARD BUFFER 2
*
*
*  THE FOLLOWING PAIR OF ADDRESSES ARE EXCHANGED
*  EACH TIME THROUGH THE CARD READING LOOP.
*
*
          BSS         E       0       MAKE NEXT LOCATION EVEN
BFADR     DC          BUF1    ADDRESS OF CARD BUFFER 1
          DC          BUF2    ADDRESS OF CARD BUFFER 2
```

1403 Conversion Subroutines

Two subroutines are provided with the Monitor system that may be used by Assembler object programs to convert EBCDIC to 1403 Printer Code. These subroutines are EBPRT and ZIPCO.

Using the execution times listed in the Subroutine Library manual, the average time EBPRT requires to convert a 120 character line is 156 ms. This compares with an estimate of 72 ms per line for ZIPCO.

Considering that the available times on the 1403 Printer are

Model 6 (340 LPM): 176 ms/line
Model 7 (600 LPM): 100 ms/line

it would be difficult or impossible to run the printer at rated speed, depending on the model, using EBPRT. If overlapped I/O were attempted, it would be impossible to run either model at rated speed.

The assembly language programmer is therefore advised to use ZIPCO for all EBCDIC to 1403 Printer code conversions.

TIPS FOR ASSEMBLER LANGUAGE USERS

Grouping of Mnemonics

Assembler language programs can often be organized in such a manner as to improve the assembly time. The Assembler Program is divided into overlay phases, each phase processing a certain group of mnemonics. By grouping mnemonics of a common type in the source program, fewer disk reads of overlay phases will be required by the Assembler. The following is a list of the mnemonics as they are grouped within the Assembler program:

A. ABS, FILE, ENT, ISS, ILS, SPR, EPR
B. DCs and imperative instructions (A, LD, EOR, BSC, etc.)
C. DEC and XFLC
D. DMES
E. HDNG, ORG, EQU, BSS, BES, LIST, SPACE, EJCT, DUMP, PDMP
F. LIBF, CALL, DSA, LINK, EXIT, EBC, DN

Each time a mnemonic is encountered during the assembly process, the overlay phase required to process it will be read into core, unless it is already residing in core.

Intermediate I/O

As the source records are read and processed by the Assembler in Pass 1, each statement is packed and saved on the disk in Working Storage. The part of the record that is saved is from column 21 to the last non-blank column. If no listing is specified, comments records are not saved on the disk.

Each record saved on the disk is preceded by a prefix word that contains the length of the associated record plus one. Up to sixteen 38-column records are saved on one sector.

WRITING ISS AND ILS

Interrupt Service Subroutines

The following rules must be adhered to when writing an ISS:

- Precede the ISS statement with an LIBR statement if the subroutine is to be called by LIBF rather than CALL.
- Precede the subroutine with an EPR (extended) or an SPR (standard) statement if precision specification is necessary.
- Precede the subroutine with one ISS statement defining the entry point (one only), the ISS number, and the ILS subroutines required. The device interrupt level assignments, and the ISS numbers used in the IBM-provided ISS and ILS routines, are shown in Table 4. See the 1130 Assembler Language Manual (Form C26-5927), for a description of the ISS statement. Note that the ISS numbers assigned by the IBM-supplied subroutine range from 1-11. ISS numbers 12-20 are assignable by the user. (They should be assigned from 20 downwards.)
- When assembling the ISS, an *LEVEL n control card must be included for each interrupt level associated with the device.
- The entry points of an ISS are defined by the related ILS. This must be taken into consideration when a user-written ISS is used with an IBM supplied ILS. The ILS executes a Branch and Store I instruction to the ISS at the ISS entry point plus n (see Table 4). The ISS must return to the ILS via a BSC instruction (not a BOSC).

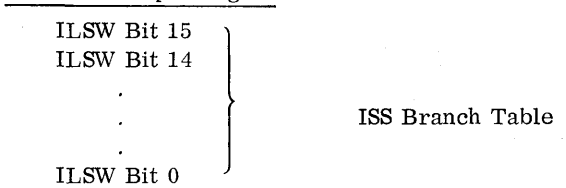
Table 4. ISS/ILS Correspondence

ISS Number	Device	Device Interrupt Level Assignments	n
1	1442 Card Reader Punch	0,4	+4, +7
2	Input Keyboard/Console Printer	4	+4
3	1134/1055 Paper Tape Reader/Punch	4	+4
4	2501 Card Reader	4	+4
5	Disk Storage	2	+5
6	1132 Printer	1	+4
7	1627 Plotter	3	+4
8	Synchronous Communications Adapter	1	+4
9	1403 Printer	4	+4
10	1231 Optical Mark Page Reader	4	+4
11	2250 Graphic Display	3	+4

Interrupt Level Subroutines

An ILS is included in a core load only if requested by an ISS that is a part of the same core load. ILS02 and ILS04 are a part of the Resident Monitor unless they are deleted from the System Library and replaced with user-written subroutines. The following rules must be adhered to when writing an ILS.

- Precede the subroutine with an ILS statement to identify the interrupt level involved.
- Precede all instructions by an ISS branch table and include one word per ILSW bit used. If the ILSW is not to be scanned, (i.e., a single ISS handles all interrupts on the level), then a one word table is sufficient. The minimum table size is one word. Table words must be non-zero. A zero must follow the branch table.



The ISS branch table identifies both the ISS subroutine and the point within the ISS which should be entered for each bit used in the ILSW. The actual linkage is generated by the Core Load Builder. Basic to this generation is the ISS number implied by bits 8-15 of the branch table word and specified in the ISS statement. This number identifies a core location in which the Core Load Builder has stored the address of the called entry point in the ISS. This entry point address is incremented by the value in bits 0-7 of the branch table word, producing the interrupt entry point address. The Core Load Builder replaces the ISS branch table word with the interrupt entry point address.

During execution, each address in the branch table may be used with an indirect branch and store I (BSI) instruction to reach the ISS corresponding to that ILSW bit position. The ILSW bit that is ON can be determined by the execution of a SLCA instruction. At the completion of this instruction, the index register specified contains a relative value equivalent to the bit position in the ISS branch table. An indirect, indexed BSI may then be used to reach the appropriate ILS.

Before processing by the Core Load Builder, each word in the ISS branch table has the following format:

Bits 0-7 -- Increment added to the entry point named in the ISS statement to obtain the interrupt entry point in the ISS for this ILSW bit. (In IBM-written ISS subroutines, this increment is +4 for the primary interrupt level and +7 for the second interrupt level.)

Bits 8-15 -- @ISTV+ the ISS number for the ISS subroutine for this ILSW bit.

- The ISS number for any entries in the IBT that represent unused bits in the ILSW must have the value @ ISTV.
- The ILS entry point must immediately follow the ISS branch address table and must be loaded as a zero. The Core Load Builder assumes that the first zero word in the program is the end of the branch table and is also the entry point of the ILS. (The table must contain at least one entry.) The interrupt results in a BSI to the ILS entry point.
- To clear the level, a user-written ILS, used with an IBM-supplied ISS, should exit via the return linkage with a BOSC instruction.
- User-written ILS must replace the equivalent IBM-supplied ILS. The user written ILS must be stored as ILS0x, where x = 0, 1, 2, 3, 4, or 5.
- The IBM-supplied ILS02 and ILS04 subroutines are stored as subtype 1. User-written replacements must be stored as subtype zero.
- The branch table for ILS04 may have no more than 9 entries.
- The branch table for the IBM-supplied version of ILS04 may have no more than 9 entries. A user-written version may support all 16 possible entries.

READING A CORE MAP AND A FILE MAP

The core maps described below are taken from the sample programs supplied with the Monitor system. (The sample program listings and operating instructions are printed in Appendix J.)

The core map for the Assembler-language sample program indicates that there were /7904 words of core storage not occupied by the core load (R41 is an informational message, not an error message). There was only one CALL (FSQR), but there were several LIBFs, e.g., FARC. The ILS02 and ILS04 subroutines are required; however, their addresses indicate that they are a part of the Resident Monitor and not in the core load proper. The entry point to the mainline program is /01FE.

The principal difference in the core map printed for the FORTRAN-language sample program is that it includes a file map. The file defined as file number 103 has been equated to a data file named FILEA, which begins at sector /01AE, is one sector in length, and is stored on a cartridge labeled 000F.

If file 103 had required more than the two sectors available in FILEA, the record count would have been reduced to make the file fit in FILEA, and the file map entry would have been

```
103 01AE 0002 000F FILEA TRUNCATED
```

The files defined as 101 and 102 are files in Working Storage because they do not appear in the *FILES record. This can be determined by looking at the rightmost entry in the file map. For files defined in the User/Fixed Area, for example, FILEA, this entry is the name of the file; otherwise, it is the address of Working Storage.

The second entry for a User/Fixed Area file is the absolute sector address of the first sector of the file. For files in Working Storage, this address is relative to the first sector of Working Storage. Thus, the absolute sector address of the first sector of file 101 is /0000 + /01B0; for file 102 it is /0001 + /01B0.

Note that the 4K example requires both LOCALs and SOCALs. The LOCALs were, of course, requested by the user, and the core map entries for the LOCAL subroutines (FLOAT, FARC, and IFIX) have been flagged. The presence of SOCALs, which were selected and constructed by the Core Load Builder, are also indicated by flags on the core map entries for the subroutines included in the various SOCALs. The number following the word "SOCAL" indicates in which of the SOCALs a particular subroutine is to be found. In this case SOCAL Option 2 was employed. This can be deduced from the fact that there are three SOCALs. Option 1 consists of only two.

Several other facts about the 4K core load can be extracted from the core map. For one thing, the core load exceeds the capacity of core storage (before SOCALizing) by /03AB words (message R40). Furthermore messages R43, R44, and R45 indicate that SOCALs 1, 2, and 3 require /0124, /06AC, and /02A2 words of core, respectively. This information indicates that, for example, since SOCAL 2 is much larger than SOCAL 1, more arithmetic and function subprograms may be called at little extra cost in core. (It would be necessary to reduce the dimension of the variable B to realize this.) Message R41 says that, after SOCALizing, there are only /0004 words of core that are not used by this core load.

The RPG core map shows that the x version of the ILS subroutines have been used. The x versions are required by RPG and are called by punching any character in column 28 of the // XEQ card.

Assembler Core Map

```
// XEQ        L
R 41  7904 (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR
 FSQR   0248
LIBF TRANSFER VECTOR
 FARC   069E
 XMDS   0682
 HOLL   0632
 PRTY   05E2
 EBPA   0592
 FADD   04E1
 FDIV   0540
 FLD    048C
 FADDX  04E7
 FMPYX  04A2
 FSTO   0470
 FGETP  0456
 NORM   042C
 TYPEO  0312
 EBPRT  02AC
 IFIX   0280
 FLOAT  0230
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  00B3
      01FE (HEX) IS THE EXECUTION ADDR
```

FORTRAN Sample 4K Core and File Map

```
// XEQ        L  2
*LOCAL,FLOAT,FARC,IFIX
*FILES(103,FILEA)
FILES ALLOCATION
  103 01AE  0001  000F FILEA
  101 0000  0001  000F  01B0
  102 0001  0001  000F  01B0
STORAGE ALLOCATION
R 40  03AB (HEX) ADDITIONAL CORE REQUIRD
R 43  0124 (HEX) ARITH/FUNC SOCAL WD CNT
R 44  06AC (HEX) FI/O, I/O SOCAL WD CNT
R 45  02A2 (HEX) DISK FI/O SOCAL WD CNT
R 41  0004 (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 EBCTB  0F53 SOCAL 2
 HOLTR  0F17 SOCAL 2
 GETAD  0ED4 SOCAL 2
 XMDS   09B2 SOCAL 1
 HOLEZ  0E9E SOCAL 2
 NORM   07DC
 FADDX  095D SOCAL 1
 FSBRX  0934 SOCAL 1
 FMPYX  0900 SOCAL 1
 FDIV   08AE SOCAL 1
 FSTOX  0788
 FLDX   07A4
 SDCOM  0920 SOCAL 3
 SDFX   08E6 SOCAL 3
```

```
SDWRT  0954 SOCAL 3
SIOFX  099A SOCAL 2
SUBSC  078E
SIOI   099E SOCAL 2
SCOMP  0982 SOCAL 2
SWRT   08AB SOCAL 2
SRED   08B0 SOCAL 2
FSTO   078C
FLD    07A8
PRNTZ  0DE0 SOCAL 2
CARDZ  0D36 SOCAL 2
SFIO   09AD SOCAL 2
SDFIO  0959 SOCAL 3
IFIX   087C LOCAL
FARC   087C LOCAL
FLOAT  087C LOCAL
SYSTEM SUBROUTINES
ILS04  00C4
ILS02  00B3
ILS01  0F5A
ILS00  0F75
FLIPR  0816
     04DD (HEX) IS THE EXECUTION ADDR
```

FORTRAN Sample 8K Core and File Map

```
// XEQ        L  1
 FILES(103,FILEA)
FILES ALLOCATION
  103 01AE  0001 000F FILEA
  101 0000  0001 000F  01B0
  102 0001  0001 000F  01B0
STORAGE ALLOCATION
R 41  0C9C (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 EBCTB  12CD
 HOLTB  1291
 GETAD  124E
 NORM   1224
 XMDS   1208
 FARC   11E6
 HOLEZ  11B0
 FLOAT  11A6
 IFIX   117A
 FADDX  1125
 FSBRX  10FC
 FMPYX  10C8
 FDIV   1076
 FSTOX  101E
 FLDX   103A
 SDCOM  07FE
 SDFX   07C4
 SDWRT  0832
 SIOFX  0B1A
 SUBSC  1054
 SIOI   0B1E
 SCOMP  0B02
 SWRT   0A2B
 SRED   0A30
 FSTO   1022
 FLD    103E
 PRNTZ  0F60
 CARDZ  0EB6
 SFIO   0B2D
 SDFIO  0837
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  00B3
 ILS01  12D2
 ILS00  12ED
     04DD (HEX) IS THE EXECUTION ADDR
```

```
// XEQ        L              R
R 41  73FA (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR
 RGERR  0B64
 EBPT3  099C
 HLEBC  07F2
LIBF TRANSFER VECTOR
 RGMV2  0AB0
 RGMV1  0A1C
 PRNT3  0872
 ZIPC0  0752
 READ0  06F2
SYSTEM SUBROUTINES
 ILSX4  0BB3
 ILSX2  0BD5
     020F (HEX) IS THE EXECUTION ADDR
```

LOCATING FORTRAN ALLOCATION ADDRESSES

The variable allocations listed below are taken from the FORTRAN sample program in Appendix J.

```
VARIABLE ALLOCATIONS
     A(R )=00DC-0016      X(R )=00F0-00DE      B(R )=0208-00F2
    V3(I )=020E           M(I )=020F           L(I )=0210
    L2(I )=0214          N1(I )=0215          N2(I )=0216
     K(I )=021A          IK(I )=021B          I1(I )=021C


     D(R )=020A          V1(I )=020C          V2(I )=020D
    M1(I )=0211          M2(I )=0212          L1(I )=0213
     N(I )=0217           I(I )=0218           J(I )=0219
```

The variable array A is to be found between core locations /00DC + /001E + $ZEND and /0016 + /001E + $ZEND, inclusive. That is, A_1 is at /00DC + /001E + $ZEND, A_2 at /00DB + /001E + $ZEND, etc. The /001E term is the length of the Core Image Header and $ZEND is the address of the first core location following DISKZ.

The other allocation addresses, e.g., statement allocation, may be calculated in a similar manner.

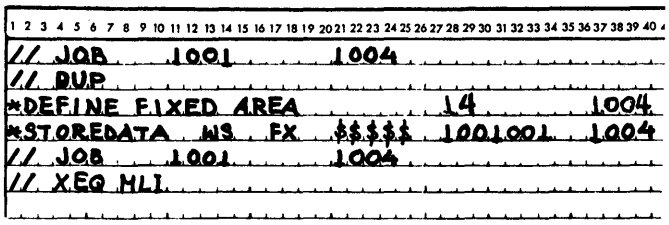
INITIALIZING $$$$$ DATA FILES FOR USE WITH FORTRAN UNFORMATTED I/O

The user must define a Data File with the name $$$$$ prior to executing a FORTRAN mainline program or subroutine that uses unformatted I/O. This Data File must be located in the Fixed Area. One file may be defined in the Fixed Area of each cartridge on the system; however, only one $$$$$ file may be referenced in any one job.

The following example shows a $$$$$ file being defined on a satellite cartridge.

The satellite cartridge ID is 1004
The system cartridge ID is 1001
A file of 100 sectors is desired

After the file is defined, program ML1 which uses unformatted I/O can be executed. Note that no *FILES card is required at execution time to define the $$$$$ file.



Sample program 3, containing the statements END FILE, BACKSPACE, and REWIND, is included in Appendix J. The program writes three logical records of different lengths to file $$$$$. Each logical record begins at a sector boundary and extends into additional sectors as required. Refer to unformatted disk I/O description in Table 3.

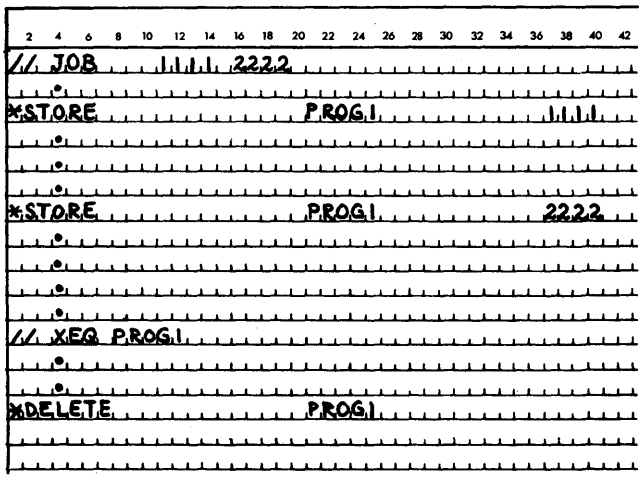
After completion of each WRITE (of records A, B, and C), a pointer is "moved" to the beginning of the next logical record. In the case of the END FILE statement, the pointer is similarly positioned beyond the record generated by END FILE. The first BACKSPACE statement moves the pointer to the beginning of record C, which is subsequently read into area F. Following the REWIND statement, which sets the pointer to logical record A, a READ with no area specified is executed that has no effect except to advance the pointer to record B. Only the first half of B is read into E, since the record lengths are in the ratio 2 : 1.

USE OF DEFINED FILES

When an *FILES Supervisor control record is used following a //XEQ Monitor control record, or a * STORECI control record, the Core Load Builder attempts to locate the file name by searching LET or FLET. If the name is found, the sector address of this Data File is inserted in the file table (created as a result of the FORTRAN DEFINE FILE statement or the Assembler FILE mnemonic) identified by the file number specified on the *FILES record. If the file name is not found in LET and FLET, the Core Load Builder causes this file to be a Working Storage file. A suggested way of initially allocating a disk area for a Data File is to perform a * STOREDATA DUP operation from Working Storage to the User or Fixed Area. The number of sectors stored should be determined on the basis of the number of records the file is to contain, and the size of each record. Note that records do not continue across sector boundaries. Once the number of sectors required has been determined, this number of sectors should be specified on the * STORE DATA control record, provided the User Area or Fixed Area is large enough to contain this file.

DUPLICATE PROGRAM AND DATA FILE NAMES

On a multi-drive system, it is possible to have more than one program or data file with the same name. This can cause problems when attempting to execute or delete the named program.
Example:



This sequence of instructions will cause PROG1 on the cartridge labeled 1111 to be executed when PROG1 on 2222 may have been desired. A similar problem can occur on a DELETE operation. The above DELETE instruction would delete the program on 1111, not the one on 2222.

The answer to this problem is to avoid having two programs or data files with the same name. If an unknown cartridge is on line and it is not needed for the job, disable it.

NAME CONFLICTS

In STORE and DELETE operations, care should be taken to avoid name conflict with IBM-supplied programs. If the program to be stored or deleted carries the same name as an IBM program, the system may execute the operation on the wrong program.

RESTORING DESTROYED CARTRIDGES

Cartridges that contain data and/or programs in the User Area or Fixed Area, and which may be difficult to replace, can sometimes be restored to use after being rendered unusable. If only sector addresses are affected, DCIP may be used to initialize sector addresses only. (See "Stand-Alone Utility Programs").
If some part of the Monitor System has been destroyed (including LET, FLET, User Area and Fixed Area), a reload function may be performed with the System Loader. In this case, the entire Monitor System deck, except the System Library, should be processed by the System Loader.
In cases in which individual words on the disk have been destroyed, the disk patch feature of DCIP may prove very useful in restoring these words.

REELING

"Reeling", that is, continuing a long data file from one cartridge to other cartridges can be done with the aid of SYSUP and linking. Such an operation might be performed as follows. Suppose a single-drive user intends to process sequentially a long data file that will not fit on one cartridge. The first part of the file can be defined on one cartridge and the second part on another. The programs that access this file can be written as two links, the first of which processes the first part of the file, the second the rest of the file. Assuming that the program is written in FORTRAN, the termination of the first link would consist of a PAUSE (to allow for mounting the second cartridge in place of the first), followed by CALL SYSUP, followed by CALL LINK to the second link, which then processes the second part of the data file. When SYSUP is called, DCOM and COMMA are changed to reflect certain IBM System Area allocations of the second cartridge.

The only constraint is that the second cartridge must be a system cartridge. If the FORTRAN Compiler is not present on the second cartridge, the second link to be executed can be compiled on the first cartridge, dumped to cards, and stored as described on the second cartridge. The sample program 5 in Appendix J illustrates how this could be accomplished. In the sample, both cartridges are system cartridges and both contain a Fixed Area but only cartridge llll includes the FORTRAN Compiler. The second link was compiled, dumped to cards, and stored on cartridge 2222, which contains the second part of the file. A terminating pattern (decimal 999) is written at the end of the file on each cartridge and could be used as an end-of-file indicator for subsequent operations on these files.

One-word integers were specified in LINK1 so that the next higher core location after L (2) would be zero as a result of the L (1) equal 0 statement. Since the array is stored in reverse order, SYSUP will find cartridge ID 2222 (hex) followed by 0. Refer to SYSUP, in "Monitor System Library". Another method, suitable to any FORTRAN precision, would utilize a call to an Assembly language subroutine with undefined precision that subsequently calls SYSUP and includes the SYSUP list or array.

The sample program 6 in Appendix J illustrates sequential file processing with two cartridges and two disk drives. The multiple-drive user may avoid the SYSUP/CALL LINK by naming both cartridges on the // JOB record. Just as in the previous description, his program must be written so as to process the two portions of the file separately, even though they may both have the same name. In such a case, the *FILES record could name two files, both with the same name but with different cartridge IDs.

All files referenced in a given core load must be identified in LET/FLET by the Core Load Builder when the core load is built. This applies alike to DSA and *FILES references. If desired, the program may be divided into links, each with its own associated file. If sufficient drives are not simultaneously available for all cartridges involved to be specified, a reeling method must be used whereby any cartridge with a data file named in an *FILES record must be on-line at the time the *FILES record is processed as a result of either a // XEQ or *STORECI record. Similarly, a DCI program that accesses files in a Fixed Area must be executed in the same disk cartridge environment in which it was built. For example, if the sample program 5 in Appendix J were stored in DCI format with cartridge llll on logical drive 0 and cartridge 2222 on logical drive 1, it must be executed with those same cartridges on those same logical drives. These requirements are due to the fact that the Core Load Builder must assign absolute sector addresses, including logical drive codes, for UA/FXA files as the core load is built, and of course it must find these addresses in LET/FLET:

MAINLINE PROGRAMS THAT USE ALL OF CORE

Before writing a program that occupies all or nearly all of core, the user should weigh the advantage gained against the possible later rewriting required if IBM-supplied subroutines used by the core load are expanded due to modification.

TIPS FOR FORTRAN LANGUAGE USERS

It is strongly recommended that the use of the 1130 device code be avoided in READ and WRITE statements, the use of integer variables in such cases allows for easier modification.

CONVERTING FROM VERSION 1 TO VERSION 2

- Data files and DSF programs must be dumped from the Version 1 cartridge to cards or paper tape and stored on the Version 2 cartridge under DUP control,

- The five-character alphameric disk cartridge labels used in Version 1 must be changed to four-character hexadecimal labels.

- DCI programs on a Version 1 cartridge cannot be dumped and stored on a DM2 cartridge. They must be stored in DSF on the DM2 cartridge and then converted to DCI under DUP control, which implies that the original DSF mainlines must be available.

- Since the FORTRAN I/O device numbers are fixed, some reprogramming is necessary whenever programs that use a given I/O device are required to employ a different device. For example, if a program that uses an 1132 Printer is to use a 1403 Printer, the device numbers must be changed in all READ and WRITE statements that reference the printer.

For another solution see tips on EQUAT record over page.

TIPS FOR USE OF EQUAT RECORD

The *EQUAT function is used to change LIBF/CALL references in core loads that are to be built, without the necessity of recompiling or reassembling them. For example, suppose that one has a FORTRAN mainline that prints on the 1132, but it is desirable to have that print out on the 1403 instead. Without the *EQUAT function it would be necessary to change the *IOCS record and recompile the program to change from 1132 to 1403 printout. With it one has only to specify on the *EQUAT record that PRNZ (the 1403 subroutine is to be substituted, when the core load is built, for PRNTZ (the 1132 subroutine) . In such cases the Core Load Builder compares each call it encounters with the name(s) on the left of the equal sign (=) in the *EQUAT record. Every time a match is made the Core Load Builder substitutes the name on the right of the equal sign for the name it has encountered in the DSF code. Note that the *EQUAT function is associated with the JOB record, which implies that all core loads that are built in a given job will be built from the same substitution list.

The use of this function is not restricted to I/O substitutions. One might, for example, have several versions of a subroutine, each stored under a different name. With this function the effects of using each of these subroutines in a core load could easily be observed without resorting to recompile/reassemble the calling program(s) .

The user should bear in mind that the calling sequences of any substitute pair, i.e., the subroutines named on either side of an equal sign, must be identical, since the Core Load Builder does no more than substitute one call for the other. Thus, CARDZ could not be substituted for PRNZ because the 120-word count associated with PRNZ is incompatible with the 80-column count associated with CARDZ. The FORTRAN user must also become aware of the subroutines that are evoked from *IOCS record entries. The entries and the ISS subroutines they imply are given below:

ENTRY	CORRESPONDING SUBR: CALL
CARD	CARDZ
2501 READER	READZ
1442 PUNCH	PNCHZ
TYPEWRITER	TYPEZ
KEYBOARD	WRTYZ
1132 PRINTER	PRNTZ
1403 PRINTER	PRNZ
PAPER TAPE	PAPTZ
PLOTTER	PLOTX
DISK	DISKZ
UDISK	DISKZ

Another example of the use of the *EQUAT function is where a FORTRAN program does its printing on the 1132, and this program also exercises the Synchronous Communication Adapter. These two I/O devices cannot be overlapped unless the 1132 is serviced by PRNT2. By using the *EQUAT function to change PRNTZ (the Subroutine used by FORTRAN I/O for 1132 printing) to the name PRTZ2 (a special subroutine to interface between PRNTZ and PRNT2), 1132 printing can then be performed by PRNT2, and thus overlapped with the SCA.

DUMPING DATA FILES TO CARDS AND RESTORING

It is often advisable to dump important data files to cards so that they may be restored if the cartridge containing them is destroyed. DUP punches sequence numbers in cc 73-80 of the data cards as the file is dumped. The numbers start with one and are incremented by one with each card. The last sequence number, then, is actually the number of data cards dumped, a parameter that is required on the STOREDATA card when restoring the file to the UA/FXA.

COPYING AND INITIAL LOADING DISK CARTRIDGES

Prior to the introduction of the improvements in the System Loader and DCIP in Modification 2, much time was wasted in copying to a cartridge that contained a faulty table of defective cylinders or invalid cartridge ID. The same was true in initial loading such cartridges. The improvements in the two programs include the setting and checking of a word in sector @IDAD that tells the kind of cartridge it is. This word, @DTYP, contains minus one if it is an initialized DM1 cartridge minus two if it is a DM2 non-system cartridge, a zero if it is a DM1 system cartridge, and a plus two if it is a DM2 system cartridge. This word is checked by DCIP before a copy is performed. DCIP expects to copy from a system cartridge (zero or +2) to a non-system cartridge (-1 or -2). The card System Loader checks to see that this word indicates a DM2 system cartridge (+2) before performing a reload operation. The user should use the disk patch facility in DCIP to update word @DTYP to the proper value for all cartridges built with the aid of DPIR (DM1) or the first release of DCIP. The System Library programs COPY and DISC perform as the copy and initialize portions, respectively, of DCIP with respect to word @DTYP.

CORE LOADS UTILIZING LOCALs

Core loads that utilize LOCALs will not necessarily run the same way with LOCALs as they would on a larger machine without LOCALs. This is due to the fact that every time a LOCAL is fetched, it is fetched in its initial state from the disk. Thus, unless it comprises read-only code, it will execute differently the second and subsequent times it is fetched. This same rationale applies equally well to SOCALs, at least in theory, but the IBM-supplied subroutines that are stored with SOCAL subtype codes are read-only code.

USE OF INDEX REGISTER 3

Unless a special set of interrupt level subroutines is used (see // XEQ) , it is not possible to use index register 3 under certain conditions, even if it is saved and restored. The conditions include core loads that overlap I/O operations and core loads that use the Synchronous Communications Adaptor. In general, this register is reserved to point to the Transfer vector.

DISK FILE ORGANIZATION AND PROCESSING

The disk I/O subroutines supplied with RPG: direct access, sequential access, and Index-Sequential Access Method (ISAM), can be used by RPG and Assembler language programmers. The key to the use of the disk I/O subroutines is an understanding of the basic principles of disk file organization and disk file processing.

File Organization

File Organization is the method of arranging data records on a direct access storage device, i.e., building the file.

The two types of file organization available with DM2 are sequential and indexed-sequential (ISAM).

Sequential File Organization. A sequential organized file is one in which records are placed on the disk in the same order they are read in, one after another. Card files are always organized this way. That is, record six cannot be written until record five is written, record five until record four, etc. Sequential files may be processed sequential or randomly.

Index-Sequential (ISAM) File Organization. An indexed-sequential file is one in which records are placed on the disk in ascending collating sequence by record key. This key may be a part number, man number or any other identifying information that is present in the records on the file. In addition, the indexed-sequential file uses an index to indicate to the processing program the general location of the desired records. Each index entry contains a cylinder address and the highest record key on that cylinder. All index entries are formed into an index table. For cylinders that have overflowed, the index also contains the overflow sector address and key of the first sector overflowed from that cylinder.

Index tables are analogous to the index card file in a library. If you know the name of a book (record key), you can look in the card file (index table) until you find the card (entry) for that book. On the card you will find a number (cylinder address) where the book (record) is located. You go to the shelf (seek) and find the number (cylinder address) you are look ing for. Now you can search for the particular book (record) by title (record key).

Records on an indexed-sequentially organized file may be processed sequentially or randomly.

File Processing.

File processing is the method of retrieving data records from the file, i.e., using the file.

Four methods of file processing are available with DM2 RPG:

1. Sequential processing of sequentially organized files
2. Random processing of sequentially organized files
3. Sequential processing of indexed-sequentially organized (ISAM) files
4. Random processing of indexed-sequentially organized (ISAM) files.

Sequential Processing (Sequential Files). All records in the file are processed in order starting with the first physical record in the file.

Random Processing (Sequential Files). In random processing the sequence of record processing is not related to the physical sequence of records on the file. To find a record in a sequentially organized file, the record number must be supplied to the program. The record number indicates the relative position (sequential location) of the record in the file. The disk I/O routine calculates the sector address from the record number and reads the proper record.

Sequential Processing (Indexed-Sequential Files). All records in an ISAM file are available in a sequence determined by record key. Processing may start at the beginning of the file or at any point within the file.

Random Processing (Indexed-Sequential Files). To find a random record in an ISAM file, the files index is searched using the record's key. The matching entry in the index points to the cylinder containing the record. That cylinder is then searched for the desired record. The match is again made by record key. This kind of processing may be called processing in a random sequence with record keys.

CALCULATING SEQUENTIALLY ORGANIZED AND ISAM FILE SIZE

The file is initially established on the disk by using a DUP STOREDATA function. STOREDATA sets aside a specified number of sectors for the file and enters the file name in LET or FLET. This file name must be used in all future references to this file.

Sequentially Organized Files

The number of sectors needed for a file depends on record size and number of records. The records are fixed length and can be defined as any size between 320 words (640 characters) and 1 word (2 characters). Note that records cannot extend across sector boundaries. Thus a 320 word record (one sector) and a 161 word record would each require one sector of disk space. Careful planning is required in calculating optimum record size for your file.

Output file size in sectors=number of records /(number of records that can be contained in each sector).

Output record size in words=numbers of records+1/ (number of records that can be contained in each sector).

Note 1. If the above formulae produce answers with fractional parts, the file and record size are obtained by rounding off to the next higher whole number.

Note 2. The number of records that can be contained in each sector (the denominators of the first two formulae)=320/(record size in words). The remainder, if any must be ignored. 320 is the number of words per sector.

To change record sizes or add records to a sequential file the file must be rebuilt. If the revised file requires additional sectors it must be redefined and rebuilt. A sequentially organized file is built using the sequential access routine. It may be processed by either the sequential access routine or the direct access routine. These routines are described in the 1130 Subroutine Library manual, Form C26-5929.

ISAM Files

The number of sectors required for an ISAM file is computed by the following formula. (The remainder in all cases should be disregarded.)

Prime data sectors + Index sectors + Overflow sectors + 1 (File label)

Where:

Prime data sectors =

$$\frac{\text{Approximate number of records in file + number of records per sector - 1}}{\text{Number of records per sector}}$$

$$\text{Number of records per sector} = \frac{320}{\text{Record size} + 2}$$

The maximum record size is 318 words. Records cannot cross sector boundaries.

Index sectors =

$$\frac{\text{Number of prime data cyliners + number of index entries per sector - 1}}{\text{Number of index entries per sector}}$$

Number of prime data cylinders =

$$\frac{\text{Number of prime data sectors} + 7}{8}$$

Number of index entries per sector =

$$\frac{320}{\text{Index entry size}}$$

Index entry size = 2 (key length in words) + 3
Key length is a maximum of 25 words (50 characters). If the length of the key in characters is odd add one when calculating the number of words, i.e., 49 characters require 25 words.

Overflow sectors = The number of sectors the user wishes to allot to record overflow before the file must be rebuilt. The overflow area is automatically assigned to start at the sector following the last sector of prime data. This assignment is done by the ISAM load (close) routine.

When computing file size always add one sector for the file label.

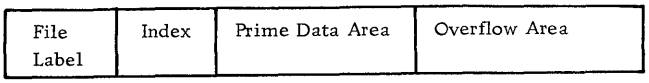
If desired, an assembler language program can be used to perform the above calculations. The programmer need only know the index entry size (calculation shown above), the length of a record in words, the approximate number of records in the file and an estimate of the number of sectors of overflow area needed.

A program to calculate all values computed above is included in this manual as sample program 7 in Appendix J. The values calculated by the program or by the manual method will be required as entries in the Disk File Information (DFI) tables for the ISAM routines.

An indexed-sequential file is built using the ISAM load routine, expanded using the ISAM add routine and processed by either the ISAM sequential or ISAM random routine. These routines are described in the 1130 Subroutine Library manual, Form C26-5929.

CONTENTS OF AN ISAM FILE

An ISAM file comprises the following: file label; file index; prime data area; overflow area.



The relative position of these components within the ISAM file is as follows

File Label | Index | Prime Data Area | Overflow Area

ISAM File Label. The first sector of any ISAM file contains the file label. This label contains information required by the ISAM routines for all future processing of the file. The file label is built by the ISAM load function, updated by ISAM add, and used by ISAM random and sequential. All label operations are performed automatically by the ISAM routines. The user need perform no label operation other than reserving one sector for the label when the file is initially defined.

The format of the ISAM label is shown in Figure 11.2.

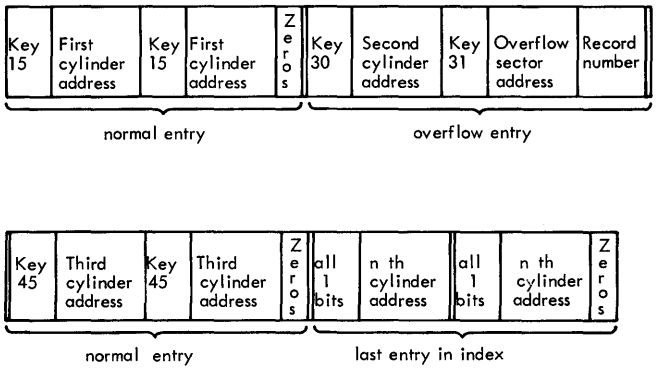
•Figure 11.2. Format of an ISAM Label

Word Number	Label Entry Description
1	Key length
2	Record length
3	Number of index entries per sector
4	Index entry length
5	Number of records per sector
6	Record number of last prime data record
7	Index entry number of last entry in file
8	Sector address of last prime data record
9	Sector address of last index entry
10	Sector address of next overflow record
11	Record number of next overflow record

ISAM File Index. The ability to read or write records anywhere in a file is provided by the file index. An entry in this index contains a cylinder address and the highest key that is associated with that cylinder. The ISAM routines locate a given record by searching the index for the key and then searching the specified cylinder for the desired record, again searching by key. To increase the efficiency of the ISAM routines, one sector of the index is retained in core storage for each file.

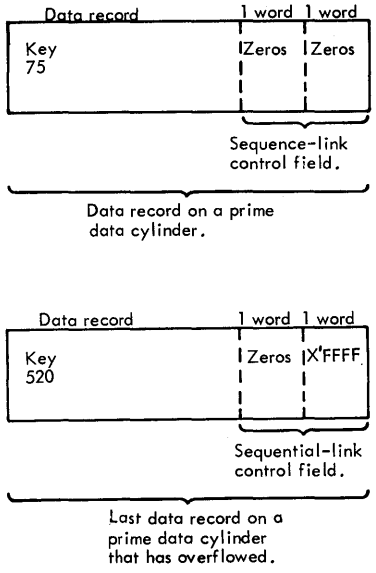
The key may be a part number or an employee name or any other identifying information that is contained in any record of the file. The key entries in the index are the numbers of the highest key on each cylinder in ascending collating sequence. The end of file record key is the key with the highest possible value, i.e., all bits are ones.

A portion of an index or index table is shown below. Note that each entry contains two sets of the same information. The second set is overlayed to show overflow data when the effected cylinder overflows.



Prime Data Area. This area contains the data records placed in the file by the ISAM load routine. The records must all be the same length (maximum 318 words). ISAM adds a two-word control field to each record. This control field, called the sequence-link control field, is used in the overflow area as a chaining indicator. It is used in the prime data area to indicate whether or not a cylinder has overflowed.

Prime data area records appear as follows.

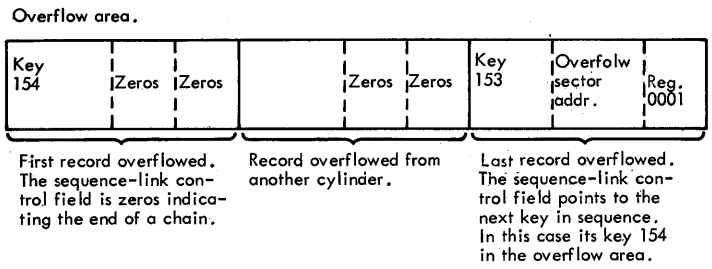


Overflow Area. When a new record is added to an indexed-sequential file it is placed according to key sequence. If records were to remain in precise physical order the insertion of each new record would require all records with higher keys to be shifted up. However, because ISAM files have an overflow area, a new record can be entered into its proper position on a cylinder and only cause records with higher keys on that cylinder to be shifted. The record that is forced off the end of the cylinder by the addition of the new record is written in the overflow area.

The index entry of any cylinder that has over flowed points to the overflow sector address and record number of the overflowed record in the overflow area. If two or more records in key order are added, the overflowed records are chained together in the overflow area through the entries in their sequence-link control field. The entry in the first record points to the second, the second to the third, etc. The last overflow record in the chain has a sequence-link control field of all zeros.

The number of cylinders to be allotted to the overflow area must be determined by the programmer when the file is initially defined. Records are placed in the overflow area in the order they have overflowed, not in key sequence.

To illustrate the overflow area, assume that on cylinder six of a defined file the last three entries have keys 150, 152 and 154. Key 154 would identify cylinder six in the index. Now we add a record with key 153, a record on another cylinder and a record with key 151. The overflow area would appear as shown below. Key 152 would identify cylinder six in the index. The overflow entry for cylinder six in the index would point to the overflow area.



DELETING DUPLICATE RECORDS CAUSED BY A DISK ERROR DURING AN ISAM ADD OPERATION

If a disk error occurs during an ISAM add (error code code/5004 in the accumulator) it may cause a record to be duplicated on the file. To check for a duplicate record, list the file or part of the file using ISAM sequential retrieve. If there is a duplicate record, one copy must be deleted.

To determine which record to delete, dump the file using a DUP DUMP and check the index entry for the affected cylinder. If the key of the duplicate record is less than or equal to the first key in the index entry, delete the second of the two records. If the key of the duplicate record is greater than the first key in the index entry, delete the first of the two records. In both cases, the remaining record is the one that will be processed by the ISAM random retrieve function.

Note that the duplicate record cannot be physically deleted. It is "Deleted" by performing a sequential read and flagging the copy that is no longer to be used.

The System Library is a group of disk-resident subprograms and mainline programs that perform I/O, conversion, arithmetic, and disk initialization and disk maintenance functions. A paper tape utility program (PTUTL) is also included in the System Library. Appendix F is a listing of the Monitor System Library.

ADDING AND REMOVING SUBROUTINES

Subroutines can be added to or deleted from the Monitor System Library as desired by the user. The DUP control record STORE is used to add a subroutine and the DUP control record DELETE is used to remove a program (see DUP Control Records). Each program in the IBM-supplied system deck is preceded by a DUP STORE control record.

The user should not remove subroutines that are called by other subroutines left in the System Library (refer to Appendix F for a list of subroutines called by other subroutines). Neither should he delete any of the mainline programs, since they may be required by Monitor programs.

SYSTEM LIBRARY SUBROUTINES

The 1130 Monitor System Library contains a group of programs that aid the programmer in making efficient use of the 1130 Computing System. Descriptions of the programs and methods for programming them are contained in the publication, IBM 1130 Subroutine Library (Form C26-5929). From an operational standpoint, the programs of particular interest are the ISSs, which manipulate the I/O devices attached to the 1130 Computing System and handle all programming details peculiar to each device. Table 5 lists the ISSs supplied with the 1130 Monitor system.

NOTE: User-written ISSs should be numbered from 20 down to avoid conflict with IBM-assigned ISS numbers (see Digit 1 under Preoperative Errors).

NOTE: Although the disk subroutines are technically not ISSs, they have most of the characteristics of an ISS.

The following paragraphs describe the use of some of the IBM-supplied ISS subroutines and discuss preoperative errors and I/O error restarts in which special handling is required. All addresses are given in symbolic form. See the table of equivalence in the listing of the Resident Monitor (Appendix H) to equate the symbolic to the absolute addresses. ISS preoperative error WAITs are listed in Appendix A.

PREOPERATIVE ERRORS

A preoperative error is an error condition detected before an I/O operation is started. It denotes either an illegal parameter, an illegal specification in the I/O area, or a device not-ready condition. This error causes a trap to $PRET and the following conditions:

- The Instruction Address Register displays the address $PRET+1.
- The Accumulator displays an error code represented by four hexadecimal digits.

  Digit 1 identifies the ISS called:

  1 - CARDx or PNCHx
  2 - TYPEx or WRTYx
  3 - PAPTx
  4 - READx
  5 - DISKx
  6 - PRNT1, PRNT2, or PRNTZ
  7 - PLOT1
  8 - SCATx
  9 - PRNT3 or PRNZ
  A - OMPR1

  Digits 2 and 3 are not used.
  Digit 4 identifies the error:

  0 - Device not ready
  1 - Illegal parameter or illegal specification in I/O area

Table 5. 1130 Disk Monitor System ISS Names

Device	Subroutine
1442 Card Read Punch	CARDZ, CARD0, or CARD1
2501 Card Reader	READZ, READ0, or READ1
1442 Card Punch	PNCHZ, PNCH0, or PNCH1
Disk	DISKZ, DISK1, or DISKN
1132 Printer	PRNTZ, PRNT1, or PRNT2
1403 Printer	PRNZ, or PRNT3
Keyboard/Console Printer	TYPEZ, or TYPE0
Console Printer	WRTYZ, or WRTY0
1134/1055 Paper Tape Reader Punch	PAPTZ, PAPT1, PAPTN, OR PAPTX
1627 Plotter	PLOT1
1231 Optical Mark Page Reader	OMPR1
Synchr. Comm. Adapter	SCAT1, SCAT2, or SCAT3

- $PRET contains the address of the call in question.

The ISS is set up to attempt initiation of the operation a second time if the CALL is re-executed. Pressing the PROGRAM START key will return control to the ISS for a re-execution of the call.

When a preoperative error is encountered the operator can:

- Correct the error condition if possible and press PROGRAM START, or
- Note the contents of the Accumulator and location $PRET, dump core storage, and proceed with the next job.

1442 CARD SUBROUTINE ERRORS (CARDx AND PNCHx)

Error Parameters

CARDZ, CARD0, PNCHZ, or PNCH0. There is no error parameter. If an error is detected during processing of an operation-complete interrupt, the subroutine traps to $PST4, with interrupt level 4 on. After the 1442 is made ready, pressing the PROGRAM START key will cause the operation to be reinitiated.

CARD1 or PNCH1. There is an error parameter. If an error is detected during processing of an operation-complete interrupt, the user program can elect to terminate (clear "subroutine busy indicator" and turn off the interrupt level) or to retry. A retry consists of waiting at $PST4 with interrupt level 4 on and then re-initiating the function.

Last Card. A read or feed function requested after the last card has been detected causes the last card to be ejected, and a trap to $PRET occurs. A punch function will punch and then eject the last card with a normal exit.

1442 Errors and Operator Procedures

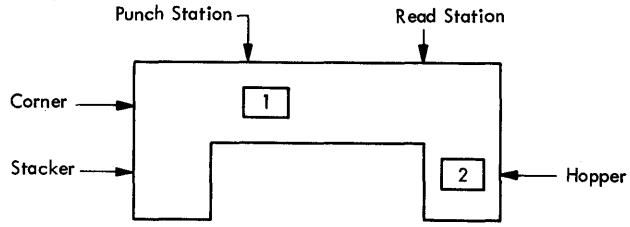
If a 1442 error occurs, the 1442 becomes not-ready until the operator has intervened. Unless the stop is caused by a stacker full (no indicator) or chip box indication, the 1442 card path must be cleared before proceeding. The 1442 error indicators and the position of the cards in the feed path should be used to determine which cards must be placed back in the hopper.

For the card subroutines, a retry consists of positioning the cards (i.e., skipping the first card in the hopper, if necessary, on a read or feed operation) and reinitiating the function whenever the card reader becomes ready.

Read errors do not apply to the 1442-5.

Hopper Misfeed. Indicates that card 2 failed to pass properly from the hopper to the read station during the card 1 feed cycle.

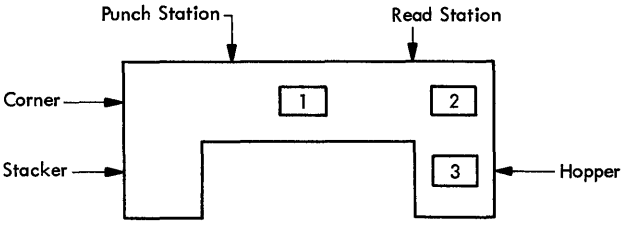
Card positions after error:



Error indicator: HOPR
Operator procedure: When program halts, press NPRO to eject card 1, place card 1 in hopper before card 2, and ready the 1442.

Feed Check (punch station). Indicates that card 1 is improperly positioned in the punch station at the completion of its feed cycle.

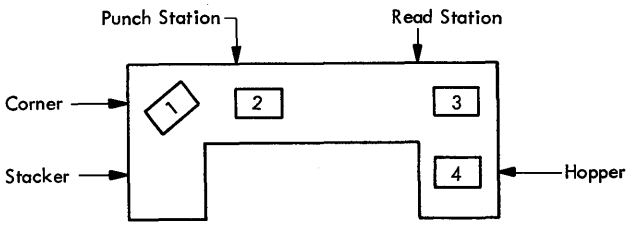
Card positions after error:



Error indicator: PUNCH STA
Operator procedure: When program halts, empty hopper, clear 1442 card path, place cards 1 and 2 in hopper before card 3 and ready the 1442.

Transport. Indicates that card 1 has jammed in the stacker during the feed cycle for card 2.

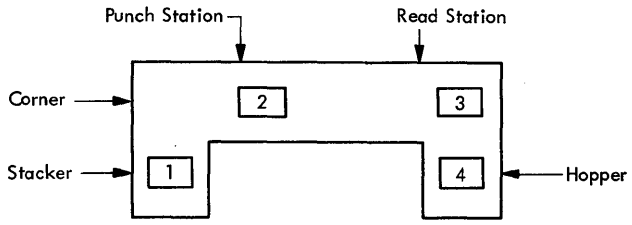
Card positions after error:



Error indicator: TRANS
Operator procedure: When program halts, empty hopper, clear 1442 card path, place cards 2 and 3 in hopper before card 4, and ready the 1442.

Feed Cycle. Indicates that the 1442 took an unrequested feed cycle and, therefore, cards 1, 2, and 3 are each one station farther ahead in the 1442 card path than they should be.

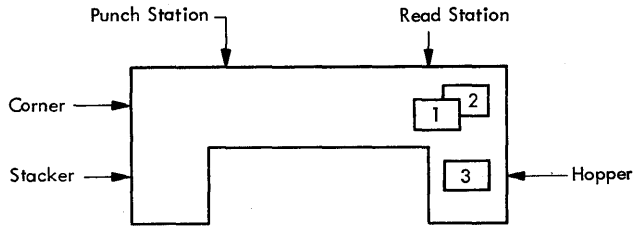
Card positions after error:



Error indicator: FEED CLU
Operator procedure: When program halts, empty hopper, press NPRO to eject cards 2 and 3, place cards 1, 2, and 3 in hopper before card 4, and ready the 1442.

Feed Check (read station). Indicates that card 1 failed to eject from the read station during its feed cycle.

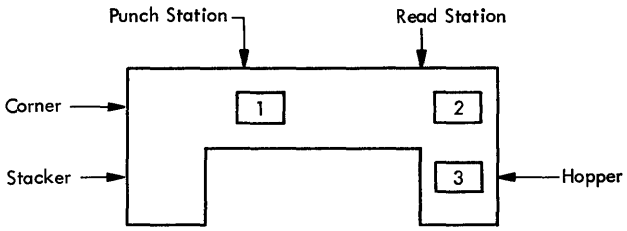
Card positions after error:



Error indicator: READ STA
Operator procedure: When program halts, empty hopper, clear 1442 card path, place cards 1 and 2 in hopper before card 3, and ready the 1442.

Read Registration. Indicates incorrect card registration or a difference between the first and second reading of a column.

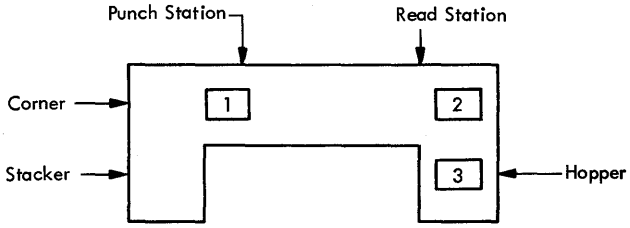
Card positions after error:



Error indicator: READ REG
Operator procedure: See Feed check (punch station). Repeated failures of this type might indicate a machine malfunction.

Punch Check. Indicates an error in output punching.

Card positions after error:



Error indicator: PUNCH
Operator procedure: When program halts, empty hopper, check card position and press NPRO to clear 1442 card path. If necessary, correct card 1 to prepunched state. Place (corrected) card 1 and card 2 in hopper before card 3 and ready the 1442.

2501 CARD SUBROUTINE ERRORS (READx)

Error Parameters

READZ or READ0. There is no error parameter. If an error is detected during processing of an operation-complete interrupt, the subroutine traps to $PST4, with interrupt level 4 on. After the 2501 is made ready, pressing the PROGRAM START key will cause the operation to be reinitiated.

READ1. There is an error parameter. If an error is detected during processing of an operation-complete

interrupt, the user program can elect to terminate (clear "subroutine busy indicator" and turn off the interrupt level) or to retry. A retry consists of waiting at $PST4 with interrupt level 4 on until the 2501 becomes ready, and then reinitiating the function.

Last Card. A read function requested after the last card has been detected causes a trap to $PRET.

2501 Errors and Operator Procedures

If a 2501 error occurs, the 2501 becomes not-ready until the operator has intervened. Unless the stop is caused by a stacker full or cover open (ATTENTION), the 2501 card path must be cleared before proceeding. The 2501 error indicators and the position of the cards in the feed path should be used to determine which cards must be placed back in the hopper.

For the card subroutines, a retry consists of positioning the cards (i.e., skipping the first card in the hopper, if necessary) and reinitiating the read function whenever the card reader becomes ready.

FEED CHECK. A feed check indicates that a card is mispositioned in the feed path or a card has failed to feed from the hopper.

When the program traps to $PST4, empty the hopper and clear the 2501 card path. If a card is improperly positioned at the pre-read station (it has not been read), place this card ahead of the cards remaining to be read, place the deck back in the hopper, and ready the 2501.

READ CHECK. A read check indicates incorrect card registration or a difference between the first and second reading of a column.

When the program traps to $PST4, empty the hopper, NPRO, place the last two cards in the stacker ahead of the deck remaining to be read, place this deck back in the hopper, and ready the reader.

CONSOLE PRINTER SUBROUTINE ERRORS (TYPEZ, TYPE0, WRTYZ, and WRTY0)

If the carrier attempts to print beyond the manually positioned margins, a carrier restore (independent of the program) occurs.

Subroutine printing begins wherever the carrier is positioned as a result of the previous print operation. There is no automatic carrier return as a result of a call to the subroutine.

If the Console Printer indicates a not-ready condition after printing has begun, the subroutine traps to $PST4 with interrupt level 4 on. After the Console Printer is made ready, pressing the PROGRAM START key will cause the operation to be reinitiated.

KEYBOARD SUBROUTINE FUNCTIONS (TYPEZ and TYPE0)

Re-entry

When the ERASE FIELD key is pressed, a character interrupt signals the interrupt response subroutine that the previously-entered Keyboard message is in error and will be re-entered. The subroutine prints two slashes on the Console Printer, restores the carrier to a new line, and prepares to replace the old message in the I/O area with the new message. The operator then enters the new message. The old message in the I/O area is not cleared. The new message overlays the previous message, character by character. If the previous message was longer than the new message, characters from the previous message remain (followin the NL character which terminated the new message).

When the interrupt response subroutine recognizes the end-of-message control character, it assumes the message has been completed, stores an NL character in the I/O area, and terminates the operation.

TYPEZ does not print two slashes when ERASE FIELD key is pressed.

Backspace

When the backspace key is pressed, the last graphic character entered is slashed and the address of the next character to be read is decremented by +1. If the backspace key is pressed twice consecutively, the character address is decremented by +2, but only the last graphic character is slashed. For example, assume that ABCDE has been entered and the backspace key pressed three times. The next graphic character replaces the C but only the E is slashed. If the character F had been used for replacement, the paper would show ABCDE/FF but ABFFF would be stored in the buffer.

TYPEZ treats the backspace key as if it were the erase field key.

PAPER TAPE SUBROUTINES (PAPTx)

If the reader or punch becomes not ready during an I/O operation, the subroutines exit to the user via the error parameter. The user can request the subroutine to terminate (clear device busy on the interrupt level) or to wait at $PST4 (postoperative error trap) waiting for operator intervention (interrupt level 4 on).

If the 1134/1055 indicates a not-ready condition after an operation has been initiated, the subroutines trap to $PST4 with interrupt level 4 on. After the device has been made ready, pressing the PROGRAM START key will cause the operation to be reinitiated.

SYSTEM LIBRARY MAINLINE PROGRAMS

The 1130 System Library mainline programs provide the user with the ability to perform disk maintenance and paper tape utility functions by requesting execution of the appropriate program directly through the job stream.

DISK MAINTENANCE PROGRAMS

The disk maintenance programs are mainline programs that are a part of the System Library and that initialize and modify disk cartridge IDs, addresses, and tables required by the Monitor system. Normally, they should never be deleted from the System Library.

The disk maintenance mainline programs are:

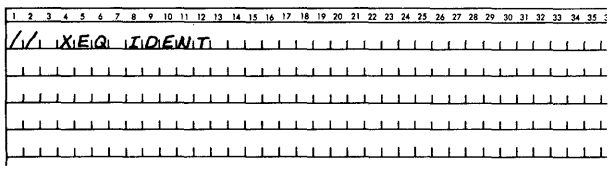
IDENT	-	Print Cartridge ID
DISC	-	Satellite Disk Initialization*
DSLET	-	Dump System Location Equivalence Table
ID	-	Change Cartridge ID
COPY	-	Disk Copy
ADRWS	-	Write Sector Addresses in Working Storage
DLCIB	-	Delete CIB
MODIF	-	Monitor System Update

The disk maintenance programs (except ADRWS) are called by an XEQ monitor control record. Some disk maintenance programs also require an ID control record. The format and use of the ID control record is described under the program descriptions which follow.

IDENT (Print Cartridge ID)

This program prints the ID and physical drive number of each cartridge mounted on the system. The program overrides any cartridge IDs specified on the JOB card and operates with all ready drives. IDENT will read and print illegal IDs including negative numbers.

The calling sequence for IDENT is:



*All new cartridges must be initialized with DCIP before any operation is performed under Monitor control. DCIP also provides a disk copy function similar to the COPY program in the System Library.

Printout

PHYSICAL DRIVE	CART. ID
00	XXXX
01	XXXX
02	XXXX
03	XXXX
04	XXXX

where

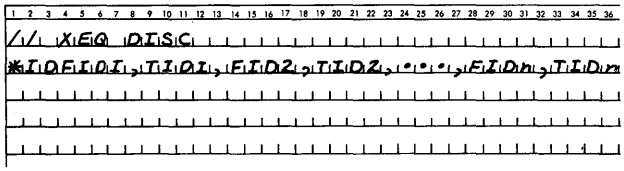
XXXX is the cardridge ID. Only the IDs on ready drives are printed.

DISC (Satellite Disk Initialization)

This program re-initializes up to four satellite cartridges -- all but the master cartridge (see DCIP). DISC gives the user the ability to re-initialize a disk cartridge on line. It writes the sector addresses, defective cylinder addresses, a cartridge ID, a LET, a DCOM, an error message program, and a CIB on each cartridge initialized.

DISC overrides all cartridge IDs specified on the JOB card except the master cartridge ID.

The calling sequence for DISC is:



where

FID1 through FIDn are the IDs currently on the satellite cartridges to be re-initialized (identified by IDENT or a JOB record).

TID1 through TIDn are the IDs to be written on the satellite cartridges by this program. A valid cartridge ID is a number between /0001 and /7FFF.

DISC Operation

DISC causes all selected satellite drives to seek home. The program then writes sector addresses and three distinct bit patterns (/AAAA, /5555, and /0000) on all sectors on the first cylinder of the first disk cartridge being reinitialized. The program reads back each pattern after it is written. If no error occurs on any of the patterns, DISC continues to the next cylinder. This procedure is repeated until all 203 cylinders have been checked. The program then starts reinitialization on the next cartridge selected. If a read error occurs, the cylinder on which the error occurred is rewritten

and reread 50 times using the same pattern that caused the error to appear. If a second error occurs, the first sector address of the cylinder is placed in the defective cylinder table in word 1 of sector @IDAD. If a second and third defective cylinder are found, their cylinder addresses are written in words 2 and 3 of sector @IDAD. If there are no defective cylinders on the cartridge, words 1, 2, and 3 contain /0658. The cartridge ID is written in word 4 and the copy ID is written in word 5 of sector @IDAD. An error message (NON-SYST. CART. ERROR) and the program to print it on the Console Printer is also stored in sector @IDAD. The error message is printed if a cold start is attempted on this non-system cartridge.

Printout

When DISC is executed, the user punched *ID record is printed on the principal print device. Following this printout one or more of the following error messages may be printed.

CARTRIDGE XXXX INVALID...LOG0 CARD ID

The ID of the master cartridge (logical drive 0) has been specified as a current ID on the *ID card.

CARTRIDGE XXXX NEW LABEL IS INVALID

A new ID is outside of the range /0001 - /7FFF.

CARTRIDGE XXXX IS NOT AVAILABLE

A selected cartridge is not on the system

CARTRIDGE XXXX IS DEFECTIVE

Sector @IDAD or more than 3 cylinders are defective on a satellite cartridge being reinitialized (to identify the defective cylinders, initialize the cartridge using DCIP).

Following the reinitialization of the selected cartridges, the following message is printed.

XXXXYYYY NOT DONE

or

COMPLETE

XXXX is the old (FID1) cartridge ID

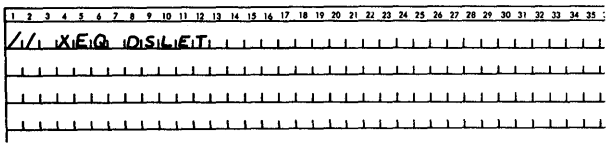
YYYY is the new (TID1) cartridge ID

One line is printed for each satellite cartridge that is reinitialized. A NOT DONE message should appear only if an error message has previously been printed.

DSLET (Dump System Location Equivalence Table)

This program dumps the contents of SLET to the principal print device. Each entry printed consists of a symbolic name, phase ID, core address, word count, and disk sector address. A SLET dump is shown in Appendix I.

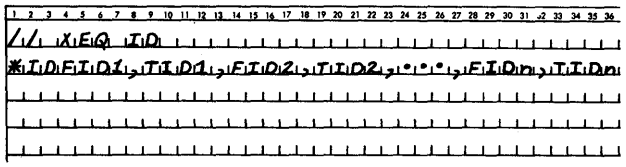
The calling sequence for DSLET is:



ID (Change Cartridge ID)

This program changes the ID on up to four satellite cartridges.

The calling sequence for ID is:



where

FID1 through FIDn are the IDs currently on the satellite cartridges being changed (these IDs must be in the same logical order as the entries on the JOB card),

TID1 through TIDn are the new IDs to be written on the selected satellite cartridges. A valid cartridge ID is a number between /0001 and /7FFF.

Printout

FFFF TTTT NOT DONE

or

COMPLETE

where

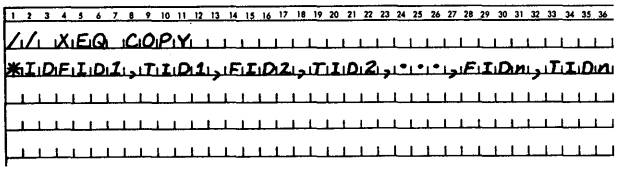
FFFF is the FROM ID
TTTT is the TO ID
NOT DONE is printed if a selected cartridge is not found on the system.

One line is printed for each cartridge ID that is changed (maximum 4).

COPY (Disk Copy)

This program copies the contents (except the defective cylinder table and the cartridge ID) of one cartridge onto another. The copy ID (word 5 of sector @IDAD) is incremented by one on the destination cartridge. The cartridge to be copied onto must have previously been initialized (see DISC or DCIP).

The calling sequence for COPY is:



where

FID1 through FIDn are the IDs of the cartridges to be copied,

TID1 through TIDn are the IDs of the cartridges onto which the copies are to be made.

If multiple copies are to be made from a single master, FID1 through FIDn will all contain the same ID.

If a system cartridge from a system with a different configuration is copied, it will be necessary to reconfigure the cartridge before a Cold Start can be performed (see System Reload).

Printout

FFFF TTTT NOT DONE

or

COMPLETE

or

NOT PRES

or

NO. ERROR

where

FFFF is the FROM ID
TTTT is the TO ID
NOT PRES indicates that the ID requested was not found.
NO. ERROR indicates that the ID requested exceeded /7FFF.

One line is printed for each copy requested on the *ID record. The printout occurs at the end of the job.

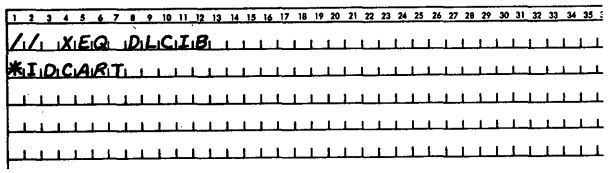
ADRWS (Write Sector Addresses in Working Storage)

This program, linked to from DUP on detection of the DUP control record, DWADR, writes the sector address on each sector of Working Storage of the disk cartridge specified in the DWADR control record (see *DWADR in DUP Control Records). ADRWS is intended for system use only.

DLCIB (Delete Core Image Buffer)

This program deletes the CIB from a non-system cartridge. If a User Area is defined, it is moved two cylinders closer to cylinder 0. The new addresses of disk areas moved as the result of the deletion of the CIB are reflected in DCOM on the master cartridge, on the non-system cartridge from which the CIB is deleted, and in COMMA.

The calling sequence for DLCIB is:



where

CART is the ID of the non-system cartridge from which the CIB is to be deleted.

Printout

CART	UA/FX	FPAD
XXXX	YYYY	NNNN

or

XXXX ERROR

where

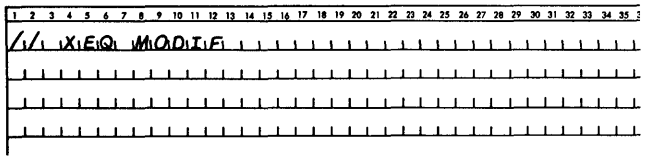
XXXX is the cartridge ID
YYYY is the User Area sector address
NNNN is the File Protect Address
ERROR is printed if the CIB was not deleted (cartridge not found on system or cartridge not specified on JOB card)

MODIF--SYSTEM MAINTENANCE PROGRAM

Included in the System Library is a system maintenance program, MODIF, that provides the user with the ability to update the Monitor system on the master cartridge. This program makes changes to the version and modification level word in DCOM, and can be used to update both System Programs and/or the System Library. A card deck or paper tape containing corrections to update the Monitor system to the latest version and modification level is supplied by IBM. Every modification must be run to update the version and modification level, even if the affected program has been deleted from the system.

NOTE: The replacement of a system program phase that contains reload entries (references to SLET generated by the System Loader during an initial or reload operation) cannot be performed by MODIF. MODIF does not update the System Reload Table. The replacement phase must be loaded by a system reload.

The calling sequence for MODIF is:



System Program Maintenance

Typical input for System Program update is shown in Figure 12.

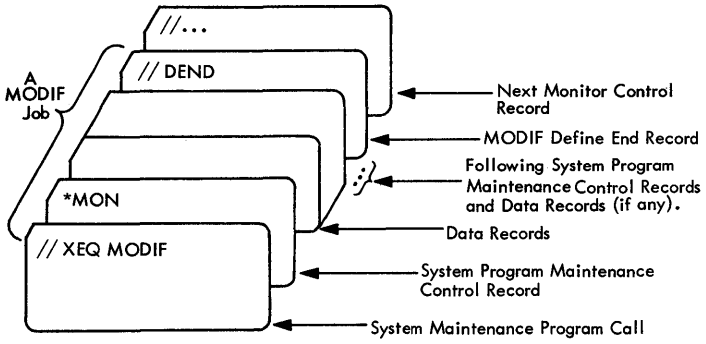


Figure 12. Layout of an Input Deck for a System Program Update

System Program Maintenance (Patch) Control Record

Each Monitor program phase to be changed requires a patch control record. If MODIF determines in analyzing SLET that the FORTRAN Compiler or the Assembler has been voided from the disk, modifications to these programs are not made; however, the version and modification levels for these programs are updated in DCOM.

The format of the patch control record is as follows.

Card Column	Contents	Notes
1–5	*MON	These characters identify a system patch to the FORTRAN Compiler, RPG Compiler, Assembler, DUP, Supervisor, Core Load Builder, System Device Subroutines[1], or Core Image Loader
5	blank	
6–8	vmm	The version (v) and modification level (mm) are specified in hexadecimal.
9	0 or G or R	0 = System Modification Update G = General temporary fix[4] R = Restricted temporary fix[5]
10	blank	
11–14	xxxx	The SLET ID of the Monitor program phase to which the patch is to be made is specified in hexadecimal. 0000 indicates an absolute patch (see columns 28–31, 33–36).
15	blank	
16–19	nnnn	"nnnn" specifies (in hex) the number of patch data records following this patch control record.
20	blank	
21	B or H	This character identifies the format of the patch data records (binary system format or hex patch format)
22	blank	
23–26	pppp	"pppp" specifies (in hex) the total number of patch control records to be processed. This parameter is required on the first patch control record only.[2]
27	blank	
28–31	dsss	The drive code (d) and sector address (sss) of the program to be patched are specified in hexadecimal. This field is used only when the SLET ID (columns 11–14) is 0.
32	blank	
33–36	cccc	"cccc" specifies (in hex) the core address of the first word of this sector. This field is used only when the SLET ID (columns 11–14) is 0.[3]
37–80	Not used	

Notes:

1. Modifications to subroutines in the System Device Subroutine Area must be made with a *MON patch, not a *SUB DELETE and STORE.
2. A MODIF job may perform both System Program and System Library maintenance (see System Library Maintenance). In such a case the number in columns 23–26 must include the *SUB card in the count. Only one subroutine control record is allowed in any MODIF job, and it must be the last MODIF control record (not counting // DEND) in the stacked input.
3. Core addresses can be obtained from the microfiche listing.
4. This fix can only be installed on a system with the same level or one higher than that indicated in *MON or *SUB card. It will not change the level of the system.
5. This fix can only be installed on a system with the same level as indicated in a *MON or *SUB card.

Patch Data Record Formats

Patch Data Records may not contain CALLs or LIBFs, nor will the relocation indicators be used.

Binary System Format.

Word	Contents
1	Location
2	Checksum
3	Type Code (first 8 bits) 00001010
4-9	Relocation Indicators
10-54	Data words 1 through 45
55-60	ID and sequence number or may be blank

Hex Patch Format.

Card Column	Contents	Notes
1-4	aaaa	"aaaa" specifies (in hex) the core address (origin) of the patch. Each patch record must have a core address.
5	blank	
6-9, 11-14, 16-19, etc.		Each 4-column field contains one word of patch data (in hex). Up to 13 words of patch data can be specified per record. A blank column follows each word.
66-68, 73-80	Not used	

Hex patch cards may contain ID/sequence numbers. Zeros must be punched as leading blanks will not be assumed.

System Library Maintenance

Changes to the System Library require the deletion of the old program and the storing of the new one. MODIF updates the version and modification level word; the actual operation is performed by a DUP DELETE operation, followed by a DUP STORE operation.

Typical input for System Library maintenance is shown in Figure 13.

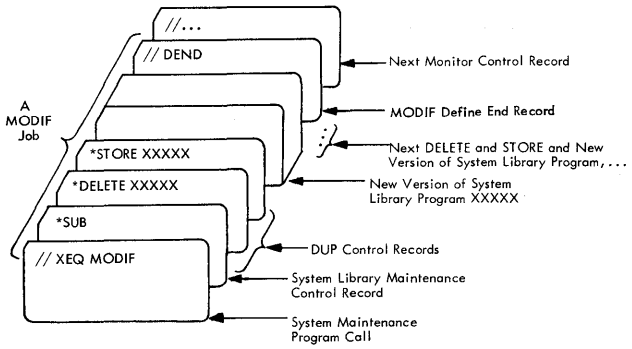


Figure 13. Layout of an Input Deck for a System Library Update

System Library Maintenance (Subroutine) Control Record

Only one subroutine control record may appear in a MODIF job; however, any number of DELETEs and STOREs may be performed with that control record. System Library maintenance may also be performed concurrently with System Program maintenance (see note 2, columns 23-26 of *MON card).

The format of a subroutine control record is as follows

Card Column	Contents	Notes
1-4	*SUB	These characters identify a system patch to the System Library.
5	blank	
6-8	vmm	The version (v) and modification level (mm) are specified in hexadecimal.
9	0	Reserved
10-15	blank	
16-19	nnnn	"nnnn" specifies (in hex) the number of deletes and stores to be processed.
20-80	Not used	

All Maintenance

Define End Record

All MODIF jobs must end with a card punched as follows.

Card Column	Contents	Notes
1-7	//bDEND	
8-80	Not used	

This card terminates MODIF execution and passes control to the Supervisor.

Operating Procedures

The card deck or paper tape supplied by IBM is to be run as a Monitor job.

When a modification is completed successfully, the following messages are printed on the principal printer.

MODIF EXECUTION 0WXX
MODIF TERMINATION 0YZZ

where

WXX is the old version and modification number, and YZZ is the new version and modification number.

MODIF Error Messages

If an error occurs during MODIF execution, an error message is printed on the principal printer. The format of the error message is as follows.

ERROR# XXXX XXXX

Following the printing of the error message, the system will WAIT. All MODIF errors and their recovery procedures are listed in table 6.

Table 6. MODIF Error Messages.

Error Number	Description	Recovery Options*	First Hex Number Printed	Second Hex Number Printed
1	Invalid patch control record (*MON or *SUB)	A. Correct error and reread from corrected patch control record. (If the error has occurred on the first patch control record, restart the modification.) B. Terminate modification, CALL EXIT.		
2	Checksum error on binary patch data record.	A. Rechecksum and reread from preceding patch control record. (If the error has occurred on the first patch control record, restart the modification.) B. Terminate modification, CALL EXIT. C. Reread card in error (cards may be out of order).	Amount of checksum difference.	Number of binary records read after patch header (including record in error).
3	Invalid hex data record.	A. Correct error and reread from preceding patch control record. B. Terminate modification, CALL EXIT. C. Reread card in error.		
4	Modification level error in system modification update.	A. Correct error and reread from corrected patch control record. B. Terminate modification, CALL EXIT.	Present version and modification level (from DCOM on disk).	Level of version and modification (from patch control record).
5	New modification level lower than current level in system modification update.	A. Correct error and reread from corrected patch control record. B. Terminate modification, CALL EXIT. C. Reduce level and continue.	Present version and modification level (from DCOM on disk).	Level of version and modification (from patch control record).
6	Monitor control record or // DEND card read before required number of patches read.	A. Read new patch control record. B. Terminate modification, CALL EXIT.	Number of patches not installed.	
7	DCOM configuration indicators do not agree with SLET or, Required system I/O routine missing.	A. Restart MODIF execution. B. Terminate modification, CALL EXIT.	Contents of Accumulator when error was detected.	Address +2 from which error branch was executed.
8	DUP control record errors (DELETES or STORES).	Print error indicators and WAIT. Press START to continue.	XXYY where XX is the number of DUP errors detected (see DUP error printout) and YY is the number of DUP control records not processed.	Number of DUP control records specified on *SUB patch control record.
9	SLET ID not found.	Print error indicators and WAIT. Press START to read patch data cards without processing and read new control record.	SLET ID in question.	
A	Patch exceeds space allotted on disk for this phase.	Print error indicators and WAIT. Press start to read patch data cards without processing and read new control record.	High core patch address.	High core SLET address.
B	// DEND card not found (patches completed but version and modification level in DCOM not updated.	Press START to CALL EXIT or, Rerun modification with // DEND card.		
C	Modification level error in General temporary fix (see note 4, page 64).	This error terminates execution. Press Program Start Key to CALL EXIT. Any preceding patches in this MODIF JOB have been installed.	Present version and modification level (from DCOM on disk).	Version and modification level (from patch control record).
D	Modification level error in Restricted temporary fix (see note 5, page 64).	This error terminates execution. Press Program Start Key to CALL EXIT. Any preceding patches in this MODIF JOB have been installed.	Present version and modification level (from DCOM on disk).	Version and modification level (from patch control record).
E	System modification update mixed with temporary fixes.	This error terminates execution. Press Program Start Key to CALL EXIT. Any preceding patches in this MODIF JOB have been installed.		

*Set console entry switches as desired for errors 1-7 and press START.
No switches on - recovery A
Switch 0 on - recovery B
Switch 15 on - recovery C

MODIF Example

The purpose of the following example is to change one instruction in the 1134/1055 System Subroutine. The SLET ID of this subroutine is /0091.

Address (from assembly listing)	Change From	Change To
/0023	/0000	/7002

If the new modification level is 8, the following control and data cards must be punched by the user.

System Program Maintenance Control Record

```
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 31 32 33 34 35 36 37
*MON 2080 0091 0011 H 0001
```

Patch Data Record: Hex Patch Format

```
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 31 32 33 34 35 36 37
0023 7002
```

These two cards together with the Monitor control cards shown in Figure 12 will perform the required modification.

At completion of execution, the following messages will print on the principal printer.

MODIF EXECUTION	0207	The execution of MODIF has been initialized on version 2 level 7
MODIF TERMINATION	0208	The patch has been installed and the new level is 8.

PAPER TAPE UTILITY (PTUTL)

This program accepts input from the Keyboard or the 1134 Paper Tape Reader and provides output on the Console Printer and/or the 1055 Paper Tape Punch.

PTUTL allows changes and/or additions to FORTRAN and Assembler language source records as well as Monitor control records.

The calling sequence for PTUTL is:

```
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 31 32 33 34 35 3
// XEQ PTUTL
```

Operating Procedure

If paper tape is the principal input, select the appropriate initializing procedure listed below and continue.

If the Resident Monitor is in core:

- Place the PTUTL execute tape in the paper tape reader.
- Press PROGRAM START.

If the Resident Monitor is not in core:

- Place the cold start paper tape record in the paper tape reader.
- Press IMM STOP, RESET, and PROGRAM LOAD on the console.
- Place the PTUTL execute tape in the paper tape reader.
- Press PROGRAM START.

The paper tape utility program is loaded into core and then comes to a WAIT with /1111 displayed in the Accumulator. This WAIT allows the operator to ready the Console Printer, paper tape reader, and paper tape punch. The user should punch a leader of delete codes on the paper tape punch.

At this time, the user can select the desired program options by turning on the appropriate console entry switches. Figure 14 shows the PTUTL console entry switch logic in flowchart form.

Console Entry Switch On	Option
0	Print record after reading
1	Read paper tape records from 1134
2	Accept Keyboard input[1]
3	Punch paper tape records on 1055
14	WAIT after punching with /3333 in the Accumulator[3]
15	WAIT after printing with /2222 in the Accumulator[2]
All switches off	CALL EXIT[3]

NOTES:

1. The keyboard input option uses TYPE0; therefore all features of that subroutine apply to PTUTL.
   - The input record cannot exceed 80 characters.
   - Pressing the backspace key cancels the last character entered.
   - Pressing the ERASE FIELD key cancels the entire record and allows the user to restart.
   - Pressing the EOF key indicates that the record is complete. The Keyboard is released and the program continues.
2. Keyboard input will replace the last paper tape record read if console entry switch 2 is turned on prior to pressing PROGRAM START.
3. The test for exit is made just before an input record is read; therefore, a convenient way to branch out of PTUTL is to perform a WAIT after punching the last record desired (console entry switch 14 on). Turn off all console entry switches and press PROGRAM START. A CALL EXIT will be executed.

Paper Tape Not-Ready WAITs

Condition	Indication	Recovery Procedures
Paper tape reader not ready	Program WAITs with /3005 in the Accumulator	Ready reader if additional tape is to be read. Set the console entry switches as desired and press PROGRAM START.
Paper tape punch not ready	Program WAITs with /3004 in the Accumulator	Ready the paper tape punch and press PROGRAM START. To repunch the record that was being processed when the not-ready occurred, set console entry switches 1 and 2 off (to prevent another record from being read), set switches 3 and 14 on (punch a record and WAIT with /3333 in the Accumulator), and press PROGRAM START. After the record is punched, return the console entry switches to the original configuration and press PROGRAM START.

Example

Assume that the following records appear on a tape.

// JOB
// *(comments)
// ASM
// DUP
ASM Control Records
Source Program

The user now desires to alter the comments record, insert a // PAUS record after the comments record, and delete the // DUP record. The procedure is as follows.

1. Load and execute PTUTL. The program will WAIT with /1111 in the Accumulator.
2. Load the source tape in the paper tape reader and ready the paper tape punch and Console Printer. Make a leader of delete codes on the punch.
3. Turn on console entry switches 1, 3, and 14.
4. Press PROGRAM START.
5. The // JOB record will be read, reproduced, and the program will WAIT with /3333 in the Accumulator.
6. Turn on console entry switches 0, 1, 2, 3, 14, and 15.
7. Press PROGRAM START.
8. The comments record in the source tape will be read and printed on the Console Printer. The program will WAIT with /2222 in the Accumulator.
9. Press PROGRAM START. The Keyboard will be selected (PROCEED light on) and the program will WAIT with /3333 in the Accumulator.
10. Enter the new comments record in the proper format.
11. Press the EOF key on the Keyboard.
12. The new comments record will be punched on the tape, replacing the old record. The program will WAIT.
13. Turn off console entry switch 1. Press PROGRAM START. The Keyboard will be reselected.
14. Enter the // PAUS record from the Keyboard and press EOF.
15. Turn off the console entry switches 0, 2, and 15. Turn on switch 1. Leave switches 3 and 14 on.
16. Press PROGRAM START.
17. The // ASM record will be read and reproduced on the punch. The program will WAIT with /3333 in the Accumulator.
18. The next record // DUP, is to be deleted; therefore, switches 0, 1, and 15 should be set on, all other console entry switches should be set off.
19. Press PROGRAM START.
20. The // DUP record will be read and printed but not punched. The program will WAIT with /2222 in the Accumulator.
21. Leave the sense switches at the present setting and press PROGRAM START. The next record on the input tape will be read into the I/O buffer, overlaying the // DUP record.
22. Turn on console entry switches 1 and 3, all others off.
23. Press PROGRAM START.
24. The remainder of the source tape will be read in and reproduced, record for record.
25. When the paper tape reader goes not-ready at the end of the source tape, the program will again WAIT with /3005 in the Accumulator. Set all console entry switches off and press PROGRAM START. A CALL EXIT will be executed.

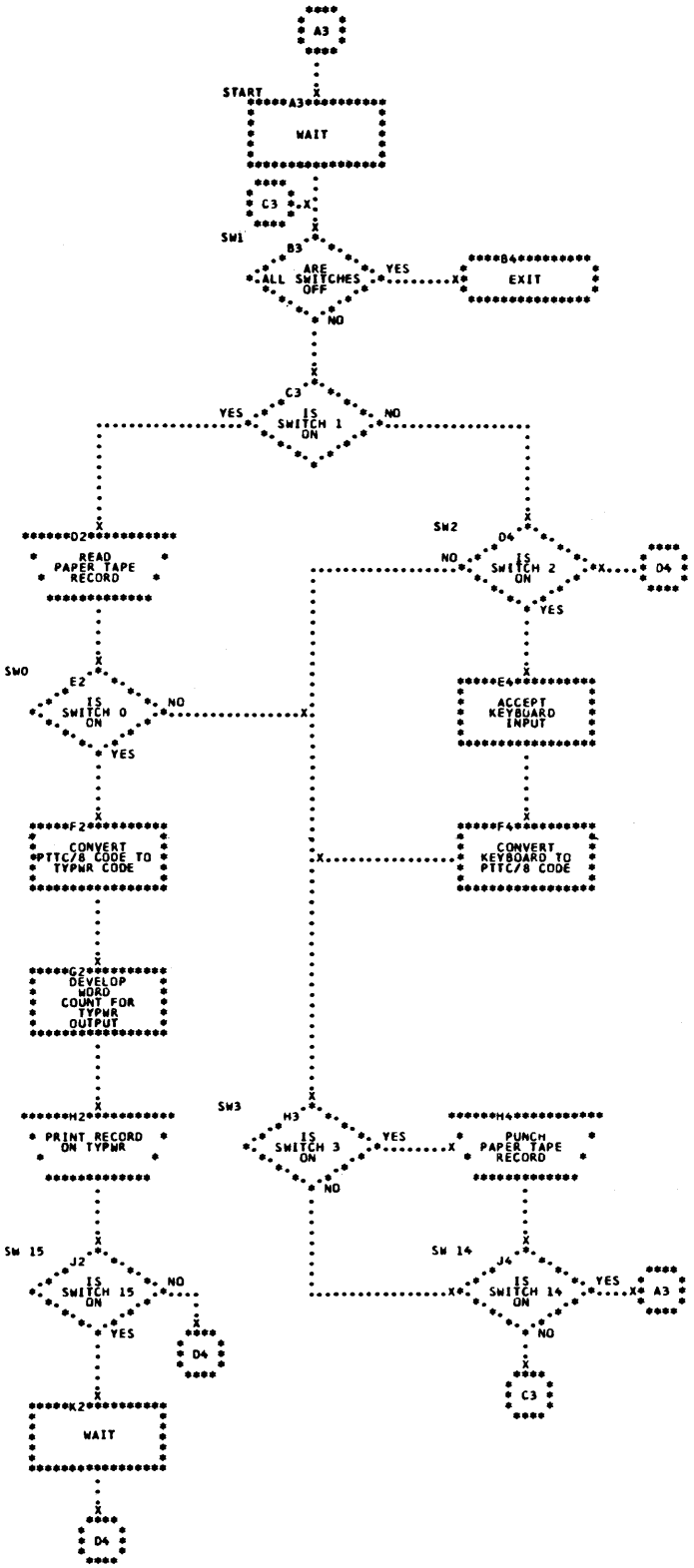


Figure 14. PTUTL Console Entry Switch Options

SYSTEM LIBRARY UTILITY SUBROUTINES

Also included in the System Library are a group of subroutines that perform utility functions for the Monitor system.

The utility subroutines are:

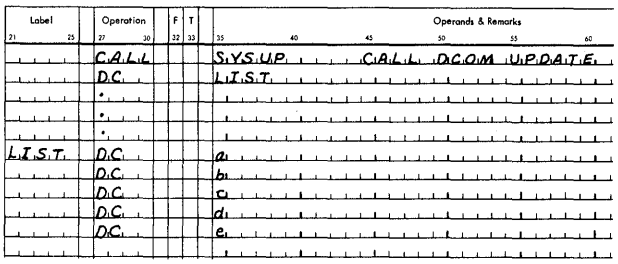
SYSUP	-	DCOM Updating Subroutine
RDREC	-	Read *ID Record Subroutine
CALPR	-	Call System Print Subroutines
FSLEN	-	Fetch Phase ID (FSLEN) or Fetch System Subroutine (FSYSU)
FLIPR	-	LOCAL/SOCAL Overlay Subroutine

SYSUP can be called by the user. The other utility subroutines are for system use only.

SYSUP (DCOM Update)

Whenever a core load requires changing disk cartridges during the job, SYSUP must be called to update DCOM on the master cartridge (logical drive 0) with the IDs and DCOM information from all satellite cartridges mounted on the system that are specified in the list or array in the calling sequence.

The Assembler language calling sequence for SYSUP is:

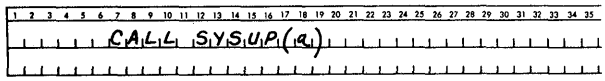


where

a is the ID of the master cartridge on the system,
b is the ID of the first satellite cartridge on the system,
c is the ID of the second satellite cartridge on the system,
d is the ID of the third satellite cartridge on the system,
e is the ID of the fourth satellite on the system.

a may be zero, in which case the master cartridge is the same as that defined for the previous job.

The FORTRAN calling sequence for SYSUP is:



where

a is the name of the last item in an array containing the IDs of the satellite cartridges on the system.

The last item in the array may be zero, in which case the master cartridge is the same as that defined for the previous job. For example:

CALL SYSUP (A(5))

The array is stored in reverse order

A(5) DC
A(4) DC
A(3) DC
A(2) DC
A(1) DC

Thus A(5) is the entry for logical 0, the master cartridge. A is a one-word integer.

NOTE: The list or array must be no longer than five words. It may be shorter. If a list or array shorter than five words is specified, the Assembler array must be terminated with an ID of all zeros (all zeros in the first entry will not terminate the array).

The FORTRAN array must be started with an ID of all zeros (all zeros in the last entry will not terminate the array). For example, a three-cartridge FORTRAN array would be specified as (A(4)) with A(1) having a DC of all zeros.

Printout

The following error messages may be printed by SYSUP (SYSUP error messages are also listed with the Supervisor Errors in Appendix A).

XXXX IS NOT AN AVAILABLE CARTRIDGE ID

A specified cartridge is not in the system.

XXXX IS A DUPLICATED SPECIFIED CARTRIDGE ID

The same ID appears more than once in the list or array in the calling sequence.

XXXX IS A DUPLICATED AVAILABLE CARTRIDGE ID

Two or more disks (specified in the calling sequence) have the same cartridge ID.

An error printout is followed by termination of execution.

CALPR (Call System Print Subroutine)

This subroutine calls FSLEN to bring the system print subroutines into core storage for the purpose of printing one or more lines on the principal printer. This subroutine is intended for system use only.

RDREC (Read *ID Record)

This subroutine is called by the Disk Maintenance Programs to read the *ID record. The *ID record is printed on the principal print device. This subroutine is intended for system use only.

FSLEN (Fetch Phase IDs and Fetch System Subroutine)

This subroutine has two entry points: FSLEN and FSYSU.

- FSLEN -- Fetch Phase IDs from SLET

  This entry point obtains the requested phase ID headers from SLET.

- FSYSU -- Fetch System Subroutines

  Fetches the requested system subroutine into core storage. This subroutine is intended for system use only.

FLIPR (LOCAL/SOCAL Overlay)

The Monitor system library contains a flipper subroutine (FLIPR), which is used to call LOCAL (load-on-call) and SOCAL (system-load-on-call) subroutines into core storage. FLIPR is used with DISKZ, DISK1, or DISKN.

FLIPR passes the total word count to DISKZ, DISK1, or DISKN to fetch the LOCAL. When a LOCAL subroutine is called, control is passed to the flipper, which reads the LOCAL into core storage if it is not already in core and transfers control to it. All LOCALs in a given core load are executed from the same core storage locations; each LOCAL overlays the previous one. FLIPR fetches SOCALs in the same manner as LOCALs.

The steps required to generate a complete multi-drive Monitor system are as follows.

- Initialize all disk cartridges using the stand-alone program DCIP.
- Punch an initial load MODE control record and system configuration deck (or tape) and insert these cards in the System Loader deck. (These records are prepared using the stand-alone utility PTUTL in the paper tape system).
- Use the System Loader to load the Monitor system to disk.
- Perform a cold start.

The complete Monitor system is now on-line and operational.

Detailed instructions for initial load and reload of the card and paper tape Monitor system are listed below. All loading and reloading is performed by the System Loader. System Loader error messages are listed in Appendix A.

CARD SYSTEM PRE-LOAD

The Monitor system for the card user is supplied on a disk cartridge and must be dumped to cards before the Initial Load procedure can be started. The dump is accomplished by loading the Monitor 2 cold start card supplied with the cartridge.

Operating Procedure

- Place the pre-load cartridge on any drive on the system and ready the drive
- Set the physical drive number of the drive containing the pre-load cartridge in console entry switches 12-15

  Switches 12-15 off, drive 0
  Switch 15 on, drive 1
  Switch 14 on, drive 2
  Switch 15, 14 on, drive 3
  Switch 13 on, drive 4

- Place the cold start card in the reader wired for IPL and ready the reader.

  -- If the IPL device is a 1442-6 or 7, place the blank cards directly behind the cold start card.

  -- If the IPL device is a 2501 and the system has a 1442, place the blank cards in the 1442 but do not ready the 1442. Make the 1442 ready when the the system WAITs after the cold start program is loaded.

- With the console Mode switch set to RUN, press IMM STOP, RESET, and PROGRAM LOAD

The format of the pre-load cartridge is such that the same cold start card that is used to fetch the Monitor System is also used to fetch the disk-to-card dump program (UCART). Thus, this dump program resides on the disk in the same place as the cold start program. During the operation of the dump the lower part of core contains the Resident Monitor, and any error condition detected by UCART, will trap to $PRET, $PST1, or $PST4 with the error indications given in appendix B of the IBM 1130 Subroutine Library, Form C26-5929. The cold start card is read in and punching begins. If the punch is a 1442-5, the first card will be blank. Throw the blank card away. If the punch runs out of cards or is not-ready as in the latter case listed above, the system executes a standard pre-operative WAIT at $PRET. Ready the punch unit and press PROGRAM START to continue. If a punch or feed error occurs, refer to the writeup on 1442 Errors and Operator Procedures in the System Library section of this manual.

The dump of the Monitor system, including RPG, - requires approximately 5000 cards.

INITIAL LOAD (CARD SYSTEM)

The user must prepare an initial load mode control card and system configuration cards (REQ) and insert these cards into the System Loader deck. These System Loader control cards must be present before the Monitor system can be loaded. An optional CORE card may also be used. See Figure 15 for the placement of these cards. The card formats are listed below.

User-Punched System Loader Control Cards

The following System Loader control cards are punched by the user (see Figure 15).

Load Mode Control Card. The load mode control card informs the System Loader whether the operation is an initial load or a reload. The load mode control card can also be used to delete the Assembler, FORTRAN Compiler, or RPG Compiler from the system. The load mode control card is placed behind the last card of the first part of the System Loader.

The format of the user-punched load mode control card is as follows:

Card Column	User Entry
1-4	MODE
8	I (initial load) or R (reload)
12	A (do not load Assembler) or blank (load Assembler)
13	F (do not load FORTRAN) or blank (load FORTRAN)
14	R (do not load RPG) or blank (load RPG)

Notes: 1. If Assembler, FORTRAN or RPG are deleted they can only be restored by an initial load.

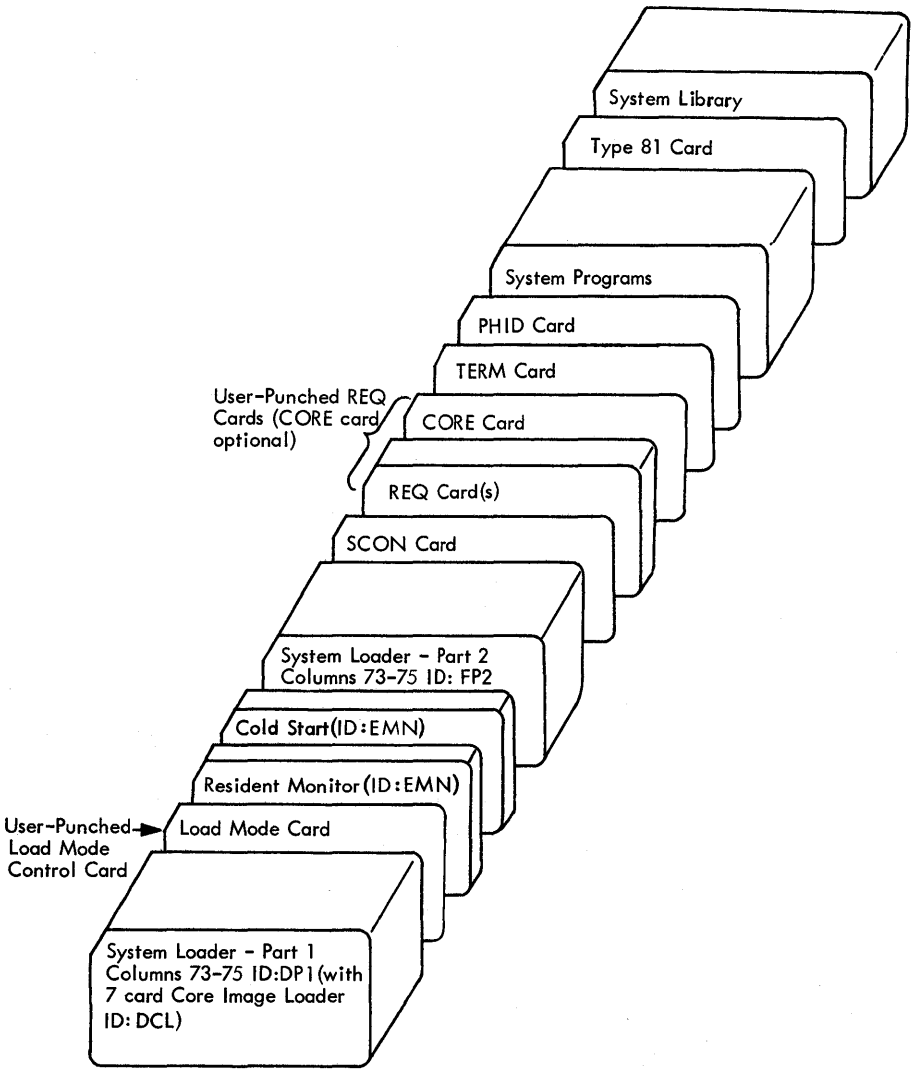


Figure 15. IBM System Load

Note 2. If any of these programs have been deleted and phases for the deleted program are present in the reload deck, the load mode control card for the reload must reflect the true system status; i.e., if FORTRAN is missing from the current system, enter an F in column 13.

System Configuration Cards (REQ). The system configuration cards are user-punched REQ cards that identify

the system I/O devices that are a part of the user's system. If an REQ card specifies the 1442, columns 15-20 of that card must contain the model number, as indicated on the REQ card format chart.

The format of the REQ cards required by the System Loader are listed below. The user should punch only those cards that identify units on the system currently being generated. Punch one card for each device. Missing or extraneous REQ cards may cause initial load operations to fail. The REQ cards must be placed between the SCON card and the TERM card in the IBM system deck.

NOTES:

1. Those I/O devices not listed on the following chart are initialized as part of the system. REQ cards are not required.
2. The principal printer is defined as the fastest printer entered on an REQ card.

Device	Card Columns 1-3	9-10*	15-20
1442 Card Read Punch Card Punch	REQ	1	1442-5 1442-6 1442-7 } whichever is applicable
Paper Tape Reader and/or Punch	REQ	3	1134 } Unit ID is optional
2501 Card Reader	REQ	4	2501
1132 Printer	REQ	6	1132
1403 Printer	REQ	9	1403

*ISS numbers, right justified. Maximum entry number ISS 20.

3. The principal I/O device is defined as follows:

Device specified on REQ Cards	Principal I/O Device
2501, 1442. PAPER TAPE	2501 Input, 1442 Output
1442, PAPER TAPE	1442 Input/Output
PAPER TAPE	Paper Tape Input/Output

CORE Card (Optional). An optional user-punched control card CORE may be placed anywhere between the SCON and TERM cards in the IBM system deck. If this card is used, the calculated (actual) core size of the system is replaced by the core size defined in the CORE card.

The format of the CORE card is as follows.

User-Defined Core Size	Card Columns 1-4	6-8
4K	CORE	04K
8K	CORE	08K
16K	CORE	16K
32K	CORE	32K

IBM-Supplied System Loader Control Cards

The following System Loader control cards are supplied with the IBM system (see Figure 15) and must be present in the IBM system deck on any system load.

SCON and TERM Card. These cards (supplied with the card system), together with the user-punched REQ cards, make up the system configuration deck. The system configuration deck must be included in the System Loader for any system load or reload. The format of the SCON and TERM cards are listed below.

SCON card, columns 1-4: SCON
TERM card, columns 1-4: TERM

Phase Identification Card (PHID). The PHID card(s) contain the beginning and ending phase numbers of the various programs in the IBM system deck. All numbers in the phase ID field of the PHID card(s) are in ascending sequence and in the order in which the system decks occur. The Resident Monitor and Cold Start Program have no phase IDs and are included in part 2 of the System Loader. The entries in the PHID card(s) are loaded into the System Location Equivalence Table (SLET) and SLET is then used by the system as an internal directory to the Monitor programs.

A primary PHID card is always necessary in the IBM system deck. A secondary PHID card may be present in addition to the primary PHID card when more than seven pairs of phase ID numbers are necessary. If a second PHID card is used its fields are formatted as shown for card one. Unused fields may be blank.

The format of the primary PHID card is as follows.

Card Columns	Entry
1-4	PHID
6-8 10-12	Phase IDs of first and last DUP phases
14-16 18-20	Phase IDs of first and last FORTRAN Compiler phases
22-24 26-28	Phase IDs of first and last Assembler phases
30-32 34-36	Phase IDs of first and last Supervisor phases
38-40 42-44	Phase IDs of first and last Core Load Builder phases
46-48 50-52	Phase IDs of first and last System Device Subroutine phases
54-56 58-60	Phase IDs of first and last Core Image Loader phases
64	'1' if a second PHID card is present
66-68	Vxx (xx is the version number)
70-72	Mxx (xx is the modification number)
73-80	Card identification and sequence number

The format of the secondary PHID card, if required, is as follows.

Card Columns	Entry
1-4	PHID
6-8	Phase IDs of first and last RPG Compiler phases
10-12	
13-65	Blank
66-68	Vxx (xx is the version number)
70-72	Mxx (xx is the modification number)
73-80	Card identification and sequence number

TYPE 81 Card. During an initial load, the type 81 card causes the principal print device and the principal I/O device entries to be placed in SLET. The Disk Communications Area (DCOM) and Location Equivalence Table (LET) are initialized and the Reload Table is established during an initial load. The IBM System Library is loaded following the reading of the type 81 card. The format of the type 81 card is as follows:

Column 3: 6 punch
Column 4: 1 punch

Operating Procedures

- Initialize a cartridge using DCIP (see Disk Cartridge Initialization Program)
- Prepare the required user-punched control records (see User-Punched System Loader Control Cards)
- Remove the Cold Start card, the stand-alone utilities, and the sample programs from behind the System Library.

After the disk cartridge has been initialized by DCIP and the user-punched System Loader control cards inserted in the IBM system deck, the Monitor system is ready to load. The complete system, ready for loading, is illustrated in Figure 15.

The steps necessary to perform a system load are as follows.

- Ready the selected disk drive
- Ready the Console Printer and the principal printer
- Set the physical drive number of the drive containing the initialized cartridge in console entry switches 12-15.

Switches 12-15 off, drive 0
Switch 15 on, drive 1
Switch 14 on, drive 2
Switch 14,15 on, drive 3
Switch 13 on, drive 4

- With the console Mode switch set to RUN, press IMM STOP and RESET.
- Place the IBM system deck in the hopper of the reader wired for initial program load (IPL).
- Press reader START. If both a 2501 and a 1142 model 6 or 7 are present, place the 1442 in a not-ready status.
- Press PROGRAM LOAD on the console.

After the type 81 card has been read, the Auxiliary Supervisor calls DUP directly to store the System Library. After the last program of the System Library has been stored, the Monitor system is on disk and can be made operational by a user-initiated cold start.

SYSTEM RELOAD (CARD SYSTEM)

The Monitor programs are divided into phases so that if changes are made within a program, only the affected phase needs to be reloaded. As in initial load, the user-punched load mode control card and REQ cards are required with the System Loader. The only difference is that the load mode control card for a reload must have an R in column 8. The programs or program phases being loaded by the reload procedure must be placed directly behind the IBM-supplied phase identification (PHID) card.

RELOAD

A program or programs specified in the second PHID record may be added to a system using the reload function. Program additions, as contrasted with phase additions, must follow the last system program currently on the cartridge. To insure that enough space exists, working storage on the cartridge must be equal to or larger than the length of the program(s) to be added, plus 31 sectors.

Additional phases for an existing program may be added as long as enough space exists in the Cushion Area to absorb the increased length of the system programs.

All phases and/or programs to be reloaded and/or added, should be in ascending phase ID sequence. Program addition(s) will require a second PHID record with the PHID range(s) specified.

Programs normally at a central location within the system program sequence cannot be added. For example, if the Assembler has been voided, it cannot be added back.

If a // XEQ MODIF record follows the Type '81' record, the reload function will link to the MODIF program to perform additional operations.

When using a 2501 Card Reader, the double-buffering procedure in the System Loader requires a blank card following the type 81 control card. The message END RELOAD will be printed by the Console Printer when the reload is completed.

If the Assembler, FORTRAN Compiler, or RPG Compiler were deleted on an initial load or by a DUP DEFINE VIOD operation, they cannot be reloaded us ing the reload procedure. They must be loaded by an initial load. If any phase of a deleted program are present in the reload deck, the apprepriate codes must be specified in column 12-14 of the reload load mode control card.

A useful option provided by the reload function is the ability to reconfigure a system cartridge with different I/O devices. Reconfiguration will be necessary if a system cartridge is copied from a system with a different contiguration. The reload deck listed below will perform this function. (To reconfigure only, place the Type 81 card directly after the PHID cards.)

- System Loader deck, part 1, with Core Image Loader
- Load mode control card (R in column 8)
- Resident Monitor/cold start deck
- System Loader deck, part 2
- System configuration deck:
  - SCON card
  - REQ cards
  - CORE card (optional)
  - TERM card

  } System reconfigured if desired
- PHID cards
- (Revised programs or program phases)*
- Type 81 control card
- Blank card

* All decks must have phase ID numbers within the limits of the IDs listed on the PHID cards.

If phases are to be reloaded the last card of the last phase must be an end-of-program card. (See End-of-Program (EOP) card, Appendix C). In this case the EOP card may have words 1, 2 and 4 through 54 blank.

During a reload operation, loading terminates with the reading of the type 81 card, and the printing of END RELOAD.

Note: If the last job on the cartridge being reloaded was in temporary mode, a Cold Start should be performed prior to the Reload in order to reset the temporary mode switch and delete temporary items.

Therefore it is recommended that a Cold Start be performed before each SYSTEM RELOAD.

Operating Procedures

With the console Mode switch set to RUN, press PROGRAM STOP on the console.

- Ready the selected disk drive.
- Ready the Console Printer.
- Set the physical drive number of the drive containing the cartridge to be reloaded in console entry switches 12-15.
  Switches 12-15 off, drive 0
  Switch 15 on, drive 1
  Switch 14 on, drive 2
  Switch 14, 15 on, drive 3
  Switch 13 on, drive 4
- Press RESET on the console.
- Place the reload deck (see listing above) in the reader wired for IPL.
- Press reader START. If both a 2501 and a 1442 model 6 or 7 are present, place the 1442 in a not-ready status.
- Press PROGRAM LOAD on the console.
- Perform a cold start to make the revised Monitor system operational.

System Program Phase Sector Break Cards

In order to allow the user to load only a portion of a Monitor program, the programs are divided into phases, each identified by a sector break card.

The sector break cards identifying the phase of the IBM system programs are listed below. Sector break cards (see Appendix C) have a 1 punch in column 4. The version and modification level are punched in the cards starting at column 67 (VxMxx).

Phase Number	Program or Program Phase Name	ID Starting in Column 73
XX	RES SKELETON SUPY, COMMA, DISKZ, COLD START PROGRAM Part of System Loader	EMN
XX	SYS LDR - PHASE 2 - OVERLAY 0	FP2
XX	SYS LDR - PHASE 2 - OVERLAY 1	FP2
XX	SYS LDR - PHASE 2 - OVERLAY 2	FP2
XX	SYS LDR - PHASE 2 - OVERLAY 3	FP2
	DUP	
01	DUP COMMON SUBROUTINES, CCAT	J01
02	DUP CTRL RECORD PROCESSOR	J02
03	DUP STORE PHASE	J03
04	DUP *FILES, *LOCAL, *NOCAL PHASE	J04
05	DUP DUMP PHASE	J05
06	DUP DUMP LET/FLET PHASE	J06
07	DUP DELETE PHASE	J07
08	DUP DEFINE PHASE	J08
09	DUP EXIT PHASE	J09
0A	DUP CARD I/O INTERFACE	J10
0B	DUP KEYBOARD INPUT INTERFACE	J11
0C	DUP PAPER TAPE I/O INTERFACE	J12
0D	DUP UPCOR PHASE SAVED BY DEXIT DURING STORECI	J17
0E	DUP PRINCIPAL INPUT WITH KEYBOARD	J17
0F	DUP PRINCIPAL W/O KEYBOARD	J17
10	DUP PAPER TAPE I/O	J17
11	DUP STORE CI	J17
12	DUP MODIF DUMMY PHASE	J17
	FORTRAN Compiler	
1F	FOR INPUT PHASE	K01
20	FOR CLASSIFIER PHASE	K02
21	FOR CHECK ORDER/STMNT NO. PHASE	K03
22	FOR COMMON SUBR OR FUNCTION PHASE	K04
23	FOR DIMENSION, REAL, INTEGER	K05
24	FOR REAL CONSTANT PHASE	K06
25	FOR DEFINE FILE, CALL LINK EXIT	K07
26	FOR VARIABLE, STMNT FUNC PHASE	K08
27	FOR DATA STATEMENT PHASE	K09
28	FOR FORMAT STATEMENT PHASE	K10
29	FOR SUBTRACT DECOMPOSITION PHASE	K11
2A	FOR ASCAN I PHASE	K12
2B	FOR ASCAN II PHASE	K13
2C	FOR DO, CONTINUE, ETC. PHASE	K14
2D	FOR SUBSCRIPT OPTIMIZE PHASE	K15
2E	FOR SCAN PHASE	K16
2F	FOR EXPANDER I PHASE	K17
30	FOR EXPANDER II PHASE	K18
31	FOR DATA ALLOCATION PHASE	K19
32	FOR COMPILATION ERROR PHASE	K20
33	FOR STATEMENT ALLOCATION PHASE	K21
34	FOR LIST STATEMENT ALLOCATION	K22
35	FOR LIST SYMBOL TABLE PHASE	K23
36	FOR LIST CONSTANTS PHASE	K24
37	FOR OUTPUT I PHASE	K25
38	FOR OUTPUT II PHASE	K26
39	FOR RECOVERY (EXIT) PHASE	K27

Phase Number	Program or Program Phase Name	ID Starting in Column 73
	ASSEMBLER	
51	ASM INITIALIZATION PHASE	M01
52	ASM CARD CONVERSION PHASE	M02
53	ASM DSF OUTPUT PHASE	M03
54	ASM INTERMEDIATE INPUT PHASE	M04
55	ASM END STATEMENT PHASE	M05
56	ASM ASSEMBLY ERROR PHASE	M06
57	ASM CONTROL CARDS 1	M07
58	ASM CONTROL CARDS 2	M08
59	ASM DUMMY PHASE (SYST SYMBOL TBL)	M09
5A	ASM SYMBOL TABLE OPTIONS PHASE	M10
5B	ASM EXIT PHASE	M11
5C	ASM PROG HEADER MNEMONICS PHASE	M12
5D	ASM FILE STATEMENT PHASE	M13
5E	ASM COMMON SUBROUTINES, ASCOM	M14
5F	ASM PROG CONTROL MNEMONICS PHASE	M15
60	ASM IMPERATIVE STATEMENTS PHASE	M16
61	ASM DECML EFLC PROCESSING PHASE	M17
62	ASM DECIMAL CONVERSION PHASE	M18
63	ASM PROG LINKING PHASE	M19
64	ASM DMES PROCESSING PHASE	M20
65	ASM PUNCH CONVERSION PHASE	M21
66	ASM INTERMEDIATE DISK OUTPUT	M22
67	ASM SYMBOL TABLE OVERFLOW	M23
68	ASM G2250 PH1	M24
	SUPERVISOR	
6E	SUP PHASE 1 - MONITOR CONTROL RECORD ANALYZER	N01
6F	SUP PHASE 2 - JOB CONTROL RECORD PROCESSOR	N01
70	SUP PHASE 3 - DELETE TEMPORARILY STORED PROGRAM LET	N01
71	SUP PHASE 4 - XIO CONTROL RECORD PROCESSOR	N01
72	SUP PHASE 5 - SUPERVISOR CONTROL RECORDS PROCESSOR	N01
73	SYSTEM DUMP-CORE-TO-PRINTER	N02
74	AUXILIARY SUPERVISOR	N03
	SUP PHASE 5 - G2250 PROCESSING	
	CORE LOAD BUILDER	
78	CORE LOAD BUILDER, PHASE 0/1	OCB
79	CORE LOAD BUILDER, PHASE 2	OCB
7A	CORE LOAD BUILDER, PHASE 3	OCB
7B	CORE LOAD BUILDER, PHASE 4	OCB
7C	CORE LOAD BUILDER, PHASE 5	OCB
7D	CORE LOAD BUILDER, PHASE 6	OCB
7E	CORE LOAD BUILDER, PHASE 7	OCB
7F	CORE LOAD BUILDER, PHASE 8	OCB
80	CORE LOAD BUILDER, PHASE 9	OCB
81	CORE LOAD BUILDER, PHASE 10	OCB
82	CORE LOAD BUILDER, PHASE 11	OCB
83	CORE LOAD BUILDER, PHASE 12	OCB
84	CORE LOAD BUILDER, PHASE 13	OCB
	SYSTEM DEVICE SUBROUTINES, DISK I/O	
8C	SYS 1403	PMN
8D	SYS 1132	PMN
8E	SYS CONSOLE PRINTER	PMN
8F	SYS 2501	PMN
90	SYS 1442	PMN
91	SYS 1134	PMN
92	SYS KEYBOARD	PMN

Phase Number	Program or Program Phase Name	ID Starting in Column 72
93	SYS 2501/1442 CONVERSION	PMN
94	SYS 1134 CONVERSION	PMN
95	SYS KEYBOARD CONVERSION	PMN
96	DISKZ	PMN
97	DISK1	PMN
98	DISKN	PMN
	CORE IMAGE LOADER	
A0	CORE IMAGE LOADER, PHASE 1	PMN
A1	CORE IMAGE LOADER, PHASE 2	PMN
	RPG Compiler	
B0	RESIDENT	PR0
B1	ENTER FILES	PR1
B2	ENTER INPUT	PR2
B3	ENTER CALCULATION	PR3
B4	ENTER OUTPUT	PR4
B5	ASSIGN INDICATORS	PR5
B6	ASSIGN FIELD NAMES	PR6
B7	ASSIGN LITERALS	PR7
B8	EXTENDED FILE AND INPUT DIAGNOSTIC	PR8
B9	EXTENDED CALCULATION AND OUTPUT DIAGNOSTIC	PR9
BA	DIAGNOSTIC MESSAGE 1	PRA
BB	DIAGNOSTIC MESSAGE 2	PRB
BC	DIAGNOSTIC MESSAGE 3	PRC
BD	ASSEMBLE 1 I/O	PRD
BE	ASSEMBLE 2 I/O	PRE
BF	ASSEMBLE 3 I/O	PRF
C0	ASSEMBLE 4 I/O	PRG
C1	ASSEMBLE TABLES	PRH
C2	ASSEMBLE CHAIN AND RAF	PRI
C3	ASSEMBLE INPUT FIELDS	PRJ
C4	ASSEMBLE CONTROL LEVELS	PRK
C5	ASSEMBLE MULTI FILE LOGIC	PRL
C6	ASSEMBLE GET ROUTINES	PRM
C7	ASSEMBLE CALCULATIONS 1	PRN
C8	ASSEMBLE CALCULATIONS 2	PRO
C9	ASSEMBLE OUTPUT FIELDS	PRP
CA	ASSEMBLE PUT ROUTINES	PRQ
CB	ASSEMBLE FIXED DRIVER	PRR
CC	TERMINATE COMPILATION	PRS

INITIAL LOAD (PAPER TAPE SYSTEM)

The tapes constituting the complete Paper Tape Monitor System, including the user-punched control record tapes are listed below.

Tape Number	Description
1	System Loader, Part 1
-	Load Mode Control Record (User-punched)
2	System Loader, Part 2, with Resident Monitor and Cold Start
-	System Configuration Records (User-punched)
3	Phase Id. (PHID) Control Record
4	Disk Utility Program
5	FORTRAN Compiler
6	Assembler
7	Supervisor, Core Load Builder, System I/O Subroutines, Core Image Loader
8	End of System Tapes Control Record (Type 81 record)
9	Standard LIBFs and CALLs
10	Extended Precision LIBFs and CALLS
11	Common LIBFs and CALLs
12	ILS, ISS, Conversion and Utility Subroutines
13	Plotter Subroutines
14	SCA Subroutines
15	Cold Start Paper Tape Record
16	DCIP Disk Cartridge Initialization Program
17	PTUTL Paper Tape Utility Program
18	Paper Tape Reproducing Program
19	1132/1403 Printer Core Dump from /01E0
20	Console Printer Core Dump

Tape 15 is used to initialize the Monitor system after it is loaded. Tapes 16-20 are stand-alone utilities and are not loaded as part of the Monitor System; however, PTUTL and DCIP are used during the loading process. Tapes 21 and 22 are the Monitor system sample programs

NOTE: If the FORTRAN Compiler and/or the Assembler are not be be loaded during an initial load, the corresponding tapes (5 and/or 6) need not be read. If the FORTRAN Compiler and/or the Assembler are not loaded, they cannot be loaded using the reload procedure. They must be loaded by an initial load.

System Loader Control Records

With the exception of the Load Mode Control Record and the System Configuration Records, all of the paper tape control records needed to load the Paper Tape Monitor System to disk storage are supplied to the user by IBM. These control records have the same functions as the corresponding IBM-supplied and user-punched control cards (see Initial Load (Card System)). The Load Mode Control record and System Configuration records must be prepared by the user. If these tapes are not prepared correctly, the System Loader will print an error message during system load (see Appendix A). A user-punched CORE record is optional.

Preparation of Load Mode and System Configuration Control Tapes

Paper tape control records must be punched in PTTC/8 (Perforated Tape Transmission Code). The formats are the same as the previously-described card formats. Paper tape control records must be separated by one NL (new line) control character. A control record that immediately follows paper tape data not followed by an NL code must be preceded by one NL code. Delete codes may precede or follow this NL code.

To initially generate a system cartridge the necessary control records can be punched using a stand-alone paper tape utility program (PTUTL).

To load the PTUTL program tape, perform the following steps:

- Place the PTUTL tape in the Paper Tape Reader, positioning the tape so that one of the delete codes beyond the program ID in the tape leader is under the read starwheels.
- With the console Mode switch set to RUN, press IMM STOP, RESET, and PROGRAM LOAD.
- PTUTL is read in and the program WAITs with /1111 in the Accumulator.
- Set console entry switches 2 and 3 on. Functions requested by these switches are:

  Switch 2-accept Keyboard input
  Switch 3-punch records on the 1055 Paper Tape Punch

  NOTE: Complete operating procedures for PTUTL are contained in the writeup for the System Library version of the Paper Tape Utility Program (see Paper Tape Utility (PTUTL).

- Ready the Paper Tape Punch. Be sure to punch a leader of delete codes.
- Use the Keyboard to prepare the user-punched System Loader control records.

Paper Tape Load Mode Record

Steps in preparation are:

- Write MODE using Keyboard input.
- Space 3 times.
- Write I or R for initial load or reload operation.
- Space 3 times.
- If the Assembler is not to be loaded, write A; otherwise space 1.
- If the FORTRAN Compiler is not to be loaded, write F; otherwise space 1.
- Press EOF on the Keyboard if no mistakes were made, otherwise press ERASE FIELD and repeat the above procedure.
- Create a trailer (and new leader) of delete codes on the paper tape punch.

Paper Tape System Configuration Tape

Steps in preparation are:

- Write SCON using keyboard input.
- Press EOF to end the SCON record.
- Write REQ
- Space 6 (or 5 in cases of a 2 digit ISS number)
- Write the ISS number for an I/O device to be configured into the system (see System Configuration Cards (REQ) for the required ISS numbers).
- Press EOF. Repeat the preceding three steps until all necessary REQ records have been punched.
- A CORE record may be added if desired. Its format is identical to the card system description.
- Write TERM
- Press EOF to end the TERM record and Configuration tape.
- Create a trailer of delete codes on the Paper Tape Punch.

Operating Procedure

- Initialize a cartridge using DCIP (see Disk Cartridge Initialization Program)
- Prepare the required user-punched control records (see Preparation of Load Mode and System Configuration Control Tapes)

After the disk cartridge has been initialized by DCIP and the user-punched System Loader control record tapes generated, the Monitor system is ready to load.

The steps necessary to perform a system load are as follows.

- Ready the selected disk drive
- Ready the Console Printer and principal printer
- Set the physical drive number of the drive containing the initialized cartridge in console entry switches 12-15.
  Switches 12-15 off, drive 0
  Switch 15 on, drive 1
  Switch 14 on, drive 2
  Switch 14, 15 on, drive 3
  Switch 13 on, drive 4
- Place the System Loader Part 1 (Tape 1) in the Paper Tape Reader.
  When loading tapes, position any of the delete codes following the program ID in the tape leader under the read starwheels.
- With the console Mode switch set to RUN, press IMM STOP, RESET, and PROGRAM LOAD
  Tape 1 is read in and the system WAITs at $PST4 or $PRET.
- Place the user-punched Load Mode control record tape in the reader and press PROGRAM START.
  This tape is read in and the system again waits at $PST4 or $PRET.
- Place the System Loader Part 2 (Tape 2) in the reader and press PROGRAM START. The system will WAIT after loading.
- Place the user-punched System Configuration Tape in the reader and press PROGRAM START. The system will WAIT.
- Load tapes 3 through 14 as required using the same procedure.

NOTE: If the FORTRAN Compiler and/or Assembler are to be deleted, tapes 5 and/or 6 need not be loaded. Load only those System Library Tapes (9 through 14) that are required for your system.

After the last required System Library Tape has been loaded, the Monitor system is on disk and can be made operational by a user-initiated cold start.

SYSTEM RELOAD (PAPER TAPE SYSTEM)

During a reload of system programs or a system reconfiguration, all System Loader Control record tapes must be used. A typical paper tape reload would include:

Tape 1
User-punched Load Mode Control record
(R for reload)

Tape 2
User-punched System Configuration tape (revised if system is being reconfigured)

Tape 3
(Revised programs or program phases)*

Tape 8

*All programs must have phase ID numbers within the limits of the IDs listed on the PHID tape.

If the Assembler or FORTRAN Compiler were deleted on initial load or deleted by a DUP DEFINE VOID operation, they cannot be reloaded using the reload procedure. They must be loaded by an initial load.

For further information regarding reload, see System Reload (Card System).

NOTE: The Paper Tape System Loader does not link to MODIF as the Card System Loader does.

1130 DISK DATA FILE CONVERSION PROGRAM (DFCNV)

DFCNV converts 1130 FORTRAN and/or Commercial Subroutine Package (1130-SE-25X) disk data files to disk files acceptable to 1130 RPG. FORTRAN disk files created using logical unit number 10 cannot be converted by DFCNV. Converted files may be processed sequentially or randomly but not as ISAM files. The program operates in a minimum 8K core DM2 system and uses DISK1 and the system device subroutines for the principal input device and the principal printer.

The program accepts all FORTRAN and Commercial Subroutine Package (CSP) disk data formats and two-word integers for conversion to acceptable RPG disk data format.

Prior to executing DFCNV, a DUP STOREDATA operation must be performed to reserve an output file in the User/Fixed Area and to enter its file name in LET/FLET.

Program input may be a disk file created by FORTRAN or CSP with or without two-word integers or the corresponding cards produced by a DUP DUMPDATA operation. The output file may be defined on the same disk cartridge as the input file or on a cartridge residing on another drive. DFCNV converts one input file to one output file; subsequent program executions must be performed to convert more than one file. The program can operate in a stacked job environment. The calling sequence is

Col.	1	4	8	19
	//	XEQ	DFCNV	1

Three types of control cards are required by the conversion program. Each control card is printed on the principal printer as it is read. Diagnostic messages are printed as errors are detected on the card being processed. All diagnostics except the warning messages cause program termination. If any error is detected on the File Description card, program termination is immediate; all other errors are diagnosed before program termination. Following successful processing of the control cards, file conversion takes place and the following message is printed.

DISK DATA FILE CONVERSION COMPLETED

NOTE: The Disk File protection delimiters $FPAD--$FPAD+4 of COMMA are modified during the conversion portion of DFCNV. These modified COMMA words must be restored prior to further processing if unforseen problems (accidential immediate stop) cause abnormal termination of DFCNV. These words are restored by DFCNV under normal program termination following successful conversion.

The first required control card is the File Description card which must immediately follow the //XEQ DECNV monitor control card. Only one File Description card is allowed by the program and it contains the following information:

Col.	Description
1-5	File name of file whose data is to be converted (must be left-justified). This field is ignored if card input is specified in column 31.[1]
7-11	File name of file where the RPG data is to be placed (must be left-justified). [1,3]
13-17	Number of records contained in the file (1-32767).[2]
19-21	Record size of the input file in words (1-320)[2]
23-25	Record size of the RPG file in characters (1-640).[2]
27	An indication of standard (S) or extended (E) precision. Any character other than S or E in this column is invalid.
29	An indication that one-word integers are used (1). This column must be blank if one word integers are not used.
31	An indication that the input file is a punched card deck (C). This column must be blank if disk file input is used.
33	An indication (W) that an object time warning message is to be given if a real number is out of range (see note under R-Field Type) upon conversion. This column must be blank if the warning option is not used.
72	The character D, identifying the card as a File Description card.

[1]A file name is a symbol and therefore must conform to the rules for symbols as stated in IBM 1130 Assembler Language, Form C26-5927.

[2]This field must be right-justified with leading zeroes or blanks.

[3]The RPG file name cannot contain any special characters, although the input file name may contain the character $. No provision has been made in DFCNV to check the RPG file name for $.

Both the input and RPG file sizes are calculated from the information given on the File Description card and are checked against their corresponding LET/FLET entries for proper size. The following

equations are used to calculate the file sizes.

Input file size in sectors = number of records/(320/input record size in words).

Output file size in sectors=number of records/(number of records that can be contained in each sector).

Output record size in words=number of records+1/(number of records that can be contained in each sector).

Note 1. If the above formulae produce answers with fractional parts, the file and record size are obtained by rounding off to the next higher whole number.

Note 2. The number of records that can be contained in each sector (the denominators of the first two formulae)=320/(record size in words). The remainder, if any must be ignored. 320 is the number of words per sector.

The second required control card(s), the Field Specification card, describes the data to be converted. DFCNV allows selective field conversion (X-field type) and rearrangement of data (m term of field types) within the converted field. It is the user's responsibility to ensure that data rearrangement and field expansion do not cause data overlay in the converted record. As many complete field descriptions may be used on each Field Specification card as can be placed in columns 1 through 71. Column 72 of each Field Specification card must contain an S. Field descriptions must be placed on the card(s) in the same order as the fields appear on the input record and must be separated by commas (e.g., FS1, FS2, . .). No embedded blanks within specifications or intervening blanks between specifications are permitted.

An optional card containing the 40 character translation table for CSP A3 format and the character A in column 72 must be included in the control cards if any F-type conversions are specified on the Field Specification card(s). This conversion table must correspond to the original table used to convert to CSP A3 format. Only one table is allowed per file; if more than one table is included in the control cards, tables subsequent to the first are ignored.

The third required card is the end-of-file record (/* card). All other DFCNV control cards must precede the /* card; if any card other than a DFCNV control card precedes the /* card, a warning message is printed and the next card is read.

The following is a description of each field type supported by the program. In each of these specification descriptions the column and field length indicators may vary from 1 to 3 digits in length; all other numeric indicators must be one digit in length.

I-Field Type

This field type describes FORTRAN integer conversion; input is an integer field. The specification is

m-Iw.t(P)

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- I identifies the field type,
- w is the field length of the converted field (maximum of 14),
- t is the number of decimal positions reserved in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Note: Since the FORTRAN integer field is regarded as a whole number with no decimal places, up to five positions to the left of the decimal point should be reserved in the converted field to hold the largest possible integer value. Alignment is at the decimal point; if five positions are not reserved, high order truncation occurs (see Diagnostic Capabilities).

J-Field Type

This field type describes two-word integer conversion; input is a two-word integer. The specification is

m-Jw.t(P)

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- J identifies the field type,
- w is the field length of the converted field (maximum of 14),
- t is the number of decimal positions reserved in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Note: Since a two word integer is regarded as a whole number with no decimal places, up to ten positions to the left of the decimal point should be reserved in the converted field to hold the largest possible integer value. Alignment is at the decimal point; if ten positions are not reserved, high order truncation occurs (see Diagnostic Capabilities). If a file contains two-word integers, standard precision must be specified on the File Description card. If extended precision is specified, any J-field type specification is invalid.

R-Field Type

This field type describes FORTRAN real variable conversion. The specification is

m-Rw.t(P)

where m is the column of the RPG record in

which the converted field begins (1 through 640),

R identifies the field type,

w is the field length of the converted field (maximum of 14),

t is the number of decimal positions reserved in the RPG field (maximum of 9),

(P) is optional and is present only if the RPG field is to be packed.

Note: If the real number of the input field is too samll to yield any significant difits in the RPG field, the RPG field is set to zeroes. If the real number is too large to yield any significant digits in the RPG field, the RPG field is set to nines, (See Diagnostic Capabilities).

B-Field Type

This field type describes FORTRAN A-conversion for integer data and CPS A1 and A2 conversion. The specification is

m-Bw.n

where

m is the column of the RPG record in which the converted field begins (1 through 640),

B identifies the field type,

w is the number of characters in the field (maximum of 255),

n is the number of characters in each unit of the input field (n=1 or 2).

Note: If CSP A1 or A2 format is converted, one word integers must be specified on the File Description card; however, no diagnostic check is made for this condition.

C-Field Type

This field type describes FORTRAN A-conversion for real data. The specification is

m-Cw.n

where

m is the column of the RPG record in which the converted field begins (1 through 640),

C identifies the field type,

w is the number of characters in the field (maximum of 255),

n is the number of characters in each unit (2 or 3 words) of the input field. For standard precision, n may range from 1 through 4; for extended precision, from 1 through 6.

D-Field Type

This field type describes CSP D1 conversion. The specification is

m-Dl_1.j=l_2.k(P)

where

m is the column of the RPG record in which the converted field begins (1 through 640),

D identifies the field type,

l_1 is the length of the CSP field (maximum of 255),

j is the number of decimal positions in the CSP field, (Maximum of 9).

l_2 is the length of the RPG field (maximum of 14),

k is the number of decimal positions in the RPG field (maximum of 9),

(P) is optional and is present only if the RPG field is to be packed.

Note: Alignment is at the decimal point. If, for example, $l_1=l_2$ and $k > j$, then k-j high order positions of the CSP field are truncated in the RPG field (see Diagnostic Capabilities).

E-Field Type

This field type describes CSP D4 conversion. The specification is

m-El_1.j=l_2.k(P)

where

m is the column of the RPG record in which the converted field begins (1 through 640),

E identifies the field type,

l_1 is the length of the CSP field (maximum of 255),

j is the number of decimal positions in the CSP field, (Maximum of 9).

l_2 is the length of the RPG field (maximum of 14),

k is the number of decimal positions in the RPG field (maximum of 9),

(P) is optional and is present only if the RPG field is to be packed,

Note: For E-field type conversion, alignment is also performed at the decimal point; high order truncation is possible (see Diagnostic Capabilites).

F-Field Type

This field type describes CSP A3 conversion. It requires a 40 character translation table. The specification is

m-Fw

where

m is the column of the RPG record in which the converted field begins (1 through 640).

F identifies the field type,

w is the number of characters in the field (not to exceed output record size in characters).

X-Field Type

This field type allows fields on the input record to be bypassed. The specification is

Xw

where X identifies the field type,
w is the number of words to be bypassed (not to exceed input record size).

Repeat Specification Option

It is possible to specify sequentially repeated identical fields using only one field type specification for any field type except the X-field type. The repeat option is implemented by immediately following the specification to be repeated with the character R and the total number of identical fields. The repeated field begins in the first vacant output column following the previous field (i.e., no skipping is allowed within a repeated specification).

Example: The following field specification describes three integer fields the first of which begins in column 15 of the RPG record, each of which is packed and 5 characters in length with 2 decimal places.

15-15.2(P)R3

The three resulting output fields are placed starting in the eight word of the record as follows:

Word:	8	9	10	11	12
Content:	XXX0	0FYY	Y00F	ZZZ0	0F40

where XXX, YYY and ZZZ represent the three integer fields.

Diagnostic Capabilities

All DFCNV error messages and their identifying numbers (listed below) are printed on the principal printer. All diagnostics are printed before data conversion begins. Any diagnostic except warning messages (indicated below) causes program termination.

No.	Message	Cause
F01	INVALID DESCRIPTION CARD FIELD-COL. XX	1) 1. Numeric field at card column XX outside allowable field range. 1) 2. Unrecognizable character in field at card column XX.
F02	FILE NAME NOT IN LET/FLET-Y	1) 1. LET/FLET entry not found for file named on File Description card. 1) 2. File name given on File Description card invalid. Y=1, input file error =0, output file error.
F03	FILE SIZE INVALID -Y	1) File size calculated from File Description data exceeds actual file size.
F04	INVALID FIELD SPECIFICATION SYNTAX-COL. XX	1. Numeric field of specification at card column XX outside allowable field range. 2. Unrecognizable character in field of specification at card column XX. 3) 3. Embedded or intervening blanks on Field Specification card. 4. J-field type specification detected at card column XX when extended precision was specified
F05	CSP A3 TABLE MISSING	No A (col 72) card precedes / * card when F-field specified.
F06	INVALID CARD SEQUENCE	2) 1. Unrecognizable card precedes / card (column 72 not D, S, or A). 1) 2. Multiple File Description cards read. 1) 3. File Description card out of order. 1) 4. No Field Specification card precedes / card.
F07	TRUNCATION OCCURS AT COL. XXX	2) High order truncation occurs in output field at column XXX.
F08	CARD INPUT INVALID	1) Card input is specified when principal input device is console keyboard.
F09	OUTPUT RECORD LENGTH INVALID	Sum of individual field lengths exceeds specified record length for output.
F10	FIELD OUT OF RANGE AT COL. XXX OF RECORD YYYYY	2) RPG real number field starting at column XXX has been set to zeroes or nines in record YYYYY.

1) Program termination immediate.
2) Warning only.
3) No column indication given.

The Supervisor initially achieves control over the 1130 Monitor System through the user-initiated Cold Start procedure. The Cold Start procedure begins with the IPL (Initial Program Load) of the Cold Start record, which causes the Cold Start program to be read into core storage from the system cartridge and control to be transferred to it.

The Cold Start program, in turn, loads the Resident Monitor into its location in lower core storage. The Cold Start procedure ends when control is given to the job initialization program in the Supervisor.

NOTE: Do not perform a cold start with an uninitialized cartridge on line.

Cold Start Procedure

To perform a cold start:

- Ready the principal print device.
- Set the physical drive number of the drive containing the system cartridge in console entry switches 12-15.
  Switches 12-15 off, drive 0
- Switch 15 on, drive 1
  Switch 14 on, drive 2
  Switch 15, 14 on, drive 3
  Switch 13 on, drive 4
- Ready the selected disk drive.
- Press IMM STOP and RESET on the console.
- Ready the Console Printer.
- Place the cold start record in the reader wired for IPL (Tape 15, paper tape system).
- Press reader START. If both a 2501 and 1442 model 6 or 7 are present, place the 1442 in a not-ready status.
- Press PROGRAM LOAD on the console.

When the Cold Start record is read, a dummy // JOB record is printed on the principal printer and the Supervisor prints cartridge status information as follows.

```
LOG DRIVE   CART SPEC   CART AVAIL   PHY DRIVE

   XXXX        XXXX         XXXX        XXXX
```

where

LOG DRIVE is always zero

CART SPEC is the cartridge ID written on the cartridge by DCIP.

CART AVAIL is the same as CART SPEC.

PHY DRIVE is the physical drive number selected in the console entry switches. This physical drive is now logical zero.

The Monitor system is now operational and is ready to receive the first JOB record from the reader. If an attempt is made to cold start a non-system cartridge, an error message -- THIS IS A NON-SYSTEM CARTRIDGE. -- is printed on the Console Printer.

The table below lists the error stops contained in the Cold Start Loader (i.e., card or paper tape).

Absolute Address	Explanation
/0012	-Invalid disk drive number in Console Entry Switches -Indicated disk drive not ready
/0044	-Disk read error -Waiting for interrupt from seek operation
/0046	-Waiting for interrupt from reading sector @ IDAD

These utility programs -- each self-loading and complete with subroutines -- are separate from the System Library and enable the user to perform operations without Monitor system control. The first three programs are available in card and paper tape, the last two in paper tape only. The utility programs are:

- Console Printer Core Dump
- Printer Core Dump
- Disk Cartridge Initialization Program (DCIP)
- Paper Tape Utility (PTUTL)
- Paper Tape Reproducing

CONSOLE PRINTER CORE DUMP

This program aids the user in debugging programs by dumping selected portions of core on the Console Printer.

Format

Each core location is dumped as a four-digit hexadecimal word with a space separating each word. The first word dumped is the starting address of the dump (as specified in the console entry switches).

Operating Procedures

- With the console Mode switch set to RUN, press IMM STOP and RESET on the console.
- Place the Console Printer Core Dump program in the reader wired for IPL and ready the reader (if the system configuration is 2501, 1442-6 or -7, make the 1442 not-ready).
- Set the margin on the Console Printer. To print the same format on each line set the number of print positions to a multiple of 5.
- Set the starting address (in hexadecimal) in the console entry switches.
- Press PROGRAM LOAD.

Dumping continues until IMM STOP is pressed. To continue, press PROGRAM START.

PRINTER CORE DUMP

This program dumps core in hexadecimal format on either the 1403 Printer or the 1132 Printer, whichever is in a ready status. If both are ready, the dump will be on the 1403.

NOTE: "Not present" is equivalent to "not ready".

Format

Dumping starts at location $ZEND. Each line contains a four-digit hexadecimal address, followed by 16 four-digit hexadecimal words. A space separates the address and each word in the printed line. An additional space is inserted between each group of four words.

To decrease dump time, the program does not print consecutive duplicate lines. Before printing a line, it compares the next 16 words with the 16 words just printed. If they are identical, the program goes on to the next 16 words in core. If they are not identical, the printer spaces one line and prints. The address printed is that of the first word on the line.

Operating Procedures

- With the console Mode switch set to RUN, press IMM STOP and RESET.
- Place the Printer Core Dump program in the reader wired for IPL and ready the reader. (If the system configuration is 2501, 1442-6 or -7, make the 1442 not-ready.)
- Ready the printer.
- Press PROGRAM LOAD.

Dumping starts at location $ZEND and continues to the end of core. The user may halt the dump at any time by pressing IMM STOP. Press PROGRAM START to continue on the 1403. The 1132 has no restart capabilities.

DISK CARTRIDGE INITIALIZATION PROGRAM (DCIP)

The Disk Cartridge Initialization Program (DCIP) is composed of

- A disk initialization subroutine
- A disk copy subroutine
- A disk dump subroutine
- A disk patch subroutine
- A disk analysis subroutine
- A disk compare subroutine

Initialization

If sector @IDAD and/or sector @DCOM on the disk are destroyed DCIP will not work properly with the exception of the initialize option.

- Writes a sector address on every sector, including defective sectors.
- Determines which, if any, sectors are defective and fills in the defective cylinder table accordingly.
- Establishes a file-protected area for the disk cartridge.
- Puts an ID on the disk cartridge.
- Establishes a DCOM, LET, and CIB.

Initialization of a cartridge is required before the Monitor system can be loaded.

The disk I/O subroutines operate with up to three defective cylinders, i.e., three cylinders that contain one or more defective sectors.

Cylinder zero must not be a defective cylinder; otherwise, the cartridge cannot be initialized. Likewise, it must be possible to write at least a sector address on every sector.

At the completion of disk initialization, several words are written on sector @IDAD. The three words starting at word @DCTB contain the address of sector zero of any defective cylinders found (maximum of three). When there are no defective cylinders, these words contain /0658, e.g., the table for a cartridge with a defect only in sector 9 (cylinder 1), would contain:

/0008
/0658
/0658

Word @CIDN contains the cartridge ID. Word @COPY, the copy code, contains zero. Word @DTYP contains a minus 2, indicating a DM2 non-system cartridge or a minus 1, indicating an initialized DM1 cartridge. Except for the non-system cartridge error message program, which starts at @CSTR, the rest of sector @IDAD contains zeros. The error message program is substituted for the cold start program as the cartridge is initialized.

After sector @DCOM has been cleared to zeros, certain parameters are initialized to indicate that this is a non-system cartridge. The parameter set, including their initial values, are listed below:

#ANDU	/0180 (disk block address)	End of User Area, adjusted (update during JOB T)
#BNDU	/0180 (disk block address)	End of User Area, base
#FPAD	/0018 (sector address)	File protect address on this cartridge
#CIDN	XXXX	Cartridge ID of this cartridge
#CIBA	/0008 (sector address)	First sector of CIB on this cartridge
#ULET	/0002 (sector address)	First sector of LET on this cartridge

An initial LET is also created on sector @RIAD. Its contents are as follows:

Word 1	LET sector number	/0000	
Word 2	Sector address of UA	/0018	
Word 3	Reserved	/0000	
Word 4	Words available in this sector	/0138	
Word 5	LET/FLET chain address	/0000	(Last LET/FLET sector)
Word 6	1st Word of 1DUMY entry	/7112	1DUMY in packed truncated EBCDIC
Word 7	2nd Word of 1DUMY entry	/4528	
Word 8	Size of 1DUMY	/6280	(Size of WS available in disk blocks)

Words 9-320 of @RIAD all contain zero.

Copy

The disk copy subroutine of DCIP

- Checks to ensure that both the cartridge to be copied and the cartridge onto which the copy is to be made have been correctly initialized.
- Copies a cartridge from any drive onto a cartridge on any other drive, making allowances for defective cylinders. The cartridge ID, copy code, and defective cylinder table are not copied from the source cartridge. Both Version 1 and Version 2 cartridges may be copied.

Dump

The disk dump subroutine of DCIP

- Dumps any disk sectors from any drive.
- Prints the dump on the fastest printer on the system (in the order of speed -- 1403, 1132, or Console Printer).

The address of the first sector to be dumped and the number of consecutive sectors to be dumped are specified in the console entry switches.

Each sector printout is 20 lines -- 16 four-digit hexadecimal words per line. Two sectors are printed on each page and each sector is preceded by a 3-word header. The first digit of the first header word is the drive number. The remaining three digits of the first header word show the physical sector address of the sector being dumped. The second header word is the sector address that actually appears on the sector being dumped. The third word is the logical sector address, taking into account any defective cylinders. If the user dumps a sector that is in a defective cylinder, the third word will contain the letters DEFC.

Patch

The disk patch subroutine of DCIP

- Allows the user to change the contents, word by word, of any disk sector.

- Prints the contents of the sector patched both before and after the changes are made. The fastest printer on the system that is ready is used for output.

The address of the sector to be patched and the relative address of the sector word to be changed are entered through the console entry switches.

The hexadecimal characters 0 through F and six special control characters are entered through the console keyboard.

A one-word store-buffer is reserved. This can be stored to replace the contents of any word of the specified sector. Each hexadecimal character entered causes this store-buffer to be shifted left 4 bits dropping off the most significant hex character and replacing the least significant hex character with the one just entered.

The special characters are used as follows:

- To move the relative address pointer forward or backward.
- To store the contents of the one-word store-buffer and increment the relative address pointer.
- To enter a new value in the relative address pointer.
- To terminate the patch function.
- To fill out the sector with the contents of the one-word store-buffer and terminate.

Termination causes the sector contents as modified to be written back on disk. The sector is then read back and printed.

Analyze

The disk analysis subroutine of DCIP

- Reads each logical sector 16 times and prints the address of the sector each time a read error occurs.
- Dumps the sector in error if requested.
- Checks each logical sector address for the correct value and prints the address and erroneous contents when an error is found. The correct sector address is then written on the sector.

The drive number of the cartridge to be analyzed is entered through the console entry switches.

Compare

The disk compare subroutine of DCIP

- Reads corresponding sectors of two drives and compares the contents word by word.

The drive numbers of the cartridges to be compared are entered through the console entry switches.

Operating Procedures

- With the console Mode switch set to RUN, press IMM STOP and RESET on the console.
- Place the Disk Cartridge Initialization Program in the reader wired for IPL and ready the reader.

  If the system configuration is 2501-1442, make the 1442 not-ready. (On the paper tape system, place the DCIP tape in the reader, positioning the tape so that one of the delete codes following the program name in the leader is under the read starwheels.)
- Make printer READY.
- Press PROGRAM LOAD.
- After the program is loaded, the following message is printed on the Console Printer.

  TURN ON:
  SW0 TO INITLZ
  SW1 TO COPY
  SW2 TO DUMP
  SW3 TO PATCH
  SW4 TO ANALYZE
  SW5 TO CMP
- Turn on console entry switch 0, 1, 2, 3, 4, or 5 and press PROGRAM START.

NOTES:

1. At any point in this program, an invalid entry in the console entry switches will cause the following message to be printed.

   ENTRY ERR ...RETRY

   Correct the error and press PROGRAM START to continue.
2. If a drive is not ready, the standard preoperative trap to $PRET is made. The Accumulator contains /50X0 where X is the number of the physical drive that is not ready.
3. All console entry switch settings are printed on the Console Printer as 4-digit hexadecimal numbers.

4. DCIP messages refer to console entry switches as "bit" switches.
5. If the system has two card readers, only the reader wired for IPL should be in the ready state.
6. A DCIP function can be aborted at any time by pressing keyboard INT REQ. The user is then given the option of repeating the current function or selecting a new function.

Initialization (Console Entry Switch 0 On)

- If console entry switch 0 is on, the following message is printed.

  ENTER DR NO. (BITS 12-15)
  SW0 ON FOR DM1 LABEL

Enter the physical drive number of the cartridge being initialized (in binary) in console entry switches 12-15.

- Turn console entry switch 0 off if a DM2 label is desired.
- Leave console entry switch 0 on to enter a DM1 label.
- Press PROGRAM START.
- If console entry switch 0 is on, the K.B. SELECT light on the console keyboard will light. Enter a five character alphanumeric label through the keyboard. After five characters have been entered initialization will commence.
- If console entry switch 0 is off, the following message is printed.

  ENTER CART ID

  Turn off all console entry switches and enter the cartridge ID in console entry switches 1-15 (four hexadecimal characters). A valid cartridge ID is a number between /0001 and /7FFF.

- Press PROGRAM START. The cartridge ID is printed.

  XXXX

- The cartridge is initialized. (The entire surface is cleared, disk addresses are written, and three distinct bit patterns are written and read back for checking purposes. In addition, the following message and a program for printing it is written on sector @IDAD, starting at word 271.

  THIS IS A NON-SYSTEM CARTRIDGE

  When the Monitor system is loaded to disk, this message is overlaid by the Cold Start program; therefore, an attempt to cold start a non-system cartridge will result in the above message being printed.

  One of the following messages is printed.

  NO DEF CYCLS
  or
  DEF CYCLS:
  XXXX...

If more than 3 defective sectors are printed, or if cylinder zero is defective, or if the sector address cannot be written on every sector, the cartridge cannot be used with the Monitor system and the following message is printed:

CART DEF

The last message printed is:

TURN ON SW 0 FOR MORE TESTING

- Set console entry switch 0 as desired and press PROGRAM START.
- If console entry switch 0 is off, the program returns to accept the next DCIP function.
  If console entry switch 0 is on, the following message is printed.

  ENTER REPEAT CNT (BITS 11-15)

  Enter the repetition count (max. 31) in binary in console entry switches 11-15. This will give additional opportunity to find marginal cylinders and reduce chances of disk errors later on.

- Press PROGRAM START.
  Initialization is repeated with each cylinder being checked with each pattern the number of times specified in the repetition count. When the pass is completed, the initialization complete messages are repeated, including any new defective cylinders found and the user is again given the option to repeat the initialization, or select the next DCIP function. All new cartridges must be initialized by DCIP.

Copy (Console Entry Switch 1 On)

- If console entry switch 1 is on, the following message is printed.
  ENTER:
  SOURCE DR (BITS 0-3)
  OBJECT DR (BITS 12-15)
  Enter the physical drive number of the source drive (in binary) in console entry switches 0-3. Enter the drive code of the object drive (in binary) in console entry switches 12-15.

Press PROGRAM START.
If the cartridge on either the source or object drive has not been initialized, the following message is printed.

X CART NOT INITLZED

where X is "SOURCE" or "OBJECT"
The program now returns to accept the next DCIP function and the option messages are printed.

If the object cartridge is a DM2 system cartridge, the following message is printed:

OBJ CART NOT FRESHLY INITLZED

The operator can either press the INT REQ key and return to initialize the cartridge, or press PROGRAM START and continue.

If both drives have been initialized, the contents of the source cartridge (less defective sector data and cartridge ID) is copied on the object cartridge. Word 5 of sector @IDAD of the source cartridge (zero when the cartridge is initialized) is incremented by 1 when written on the object cartridge. The copy number of the object cartridge will thus always be one more than the copy number of the source cartridge.
NOTE: When copying is complete, the program returns to select the next DCIP function and the option messages are printed. If a disk read/write error occurs, the following message is printed.

DISK ERR...TURN ON SW 0 TO RETRY

At the WAIT, the Accumulator contents will be /0001 for a read error or /0002 for a write error. The Extension will contain /XYYY where X is the drive code and YYY is the address of the sector in error.

Turn console entry switch 0 on and press PROGRAM START to rewrite or reread the sector in error.

Leave console entry switch 0 off and press PROGRAM START to ignore the error and continue. If the error is ignored, the contents of the object cartridge will reflect the last attempt to copy the sector in error.

Dump (Console Entry Switch 2 On)

- If console entry switch 2 is on, the following message is printed.

ENTER...PHYS DR NO. (BITS 0-3)
SCTR ADDR (BITS 4-15)

Enter the physical drive number of the drive containing the cartridge to be dumped in console entry switches 0-3. Enter the address of the first sector to be dumped in console entry switches 4-15 (hexadecimal, maximum /0657.

- Press PROGRAM START. The following message is printed.

ENTER NO. OF SCTRS TO DUMP

Enter the number of consecutive sectors to be dumped as a right-justified hexadecimal number in the console entry switches. The maximum amount will depend on the starting sector address.

- Press PROGRAM START.

The requested number of sectors will be dumped. When the dump is complete, the program returns to accept the next DCIP function and the option messages are printed.
NOTE: If a disk read error occurs, the following message is printed.

DISK ERR...TURN ON SW 0 TO RETRY

Turn console entry switch 0 on and press PROGRAM START to read the sector in error. If the reread is successful, the sector is printed and the dump continues.

Leave console entry switches 0 off and press PROGRAM START to ignore the error and continue. The sector in error is printed as it was last read from the disk.

Patch (Console Entry Switch 3 On)

- If console entry switch 3 is on, the following message is printed:

ENTER:
PHYS DR NO. (BITS 0-3)
SCTR ADDR (BITS 4-15)

Enter the physical drive number of the drive containing the cartridge to be patched in the console entry switches 0-3. Enter the address of the sector to be patched on console entry switches 4-15 (hexadecimal 0657 maximum)

- Press PROGRAM START.

The sector contents will be dumped. The following message is then printed:

ENTER RLTV ADDR OF SCTR WD TO CHANGE

Enter the relative address of the sector word in hexadecimal (0-13F) through the console entry switches.

NOTE: If a change in the sector address is desired the value -1 (FFFF) may be entered.

- Press PROGRAM START.

The KEYBOARD SELECT indicator will light and the program will wait.

NOTE: The relative address pointer is displayed in the Extension each time the program waits for Keyboard input.

- Enter through the console keyboard the four hexadecimal characters which comprise the word to be stored. The characters typed will be printed on the console printer.

NOTE: If an error is made, entering the correct four characters will replace the ones in error.

- Any of the keys for the six special control characters can be pressed with the following result:

EOF	- The last four hexadecimal characters input will be stored at the relative address displayed in the Extension.
>	- The relative pointer is incremented one word.
<	- The relative pointer is decremented one word. See note below.
*	- Patching is terminated. The sector as modified is written on the disk. The sector is read back and dumped.
.	- All the remaining words of the sector from the address pointed to by the relative address displayed in the extension to relative address /013F will be filled with the last four hex characters input. Patching is terminated.
R	- The message will be printed again requesting the relative address of the sector word to change. In this way the pointer can be changed without stepping it one word at a time.

If an invalid character is pressed, the following message will be printed:

ENTRY ERR...RETRY

Enter the correct character to continue.

NOTE: The < will not decrement the pointer beyond the first data word (relative address =0). The sector address pointer (FFFF) can only be entered through the console entry switches.

Analyze (Console Entry Switch 4 On)

- If console entry switch 4 is on, the following message is printed:

ENTER DR NO. (BITS 12-15)

Enter the physical drive number of the drive containing the cartridge to be analyzed in the console entry switches 12-15.

- Press PROGRAM START.

The program begins reading each sector 16 times. If a disk read error occurs, the following message is printed:

DISK READ ERR ON SCTR nnnn
TURN ON SW 0 TO DUMP

Turn on the bit 0 switch of the console entry switches if a dump of the sector is desired.

- Press PROGRAM START.

If an erroneous sector address is read, the following message is printed:

ADDR ERR ON SCTR nnnn
BAD SCTR ADDR WAS nnnn

The correct address will be written on the disk sector and disk analysis will continue.

Compare (Console Entry Switch 5 On)

- If console entry switch 5 is on, the following message is printed:

ENTER:
SOURCE DR (BITS 0-3)
OBJECT DR (BITS 12-15)

Enter the drive number of the cartridges to be compared.

- Press PROGRAM START.

The program compares each logical sector of the source drive with its counterpart on the object drive. If the contents of two corresponding sectors does not compare, the following message is printed:

CMP ERR ON SCTRS xxxx yyyy

When compare is finished, the program returns to accept the next DCIP function. If the function is not another COMPARE, the following message is printed:

CMP OPTION USED...RELOAD DCIP

This is required because part of the functioning program is overlaid by the additional disk buffer used by compare.

PAPER TAPE REPRODUCING PROGRAM

This program, available only with the paper tape system, is a self-loading paper tape strip that reproduces paper tapes. The program reads a character and punches it with no intermediate conversion.

Operating Procedure

- Place the paper tape reproducing program tape in the paper tape reader, positioning the tape so that one of the delete codes beyond the ID in the leader is beneath the read starwheels.

- With the console Mode switch set to RUN, press IMM STOP, RESET, and PROGRAM LOAD on the console. The reproducing program is read in and WAITs with /1111 in the Accumulator.

- Remove the reproducing program tape and place the tape to be reproduced in the reader. Place blank tape in the tape punch unit and produce several inches of delete code leader by first pressing down and holding the DELETE key. Then press the FEED key and hold until a leader of sufficient length has been punched. Release the FEED key before releasing the DELETE key.

- Press PROGRAM START to begin the tape reproducing operation. The program continues to operate until the paper tape reader goes not-ready, indicating that there is no more tape to be read. The tape reproducing routine then WAITs with /2222 in the Accumulator. If the paper tape punch is not-ready, the tape reproducing program WAITs with /3333 in the Accumulator. To restart, ready the paper tape punch, and press PROGRAM START. An unlimited number of tapes can be reproduced by this program. Be sure to create a trailer (and leader) of delete codes between the output tapes if the tapes are to be separated.

NOTE: If the PROGRAM STOP key is pressed while the program is in operation, the program WAITs with /4444 in the Accumulator. Press PROGRAM START to continue.

STAND-ALONE PAPER TAPE UTILITY PROGRAM (PTUTL)

This program, also included as an executable program in the System Library, is a self-loading paper tape utility program that allows the user to enter records from the 1134 Paper Tape Reader or the Keyboard. Program output is to the 1055 Paper Tape Punch and/or the Console Printer.

Operating Procedures

- Place the PTUTL tape in the paper tape reader so that one of the delete codes beyond the program ID is under the read starwheels.

- With the console mode switch set to RUN, press IMM STOP, RESET, and PROGRAM LOAD on the console.

- PTUTL is read in and the system WAITs with/1111 in the Accumulator.

- For complete operating instructions for PTUTL, see Paper Tape Utility (PTUTL) in the System Library.

REMOTE JOB ENTRY PROGRAM

The Remote Job Entry (RJE) feature of OS/360 extends to users the ability to introduce jobs into the OS/360 job input stream from remotely located terminals via communication lines. RJE includes a unique Job Entry Control Language which provides the additional flexibility and control required for remote entry. For a general description of RJE and the Job Entry Control Language see IBM System/360 Operating System, Remote Job Entry (Form C30-2006).

This section provides information for operators and programmers using an 1130 as a remote work station in an RJE environment, and describes machine and device requirements, input and output at the work station, communication considerations, operating procedures, operator messages, user-exit interface and generation and loading of the work station program.

MACHINE AND DEVICE REQUIREMENTS

The RJE program for an 1130 work station requires at least an 1131 Model 2B, a card reader, a card punch and a line printer (with 120 character print line). The 1130 System must be connected to a 600-2400 bit-per-second line via a Synchronous Communications Adapter in binary mode. There is an optional compress-expand feature, i.e. elimination of blanks on the line. This feature requires 16 K words of core storage if the 1132 printer is used.

A user-written subroutine may specify output on any available output device. An 1130 system with 16K words of core storage is required to support a user-written subroutine. Data directed to the user-exit is stored on disk, if not user-written subroutine is present, and can be processed by another user program after RJE processing is terminated.

INPUT AT THE WORK STATION

Input is accepted from the card reader, the Keyboard and from one or more disk storage units.

Job entries (OS/360 jobs with or without JED statements) and work station commands are acceptable input from the card reader. No JECL statements are sequence checked, but the first statement at work station startup must be an RJSTART command submitted from the card reader.

The only valid input from the Keyboard consists of work station commands. Input is accepted from the the Keyboard between job entries (only in a point-to-point line configuration) from the card reader (or disk) when the operator has indicated that he has such input to submit. The 1130 RJE Work Station Program checks this input for the JECL identifier (..followed by at least one blank) only.

Input is also accepted from one or more disk storage units. A special 1130 RJE control card (see JECL for the 1130 Work Station) is defined to control this function. This control card may be placed in the card input stream or on disk. It contains information allowing the RJE program to read input alternately from the card reader and from the disk. Data to be read from disk must be stored there prior to RJE processing by the user. This data must be stored in 80-character records in 8-bit packed code (EBCDIC) format (eight records per disk sector), in consecutive sectors. After reading this input to end of file (see JECL for the 1130 Work Station section for a description of the various end of file indications), the RJE program resumes reading from the card reader.

NOTE 1: If a user is logged on at the card reader (or disk) and another LOGON command is submitted from the Keyboard, all pending input for that user at the card reader and/or from disk will be submitted under the new LOGON user ID. To prevent this, the last LOGON command, which was submitted from the card reader or disk must be submitted as the last command from the Keyboard.

NOTE 2: Although it is possible to submit work station commands from disk, it is recommended that only job entries and OS/360 input data sets are placed on disk, in order to simplify work station operation.

OUTPUT TO THE WORK STATION

Output to the work station consists of job output and messages. Job output, consisting of SYSOUT data sets created by the job, is directed to the printer, the punch, or a user-exit subroutine. Each

job output data set is directed to the device associated with the SYSOUT class specified in the DD statement for the output data set. RJE system messages are directed to the Console Printer or the line printer.

Carriage control for printer output may be specified by a control character as the first byte of each record. Either machine code or ASA control characters are allowed. The output is single-spaced with a skip to channel 1 when channel 12 is sensed in the carriage tape and no control characters are specified or the control characters are not recognized by the equipment.

Stacker select for punched output, if available, may be specified by a control character as the first byte of each record. Either machine code or ASA control characters are allowed. Stacker 1 is selected if no control characters are specified or if the control characters are not recognized by the equipment.

The 1130 RJE Work Station Program includes a user-exit subroutine which accepts data sets directed to it and writes them on disk in an area reserved by the user.

This subroutine may be replaced by another user written subroutine to process data directed to the user-exit and to write output to any available device (see User Exit Interface for a more detailed description).

If no user-written subroutine is present, the RJE program writes user-exit data sets consecutively on disk, each data set beginning at a disk sector boundary. However, if the RJE program is reloaded, data sets previously written on disk are unprotected and may be destroyed since any additional user-exit data sets are written beginning at the first sector of the reserved area. Information is displayed to the operator for each data set written on the disk (see 1130 RJE Messages).

The primary output device for messages is the Console Printer. The secondary device is the line printer. The operator selects the line printer as the message device by turning on Console Entry Switch 0.

NOTE: Data directed to disk may be referenced later by a .. DATA command. To be able to do this, the user must define his data set as fixed blocked or unblocked with a logical record length of 80 bytes and no control characters.

COMMUNICATION CONSIDERATIONS

The 1130 RJE Work Station Program provides the standard RJE communications interface to the RJE communications network using SCAT2 and SCAT3 binary synchronous communications subroutines to provide the following capabilities:

1. Point-to-point contention operation on leased lines.
2. Point-to-point operation on switched lines.
3. Multipoint operation with the 1130 system as slave station.

All data transmissions between the central processor and a remote 1130 are in EBCDIC transparent mode except headings, which are transmitted in normal mode. Communication with the central system proceeds in three modes: monitor, receive, and transmit.

Monitor mode is entered from either transmit or receive mode. In monitor mode, the work station waits for input from the line, card reader, or Keyboard.

Receive mode is entered when output is available for the work station. In receive mode, the program reads output from the line until it receives an end-of-data indication from the central system or until the operator discontinues the output. The program then enters monitor mode.

Transmit mode is entered at work station startup and when input is available at the work station. The work station program writes to the line in transmit mode. It continues writing to the line until it has encountered a logical end-of-file (..null command or RJEND) in the input stream.

If monitor mode is entered from transmit mode with a logical end-of-file indication (caused by a ..null command, transmit mode is not entered again until operator intervention indicates that more input is available.

COMMUNICATION CONSIDERATIONS FOR SWITCHED LINES

If a switched communication line is inactive for a period of approximately 21 seconds, the central RJE program disconnects the line. This can be caused by three situations:

1. The remote RJE program cannot maintain the connection when an error on an output device is not corrected within the specified time.
2. The remote RJE program cannot maintain the connection when a user-written subroutine fails to return control within the specified time.

3. The remote RJE program cannot maintain the connection when it is waiting for an operator response. When requested to reply to some RJE messages, the operator must enter his response within the specified time.

NOTE: The operator will have approximately three minutes to reply to some RJE messages. See Operator Messages for detailed information.

OPERATING PROCEDURES

WORK STATION STARTUP

To start RJE operation, the operator loads the 1130 RJE Work Station Program by using the program name RJE in the XEQ monitor control record. This program uses the information saved on disk by the generation program (see 'Generation of the 1130 RJE Work Station Program' for a description of this program) and information from the Disk Monitor System which specifies principal I/O device and principal print device in order to load the mainline program and the subprograms necessary to perform the RJE functions corresponding to the user's configuration.

NOTE: The Console Printer cannot be the principal print device.

The RJSTART command must be the first data card in the card reader. A missing RJSTART command results in an error message to the operator. The operator then places a correct RJSTART command in the card reader and presses the PROGRAM START key to continue. If the work station is connected to the central system over a switched line a message is now given to the operator, telling him to call the central system.

The RJSTART command may be followed either by input to be sent to the central system or by an end-of-file indicator (see The Null Command). When contact is made with the central system, the RJSTART command and all other commands, if any, before the first job entry (the OS/360 job with or without the JED card) or before the end-of-file indicator are transmitted. The work station is logically attached to the RJE system with the acknowledgement of the RJSTART command. The operator receives all pending messages and immediate job output directed to users at the work station. All pending input, if any, is transmitted or the work station program monitors for output from the central system. The sequence of events is system dependent.

THE NULL COMMAND

The null command is provided for the 1130 work station to indicate end of file on the card reader. It must be the last card of an input stream. When this command is read, the card reader is effectively closed, but communication is still maintained with the central system.

Operator intervention is required to resume input from the card reader after the null command has been read. The null command is coded with the identifying characters (..) in columns 1 and 2 and all remaining columns blank.

KEYBOARD PROCEDURES

There are four control functions initiated by the operator from the Keyboard: indicating card reader input, indicating Keyboard input, discontinuing output and initiating an abnormal closedown of the RJE program. These functions are initiated by the operator first pressing the PROGRAM STOP and then the PROGRAM START keys on the console. When the message 'J90 OCR=' (Operator Communication Request) appears, the operator enters the appropriate reply to initiate the function he desires (see Operator Messages later).

If the operator has indicated Keyboard input, a message 'J93 PROCEED' will be displayed and the Keyboard select light is turned on, at the time when the program can service Keyboard input. The operator then enters the desired commands with an EOF at the end of each command. After entering the last command, an extra EOF must be entered to indicate end of input. The last EOF must not be entered until the Keyboard select light is on.

An abnormal closedown is initiated by replying with a T to the 'J90 OCR=' message. This reply causes the work station program to be terminated and the contents of core storage to be printed out.

The central system notes an error condition and logically detaches and disconnects the work station, if it is connected over a switched line. The work station is also logically detached from the central system on a leased or multipoint line, if a line operation is in progress when the operator requests the termination; and also when the central system tries to contact the work station if the line was idle when the request was made and the program has not been reloaded.

NOTE 1: If the Keyboard procedure is used when the Console Printer is already in use, the message is not printed. However, the PROGRAM START key must be pressed in order to continue processing.

NOTE 2: The INT REQ key may not be used when the RJE program is loaded, because certain information in the Skeleton Supervisor, modified by the RJE program, will then not be restored and may cause the Disk Monitor System to function improperly.

DISCONTINUING OUTPUT

Job output can be discontinued by operator intervention. The operator used the Keyboard procedure to initiate the request with a D to discontinue output.

Output is also discontinued by the 1130 RJE Work Station Program when no user-written subroutine is present for output directed to user-exit and when one of the following three errors occurs:

1. No area is reserved for user-exit output.
2. The reserved area is exhausted.
3. An unrecoverable disk write error occurs.

These errors are indicated to the operator in an error message. To correct problems 1 and 2, the operator should terminate RJE by submitting an RJEND command (after all pending input has been transmitted) and then reserve an area on disk by executing the RJE00 program (see Generation of the 1130 RJE Work Station Program section). The RJE program should then be reloaded and the output should be discontinued immediately by the operator and a CONTINUE command with the BEGIN operand should be submitted. Otherwise, data will be lost.

The same CONTINUE command should be used if the third error occurs. The data set will then be written again, starting at a new sector.

In general, once output is discontinued, no other output is transmitted to the work station until the disposition of the discontinued output is specified by the CONTINUE command.

CONTINUING OUTPUT

Disposition of discontinued output is specified with the CONTINUE command. Output is discontinued if the following conditions occur:

1. The remote operator requests discontinuation.
2. A change in form number is found at the central system.
3. The remote program requests discontinuation.
4. An irrecoverable error occurs during an output operation.

If conditions one, two or three occur, the disposition of the output is specified with the CONTINUE command. Condition four requires error recovery procedures.

ERROR RECOVERY PROCEDURES

At an 1130 work station, facilities are provided to recover from both communication errors and local device errors. Operator intervention may be necessary to correct the condition causing the error. If the error cannot be corrected within the allowed time, the central system logically detaches the work station from the RJE system. In addition, if the work station is connected over a switched line, the central system breaks the connection.

In the case of a local I/O device error a message is always issued except for a forms check on the Console Printer. This error causes the forms check light to go on, and the operator tells the system to try again by turning on console entry switch 1. The communications on the line are maintained only if the error is corrected within approximately 21 seconds.

An error on an I/O device other than Keyboard is always followed by a message describing what type of error has occurred. The explanations for the messages, and the actions taken by the program after the operator's reply are described in the 1130 RJE Error Messages section.

Irrecoverable communication errors result when communication is lost with the central system because of either line errors or a failure at the central system. In either case, the work station is logically detached by the central system and restart procedures are necessary. The response received when restart procedures are executed indicates whether the error was due to a line error or to a failure at the central system.

RESTART PROCEDURES

Restart procedures involve regaining communication with the central system and submitting an RJSTART command. The operator initiates the restart procedure by replying with an A to the line error message (see 1130 RJE Error Messages).

If the error occurs during an output operation,

output automatically resumes either where it was interrupted (after a line error) or at the beginning of the job (after a failure at the central system).

If output was written to disk at the time of a line error and if it was not a central system failure, the operator should discontinue the output and submit a CONTINUE command with the BEGIN operand.

If the output was written to the punch or the printer at the time of a line error and if it was not a failure at the central system, a duplication of the last transmission block may occur when the program is restarted. The printer will skip to a new page when RJE is restarted if the data set being printed is without control characters.

If the error occurs during an input operation, all unacknowledged input must be resubmitted. Furthermore, a line error in the middle of a job implies that the whole job must be resubmitted from the beginning. Before the job can be transmitted again with the same job name, the old job, that was partially sent to the central system, must be deleted. This is sometimes done automatically, but if not, the job must be deleted by the operator.

NOTE: The work station restart procedure after a central system failure is similar to the restart procedure after an irrecoverable line error. The primary difference is that after a system failure, an in-process output data set is written from the beginning rather than from the last valid block.

ERROR STATISTICS

Error statistics are accumulated during an RJE run. The operator tells the program that he wants these statistics printed out by turning on console entry switch 2 before the RJE run is terminated. It is also possible to get a printout of the error statistics accumulated during the last RJE run by executing a program called RJSTA that belongs to the RJE package.

CONSOLE ENTRY SWITCHES

Three console entry switches are used by the RJE Work Station Program.

0 - off - RJE messages from the central system will go to the Console Printer
on - RJE messages from the central system will go to the line printer.

1 - When the Console Printer becomes not ready, the operation will not be retried unless this switch is on.

2 - If on, the error statistics accumulated by SCAT2 or SCAT3 will be printed out on the Console Printer at the end of the RJE run.

OPERATOR MESSAGES

The first digit of the messages has the following meaning:

0 - Error in RJE00

1 - Error in the initializing part of RJE

2 - Error during the processing of the RJE program, that does not require an operator reply.

5 - Error during the processing of the RJE program, that requires a reply from the operator.

9 - Non-error message.

1130 RJE ERROR MESSAGES

J01 INVALID CARD

Explanation: This message is issued during work station program generation when the control card containing the work station information is invalid or contains invalid information (see Generation of the 1130 RJE Work Station Program).

System Action: The program exits to the Disk Monitor Supervisor.

Operator Response: The operator must reload the generation program and enter a valid data card.

J10 INVALID PRINTER

Explanation: Information from the Disk Monitor System indicates that the principal print device is not an 1132 Printer or a 1403 Printer.

System Action: The system exits to the Disk Monitor Supervisor.

Operator Response: The operator may reload the RJE program after performing a system reload and may specify either an 1132 Printer or a 1403 Printer as the principal print device (see System Reload for information on how to reload the system).

J11 INVALID READER

Explanation: Information from the Disk Monitor System indicates that the principal I/O device for the system is not a 1442 card reader or a 2501 card reader.

System Action: The system exits to the Disk Monitor Supervisor.

Operator Response: The operator may reload the RJE program after performing a system reload and may specify either a 1442 Card Reader or a 2501 Card Reader as the principal I/O device (see System Reload for information on how to reload the system).

J12 LOGICAL DRIVE X NOT IN SYSTEM

Explanation: The area on disk reserved for user-exit data is on a logical disk drive that is not present in this RJE run. The requested logical drive replaces X in the message.

System Action: The system exits to the Disk Monitor Supervisor.

Operator Response: The operator may reload the RJE program after having changed the user-exit parameters or after having introduced the requested logical disk drive.

J14 DISK ERROR OCR=

Explanation: A permanent error has been encountered while attempting to read data from disk during the initialization part of the RJE program.

System Action: The system continues according to the operator response.

Operator Response: The operator must enter one of the following codes:

T - Exit to the Disk Monitor Supervisor requesting a terminating dump of the contents of core storage on the printer.

X - Exit to the Disk Monitor Supervisor, without writing the contents of core storage on the printer.

J20 RJSTART MISSING

Explanation: The requirement for an RJSTART command has not been satisfied.

System Action: The system waits for operator action.

Operator Response: The operator must enter an RJSTART command through the card reader and press the PROGRAM START Key, in order to resume processing.

J21 .. DATA INVALID

Explanation: A .. DATA statement contains invalid parameters.

System Action: The system waits for operator intervention. The line is monitored for output from the central system.

Operator Response: To continue RJE processing, the operator must use the Operator Communication Request facility (see message J90 OCR=).

Note: This message is also issued if the requested logical disk drive is not present.

J22 INVALID INPUT

Explanation: The input entered from the console-keyboard does not start with the JECL identifier (..) followed by at least one blank.

System Action: The system waits for more input from the Keyboard.

Operator Response: The operator must enter a work station command or press EOF.

J23 INPUT ABORTED BY CENTRAL

Explanation: The central system has aborted input from the work station and will send a message explaining why the input was aborted (For details on messages received, see Messages Sent to Work Stations in IBM System/360 Operating System, Remote Job Entry Form C30-2006).

System Action: The system waits for input from the line.

Operator Response: When the message is received from the central system, the operator inspects the message and takes the indicated action. To resume input the operator must follow the procedures described under Keyboard Procedures.

J51 LINE ERROR OCR=

Explanation: An irrecoverable error has been encountered while reading or writing on the communication line, or the line cannot be opened.

System Action: The RJE program closes the communication line, if it is open, and waits for an operator response.

Operator Response: The operator must reply by entering one of the following codes from the Keyboard:

A - Input is available at the card reader. If this option is selected, the first card in the card reader must be an RJSTART command. On a switched line, the line has to be disconnected before the restart is tried. If this is not done automatically by the work station program, it has to be done by the operator. He has to dial again when the message J91 ESTABLISH LINE CONNECTION is issued.

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

X - Exit to the Disk Monitor Supervisor, without writing the contents of core storage on the printer.

J52 DISK ERROR INPUT OCR=

Explanation: A permanent error has been encountered while attempting to read input from disk. This message is issued only if a user's disk input is being read at the time the error occurs.

System Action: Reading of the input data file(s) and card reader input is discontinued. Any available output from the central system is accepted after the operator response has been entered. The system continues according to the operator's response.

Operator Response: The operator must enter one of the following codes. The response must be entered within approximately 3 minutes on a switched line.

A - Input is available at the card reader.

B - Commands are to be read from the console-keyboard.

C - Available output is accepted. (Any pending keyboard input is processed first.)

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

NOTE: A user may have to resubmit a job that has been only partially entered, but must precede this by either obtaining the output of, or deleting, the job in question.

J53 DISK ERROR OUTPUT OCR=

Explanation: An unrecoverable error has been encountered while attempting to write data on disk. This message is issued only if data is being written on disk by the IBM-supplied user-exit routine.

System Action: Output from the central system is discontinued. The disposition of the output is specified by use of the CONTINUE command. The system continues as directed by the operator response.

Operator Response: The operator must enter one of the following codes. The response must be entered within approximately 3 minutes on a switched line.

A - Input is available at the card reader. (Any pending keyboard and disk input is processed first.)

B - Commands are to be read from the console-keyboard.

C - Any pending input (keyboard, disk or card) is processed. If no pending input exits, the system maintains the line operations.

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

J54 DISK ERROR OCR=

Explanation: An unrecoverable error has been encountered while attempting to read RJE constants or error messages from disk. If this message appears, an RJE error message that indicates the original error may not appear.

System Action: The system continues according to the operator response.

Operator Response: The operator must enter one of the following codes:

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

X - Exit to the Disk Monitor Supervisor without writing the contents of core storage on the printer.

J55 END OF DISK AREA OCR=

Explanation: The user has failed to reserve space or has reserved too little space on disk for user-exit output data sets.

System Action: Output from the central system is discontinued. The system continues as directed by the operator response.

Operator Response: The operator must enter one of the following codes. The response must be entered within approximately 3 minutes on a switched line.

A - Input is available at the card reader (Any pending keyboard and disk input is processed first.)

B - Commands are to be read from the Keyboard.

C - Any pending input (keyboard, disk or card) is processed. If no pending input exists, the system maintains the line operations.

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

J56 CARD READER ERROR OCR=

Explanation: An error has occurred on the card reader which requires operator intervention.

System Action: The system waits for the operator reply.

Operator Response: The operator must enter one of the following codes. The response must be entered within approximately 3 minutes on a switched line.

A - The operator has corrected the problem and the program will resume card reader input.

E - The operator could not correct the problem. The program assumes reading an end-of-file (.. null card) indication to close the card reader.

J57 CARD PUNCH ERROR OCR=

Explanation: An error has occurred on the card punch which requires operator intervention.

System Action: The system waits for the operator response.

Operator Response: The operator must enter one of the following codes. This response must be entered within approximately 3 minutes on a switched line.

D - The operator could not correct the problem. Output from the central system is discontinued and a .. CONTINUE command has to be transmitted to resume output.

P - The operator has corrected the problem and the program will resume card punch output.

J58 PRINTER ERROR OCR=

Explanation: An error has occurred on the printer which requires operator intervention.

System Action: The system waits for the operator response.

1130 Disk Data File Conversion Program

The following information is designed to assist in understanding the program flowcharts by presenting an overall view of the purpose of each major part of the program.

FLOWCHARTS:

CHART A
CHART B
CHART C
CHART D
CHART E
CHART F
CHART G

PROGRAM NAME: DFCNV

GENERAL PROGRAM DESCRIPTION: The program converts one 1130 FORTRAN and/or Commercial Subroutine Package (1130-SE-25X) disk data file to one 1130 RPG disk data file. FORTRAN files created using logical unit number 10 cannot be converted by DFCNV. Converted files cannot be processed as ISAM files. The program accepts all FORTRAN and Commercial Subroutine Package (CSP) disk data formats and a two-word integer format (see Appendix E). The input data may be a disk data file or the corresponding cards in card data format. All printing is performed on the principal printer. The subroutine DISK1 is used to perform all disk I/O operations.

PART 1:

ENTRY POINT: FC000 (CHART A)

The system device subroutines for the principal input and print devices and input data conversion are read into core and pertinent interrupt pointers are set.
o The File Description card (D in column 72) is read and printed, and its fields are diagnosed for errors.
o LET/FLET searches are performed for input (if disk file input specified) and output files and calculated file sizes are checked against actual file sizes.

INTERNAL SUBROUTINES:

FC015 - The card read and/or print function in this section of the program is not specifically subroutinized, but it is the general card input function for the entire program.
CONVT- Subroutine which converts a right-justified EBCDIC coded decimal field of variable length to a one word binary value and advances the field pointer beyond the field just converted. It accounts for leading blanks in a File Description (D) card field and causes immediate program termination when a D-card field error is detected.
SEARC- Subroutine which checks the file name referenced by the field pointer for validity, packs the file name, adds the disk data format indicator to the packed file name and performs the LET/FLET search for the file.

ERRORS DETECTED: The errors detected in part 1 are F01, F02, F03, F06 and F08 (see Appendix F).

PART 2:

ENTRY POINT: FC016 (CHARTS B and C)

o The Field Specification cards (S in column 72) are read and printed, each field specification is diagnosed for errors and the specification information is compressed and saved.
o If it is present, the Commercial Subroutine Pack age A3 format translation table (A in column 72) is read and printed and the 40 translation characters are saved.
o The end-of-file card (/ * in column 1 and 2) is read and printed.

Note: A general flowchart of the compress/save operation described above has been provided in Chart C although this operation is in fact specific to field type. See Appendix G for a description of each field type compression.

INTERNAL SUBROUTINE:

CONVT: Subroutine is described in part 1.

ERRORS DETECTED: The errors detected in part 2 are F04, F06 and F07. It is noted that only one F04 message is printed for each field specification in error although more than one error may occur within a field specification.

PART 3:

ENTRY POINT: FC026 (CHART D)

- Final error checking is performed and program

Operator Response: The operator must enter one of the following codes. This response must be entered within approximately 3 minutes on a switched line.

D - The operator could not correct the problem. Output from the central system is discontinued and a .. CONTINUE command has to be transmitted to resume output.

P - The operator has corrected the problem and the program will resume printer output.

J59 PREOPERATIVE ERROR CODE XXXX OCR=

Explanation: A preoperative error has occurred in the user-exit subroutine, or a logical disk drive has been referenced that was present during the job processing preceding the loading of the work station program, but that has later become not ready. The preoperative error code as defined in Appendix A replaces XXXX.

System Action: The system waits for the operator response.

Operator Response: The operator must enter one of the following codes. This response must be entered within approximately 21 seconds on a switched line.

C - The operator has corrected the problem and the program will retry the operation.

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of main storage on the printer.

X - Exit to the Disk Monitor Supervisor without writing the contents of main storage on the printer.

1130 RJE MESSAGES

J90 OCR=

Explanation: The RJE program is ready to service an operator request. The operator indicates that he wants to communicate with the 1130 RJE Work Station Program by pressing the PROGRAM STOP key and then the PROGRAM START key (see Keyboard Procedures).

System Action: The system waits for the reply.

Operator Response: The operator enters one of the following codes. The response must be entered within approximately 21 seconds for switched lines and also within the same time limit on a leased or multipoint line, if a line operation is in progress.

A - Input is available at the card reader.

B - Commands are to be submitted from the Keyboard.

D - Discontinue receiving output.

N - Ignore the request

T - Exit to the Disk Monitor Supervisor, requesting a terminating dump of the contents of core storage on the printer.

J91 ESTABLISH LINE CONNECTION

Explanation: This message is displayed only at an 1130 work station on a switched line. The operator has to establish a connection with the central system.

System Action: The system waits for a completed connection.

Operator Response: The operator must perform the dial-up procedure to establish the connection with the central system (see 'Operating Procedures' in the IBM 1130 Synchronous Communications Adapter Subroutines, Form C26-3706).

J92 DATA rrrr0c0f TO DISK AT xaaa,bbbb

Explanation: This message is received only when no user-written subroutine is present. The RJE program is writing a data set to disk. The message codes have the following meaning:

rrrr - The logical record length in hexadecimal for fixed blocked or unblocked records.

c - The type of control characters used, where c may have the following values:

0 - No control characters are used.
1 - System/360 machine code control characters are used.
2 - ASA control characters are used.

f - The OS/360 record format where f may have the following values:

1 - Fixed unblocked records
2 - Fixed blocked records
3 - Variable unblocked records
4 - Variable blocked records
5 - Undefined records

x - The logical disk drive number

aaa - The starting sector address of the data set in hexadecimal.

bbbb - The length of the data set in disk blocks where there are 40 packed EBCDIC characters per block (16 disk blocks per sector). The last block may not be filled.

System Action: The user-exit data set is written on disk. The disk block length information part of the message is written when the data set is completed; therefore, if a line error or a disk error occurs before the whole data set is received, this portion of the message remains blank.

Operator Response: None

J93 PROCEED

Explanation: This message is displayed as a result of a B reply to a J90 OCR= message. The work station is ready to receive commands from the Keyboard.

System Action: The Keyboard select light is turned on and the program waits for input from the Keyboard.

Operator Response: The operator enters the desired commands with an EOF after each command. After entering the last command, he enters a further EOF to indicate that he has finished using the Keyboard. On a switched line, the operator has approximately three minutes to enter each command.

J94 PUNCHED OUTPUT

Explanation: A SYSOUT data set is to be punched on a 1442 model 6 or model 7 card read punch unit, which is also used to read card input, and a non-blank card is at the punch station.

System Action: The system waits for operator action.

Operator Response: The operator may load blank cards in the punch and then press any character key or the space bar to resume processing. If he wants the output to be punched in the prepunched cards, he simply presses any character key or the space bar as described above.

The operator must take action within approximately 3 minutes to maintain line communication. If this time limit is exceeded a line error will occur. The RJE program is restarted according to the description under J51 LINE ERROR OCR=. The punched output will be received if an RJSTART command, a null statement and the cards to be punched are placed in the card reader and the operator then replies A to the line error message.

NOTE: If punched output is to be sent to a 1442 Card Read/Punch, which is also used for reading, all punched output should be specified as deferred.

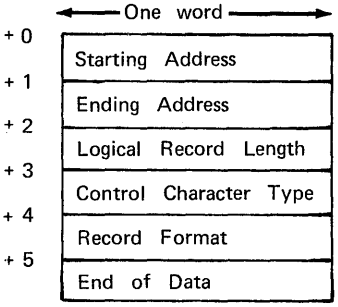
MESSAGES SENT TO WORK STATIONS

For a detailed description of all messages sent to an 1130 work station from the central RJE system, see the Messages Sent to Work Stations section in IBM System/360 Operating System, Remote Job Entry (Form C30-2006).

USER EXIT INTERFACE

The RJE program passes physical records to the user-writer output subroutine. The user's subroutine has to save and restore index register 1 and 3 for the RJE program. The user must name the subroutine entry point UEXIT and must store this routine in the User Area (after deleting the resident module with the same name). In the RJE generation program the parameter UEXIT=USER should be specified.

The user-exit subroutine gets control when output becomes available for it. Upon entry, the return address is stored in the first word of the subroutine. Index register 1 contains the address of a parameter list describing the output passed to the subroutine. This parameter list is aligned on an even word boundary. The format of this list is:



Starting Address: The address of the block received from the central system. This address has the following format. The 15 leftmost bits contain the core storage address and the rightmost bit gives the halfword, where 0 means left and 1 means right.

Ending Address: The ending address+1 of the block received from the central system. The ending address is given in the same format as the starting address above.

Logical Record Length: The length of logical records when fixed length records are passed. If variable or undefined records are passed, this word is zero.

Control Character Type

The type of control characters being used.

0 - no control characters
1 - System/360 machine code
2 - ASA code

Record Format: The code indicates the type of record..

1 - Fixed unblocked
2 - Fixed blocked
3 - Variable unblocked
4 - Variable blocked
5 - Undefined

End of Data: If zero, indicates end of data.

The characters are packed two characters per 1130 word. The blocks start on a word boundary, but they end in the middle of a word if they contain an odd number of characters.

The user-written subroutine must use the same I/O subroutines as the 1130 RJE program for I/O devices. See following table.

Device	I/O Subroutine
1132 Printer	PRNT2
1403 Printer	PRNT3
1442-6, -7 Card Read/Punch	CARD1
2501 Card Reader	READ1
1442-5 Card Punch	PNCH1
Keyboard	TYPE0
Disk	DISKZ

NOTE: The user-written routine must return control to the RJE program within approximately 21 seconds, in order to maintain the communications.

JECL FOR THE 1130 WORK STATION

JECL used for the 1130 work station is the same as that described under Job Entry Control Language in IBM System/360 Operating System, Remote Job Entry (Form C30-2006), with one addition. The additional command allows the user to alternate the source of his input between disk input and card input.

The format of this command is:

ID	Operation	Operand
..	DATA	DMS {,C / ,D,xaaa [,bbbb]}

DMS - identifies the card as an 1130 JECL command

.. - is the JECL identifier and must be in columns one and two.

DATA - must be preceded and followed by at least one blank

C - indicates that input follows on cards.

D - indicates that input follows on disk:

- x - is the logical disk drive number.
- aaa - is the disk sector address (hexadecimal).
- bbbb - is a hexadecimal number specifying the length of the disk data file in blocks where there are two blocks per 80-character record (16 blocks per sector).

If D is specified, the logical disk drive number and the sector address are required, but the block count is optional. When the block count is not specified, the user must indicate the end of data on disk by using a .. DATA command to transfer reading of data either to the card reader or to another disk area. The optional block count for disk data causes the RJE program to read data from disk until the specified number of blocks has been read, unless the end-of-file indicators are encountered first. If the RJE program reads the specified number of blocks without detecting end of disk data, reading from disk terminates and reading continues from the card reader.

Data on disk must start at the beginning of a sector and continue on consecutive sectors if necessary. Each sector must contain eight 80-character records in 8-bit code, except the last sector, which may be less than 320 words.

The .. DATA command is not recognized between a // DD DATA statement and the corresponding /* in an OS/360 job.

NOTE 1: Restart problems may occur, if jobs are chained on disk, i.e., referenced by only one .. DATA command from the card reader, and a line error occurs which requires the work station to resubmit the RJSTART command and all unacknowledged input. To avoid these problems, each job should be referenced by a .. DATA command from the card reader.

NOTE 2: The definition of the cartridges to be used during Remote Job Entry must be specified in the JOB monitor control record. The logical drive number as specified in the JOB record must be used in the .. DATA command.

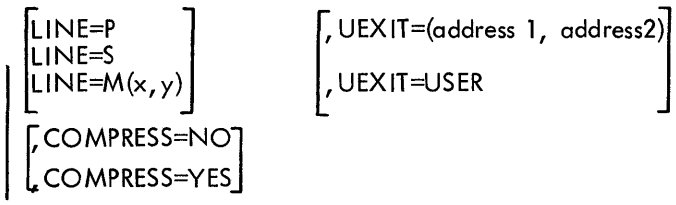
END OF FILE INDICATORS

The end of file indicator on disk is the .. DATA command, which passes the reading to another disk file or to the card reader. The end of file indicators on the card reader are the null command and the .. RJEND command.

NOTE: The null command and the .. RJEND command have the same effect if they are read from disk as if they are being read from the card reader, i.e., reading is stopped both from the card reader and from the disk.

GENERATION OF THE 1130 RJE WORK STATION PROGRAM

The 1130 RJE Work Station Program operates under the supervision of the 1130 Disk Monitor System Version 2. The user stores the RJE package in the User Area using the Disk Utility Program (DUP). The user can then describe his work station configuration by executing a program named RJE00. This program reads one data card, supplied by the user, which can contain the following parameters.



LINE=P - Specifies that the work station is connected over a point-to-point leased line.

LINE=S - Specifies that the work station is connected over a point-to-point switched line.

LINE=M(x,y) - Specifies that the work station is connected over a multipoint line where:

x - is the polling character.
y - is the selection character.

UEXIT-(address 1, address 2)

address 1 - is the starting address on disk reserved for storing data directed to the user-exit.

address 2 - is the ending address of the area reserved on disk for storing data directed to the user-exit.

The addresses must be in the form xaaa where:

x - is the logical disk drive number (from 0 to 4).

aaa - is the sector address.

The area specified must be reserved by the user prior to RJE processing.

UEXIT = USER - Specifies that the IBM-supplied user-exit routine is replaced by a user-written one.

COMPRESS= NO - Specifies that elimination of blanks is not to be used.

COMPRESS=YES - Specifies that elimination of blanks is to be used.

The parameters can be in any order and if more than one of them is specified, they have to be separated by a comma. The default options originally assumed by the RJE program are a leased point-to-point contention line, no reserved disk space for user-exit output and no elimination of blanks. If the LINE and/or COMPRESS parameters are omitted, the program assumes the last parameters specified as the current line configuration. If the UEXIT parameter is omitted, no space is reserved on disk for user-exit data.

The RJE00 program saves the information found in the parameters in a disk data file reserved for common constants used by the RJE program.

With the exception of the System Library Mainline Programs, this appendix lists all Monitor System WAITs and messages. SYSUP, the DCOM update subroutine, is also available in the System Library. The errors for the user callable version of SYSUP are listed in the System Library Utility Subroutines section of the manual. All messages for stand-alone utilities are included in the writeups of the individual programs.

System Loader, FORTRAN I/O and RPG object program errors cause the system to WAIT at $PRET. At the WAIT, bits 2 and 3 of the OPERATION REGISTER are on. FORTRAN I/O errors can be identified by the Fxxx code in the accumulator. RPG object program errors can be identified by the Cxxx code in the accumulator. A $PRET WAIT also occurs when a system I/O device is required but is not ready (see Table 18).

All error tables in this appendix are listed alphabetically by prefix letter. Unless otherwise noted, the operational and error messages are printed on the principal printer. All Monitor system control records are printed on the principal printer.

The error tables in order of appearance are as follows.

Table Number	Error Code Prefix	Program Name
8	A	Assembler
9	C	FORTRAN Compiler
10	D	Disk Utility Program (DUP)
11	E	System Loader
12	F	FORTRAN I/O
12A	G	Satellite Graphic Job Processor
13	M	Monitor Control Record Analyzer (MCRA)
14	M	Supervisor Control Record Program
15	-	SYSUP
15.1	NOTE	RPG Compiler
16	R	Core Load Builder
17	S	Auxiliary Supervisor
18	-	ISS Subroutine
18.1	-	RPG Object Program

Table 7. Assembler Error Detection Codes

Flag	Cause	Assembler Action
A	Address Error Attempt made to specify displacement field, directly or indirectly, outside range of -128 to +127.	Displacement set to zero
C	Condition Code Error Character other than +, -, Z, E, C, or O detected in first operand of short branch or second operand of long BSC, BOSC, or BSI statement.	Displacement set to zero
F	Format Code Error Character other than L, I, X, or blank detected in col. 32, or L or I format specified for instruction valid only in short form, or I format specified when not allowed.	Instruction processed as if L format were specified, unless that instruction is valid only in short form, in which case it is processed as if the X format were specified
L	Label Error Invalid symbol detected in label field.	Label ignored
M	Multiply Defined Label Error Duplicate symbol encountered in label field.	First occurrence of symbol in label field defines its value; subsequent occurrences of symbol in label field cause a multiply defined indicator to be inserted in symbol table entry (Bit 0 of first word).
O	Op Code Error Unrecognized op code	Statement ignored and address counter incremented by 2.
	ISS, ILS, ENT, LIBR, SPR, EPR, or ABS incorrectly placed.	Statement ignored
R	Relocation Error Expression does not have valid relocation.	Expression set to zero
	Non-absolute displacement specified.	Displacement set to zero
	Absolute origin specified in relocatable program.	Origin ignored
	Non-absolute operand specified in BSS or BES.	Operand assumed to be zero
	Non-relocatable operand in END statement of relocatable mainline program.	Card columns 9-12 left blank; entry assumed to be relative zero
	ENT operand non-relocatable.	Statement ignored
S	Syntax Error Invalid expression (e.g., invalid symbol, adjacent operators, illegal constant)	Expression set to zero
	Illegal character in record.	If illegal character appears in expression, label, op code, format, or tag field, additional errors may be caused.
	Main program entry point not specified in END operand.	Card columns 9-12 left blank; entry assumed to be relative zero
	Incorrect syntax in EBC statement (e.g., no delimiter in card column 35, zero character count).	Card columns 9-12 not punched; address counter incremented by 17.
	Invalid label in ENT or ISS operand.	Statement ignored
T	Tag Error Card column 33 contains character other than blank, 0, 1, 2, or 3 in instruction statement.	Tag of zero assumed
U	Undefined Symbol Undefined symbol in expression	Expression set to absolute zero
W	x- or y- coordinate, or both, not within the specified range; or invalid operand.	Operand set to zero.
X	Character other than R or I in column 32; or character other than D or N in column 33.	Field set to zero.
Z	Invalid condition in a conditional branch or interrupt order.	Condition bits in first word set to zero.

ASSEMBLER MESSAGES AND ERROR CODES

At the completion of an assembly, the following messages are printed on the principal printer.

XXX OVERFLOW SECTORS SPECIFIED
XXX OVERFLOW SECTORS REQUIRED
XXX SYMBOLS DEFINED
XX ERROR(S) FLAGGED IN ABOVE ASSEMBLY

If LIST DECK or LIST DECKE is specified, the error detection codes shown in Table 7 are punched in columns

18 and 19. For the first error detected in each statement the Assembler stores and then punches the code in column 18; the code for a second error is stored, overlaid by any subsequent errors, and punched in column 19. Thus, if more than two errors are detected in the same statement, only the first and last are indicated. These error detection codes will appear on the printout if the deck is listed.

At the end of the assembly, a message is printed indicating the number of assembly errors detected in the source program (see above). Since no more than two errors are flagged per statement, the error count may exceed the actual number of flags.

Assembler error messages are listed in Table 8.

FORTRAN MESSAGES AND ERROR CODES

Compilation Messages

Near the end of the compilation, core usage information and the features supported (control records used) are printed out as follows:

FEATURES SUPPORTED
EXTENDED PRECISION
ONE WORD INTEGERS
TRANSFER TRACE
ARITHMETIC TRACE
ORIGIN
IOCS
CORE REQUIREMENTS FOR XXXXX
COMMON YYYYY VARIABLES YYYYY PROGRAM YYYYY

where XXXXX is the name of the program designated in the *NAME control record or in the SUBROUTINE or FUNCTION statement, and YYYYY is the number of words allocated for the specified parts of the program.

The following messages are printed in the case of successful and unsuccessful compilations respectively.

END OF COMPILATION

COMPILATION DISCONTINUE

Compilation Error Messages

During compilation, a check is made to determine if certain errors have occurred. If one or more of these

Table 8. Assembler Error Messages

Error Number and Message	Cause of Error	Corrective Action
A01 MINIMUM W.S. NOT AVAILABLE... ASSEMBLY TERMINATED	Available Working Storage is less than the number of overflow sectors specified plus one sector.	Reduce the number of overflow sectors specified (number specified is zero if no *OVERFLOW SECTORS control record is used) or, If more than one drive is available on the system, use the //JOB record to specify System Working Storage on the cartridge with the most Working Storage available.
A02 SYMBOL TABLE OVERFLOW...ASSEMBLY TERMINATED	The number of sectors of symbol table overflow is greater than the number of overflow sectors available.	Use an *OVERFLOW SECTORS control record to increase the number of overflow sectors for this assembly (maximum 32 sectors).
A03 DISK OUTPUT EXCEEDS W.S.	Intermediate output (pass 1) or final DSF output (pass 2) exceeds the capacity of Working Storage less the number of overflow sectors specified.	If this error occurs during pass 1, the system will WAIT at location $PRET with /400E (2501) or /100E (1442) in the accumulator. Press PROGRAM START to continue the assembly in TWO PASS MODE. For pass 2, see options on A01.
A04 SAVE SYMBOL TABLE INHIBITED	With SAVE SYMBOL TABLE option specified: 1. Program is relocatable. 2. Program contains assembly errors. 3. Source program contains more than 100 symbols.	1. Use ABS card and reassembly. 2. Correct source program errors and reassemble. 3. Reduce the number of symbols and reassemble.
A05 XXX ERRONEOUS ORG, BSS, OR EQU STATEMENTS IN ABOVE ASSEMBLY	XXX is the number of ORG, BSS, BES, and/or EQU statements that were undefined in pass 1. At the end of pass 1, these erroneous statements are printed on the principal printer. If the error was due to forward referencing, it will not be detected during pass 2.	Where forward references have been attempted, they must be corrected before the program is reassembled.
A06 LOAD BLANK CARDS	A card containing a non-blank column between 1-71 has been read while punching a symbol table (*PUNCH SYMBOL TABLE specified for this assembly).	The system will WAIT at $PRET with /100F in the accumulator. Nonprocess run out (NPRO) the card just read. Place blank cards ahead of this card in the hopper. Press reader START and console PROGRAM START. NOTE: If the output is being punched on a 1442-5, a non-blank card cannot be detected. In addition, the punch may be damaged if an attempt is made to punch a hole where a hole already exists.
A07 *LEVEL CONTROL RECORD MISSING	The program listed above was assembled as an ISS subroutine without the required *LEVEL control record.	Reassemble using *LEVEL control record.

errors have been detected the error indications are printed at the conclusion of compilation, and no object program is stored on the disk. Only one error is detected for each statement. In addition, due to the interaction of error conditions, the occurrence of some errors may prevent the detection of others until those which have been detected are corrected. With the exception of the messages listed in table 8.1, the error message appears in the following format:

CNN ERROR IN STATEMENT NUMBER XXXXX+YY

NN is the error code number listed in Table 9. With the exception of specification statement errors, XXXXX is the last valid statement number preceding the erroneous statement and YYY is the count of statements from XXXXX to the statement that is in error. If the erroneous statement has a valid statement number, XXXXX will be the statement in error and YYY will not be printed.

For example:

```
105   FORMAT (I5, F8.4)
110   IF (A-B) 10,30,20
      A = A+1.0
ABC   B = B-2.0          (error C01)
135   GO TO 105          (error C43)
```

This example will cause the following error messages to be printed.

C01 ERROR IN STATEMENT NUMBER 110 + 002
C43 ERROR IN STATEMENT NUMBER 135

For specification statements, XXXXX is always 00000 and YYY is the count of the number of specification statements in error. YYY is never 000, i.e., for the first error YYY is 001. Specification statements are not counted unless they contain an error. Statement numbers on specification statements and statement functions are ignored. NN is the error code.

For example:

```
1   DIMENSION    C(10,10)
2   DIMENSION    D(5,5)
3   DIMENSION    E(I,6,6)     (error C08)
4   DIMENSION    F(4,4)
5   DIMENSION    G(2,2))      (error C16)
```

This example will cause the following error messages to be printed.

C08 ERROR AT STATEMENT 00000 + 001
C16 ERROR AT STATEMENT 00000 + 002

Table 8.1 FORTRAN Error Messages

Error Number and Message	Cause of Error
C85 ORIGIN IN SUB-PROGRAM	An ORIGIN control record was detected in a subprogram compilation.
C86 INVALID ORIGIN	An attempt has been made to relocate a word at an address exceeding 7FFF (hexadecimal).
C96 WORKING STORAGE EXCEEDED	The working storage area on disk is too small to accommodate the compiled program in disk system format.
C97 PROGRAM LENGTH EXCEEDS CAPACITY	The error occurs when the program in internal compiler format is too large to be contained in core working storage, and the program must be reduced in size in order to compile.
C98 SUBROUTINE INITIALIZE TOO LARGE	During compilation of Sub-programs a subroutine initialize statement (CALL SUBIN) is generated. The CALL SUBIN statement initializes all references to "dummy" variables contained within the subprogram to the appropriate core location in the calling program. The nature of the FORTRAN compiler limits the size of any statement in internal compiler format to 511 words. In the case of CALL SUBIN, the size is calculated by the following formula: S = 5 + ARG + N where ARG is the number of arguments in the subroutine parameter list and N is the total number of times the dummy arguments are used within the subprogram. S is the total size of the CALL SUBIN statement; if S ever exceeds 511, an error occurs and the above error message is printed.
C99 CORE REQUIREMENTS EXCESSIVE	The error occurs when the total core requirements exceed 32767 words.

Table 9. FORTRAN Error Codes

Error Number	Cause of Error
C01	Non-numeric character in statement number.
C02	More than five continuation cards, or continuation card out of sequence.
C03	Syntax error in CALL LINK or CALL EXIT statement.
C04	Undeterminable, misspelled, or incorrectly formed statement.
C05	Statement out of sequence.
C06	Statement following STOP, RETURN, CALL LINK, CALL EXIT, GO TO, or IF statement does not have statement number.
C07	Name longer than five characters, or name not starting with an alphabetic character.
C08	Incorrect or missing subscript within dimension information (DIMENSION, COMMON, REAL, or INTEGER).
C09	Duplicate statement number.
C10	Syntax error in COMMON statement.
C11	Duplicate name in COMMON statement.
C12	Syntax error in FUNCTION or SUBROUTINE statement.
C13	Parameter (dummy argument) appears in COMMON statement.
C14	Name appears twice as a parameter in SUBROUTINE or FUNCTION statement.
C15	*IOCS control record in a subprogram.
C16	Syntax error in DIMENSION statement.
C17	Subprogram name in DIMENSION statement.
C18	Name dimensioned more than once, or not dimensioned on first appearance of name.
C19	Syntax error in REAL, INTEGER, or EXTERNAL statement.
C20	Subprogram name in REAL or INTEGER statement or a FUNCTION subprogram containing its own name in an EXTERNAL statement.
C21	Name in EXTERNAL that is also in a COMMON or DIMENSION statement.
C22	IFIX or FLOAT in EXTERNAL statement.
C23	Invalid real constant.
C24	Invalid integer constant.
C25	More than 15 dummy arguments, or duplicate dummy argument in statement function argument list.
C26	Right parenthesis missing from a subscript expression.
C27	Syntax error in FORMAT statement.
C28	FORMAT statement without statement number.
C29	Field width specification greater than 145.
C30	In a FORMAT statement specifying E or F conversion, w greater than 127, d greater than 31, or d greater than w, where w is an unsigned integer constant specifying the total field length of the data, and d is an unsigned integer constant specifying the number of decimal places to the right of the decimal point.
C31	Subscript error in EQUIVALENCE statement.
C32	Subscripted variable in a statement function.
C33	Incorrectly formed subscript expression.
C34	Undefined variable in subscript expression.
C35	Number of subscripts in a subscript expression, and/or the range of the subscript(s) does not agree with the dimension information.
C36	Invalid arithmetic statement or variable; or, in a FUNCTION subprogram the left side of an arithmetic statement is a dummy argument or in COMMON.
C37	Syntax error in IF statement.
C38	Invalid expression in IF statement.
C39	Syntax error or invalid simple argument in CALL statement.
C40	Invalid expression in CALL statement.
C41	Invalid expression to the left of an equal sign in a statement function.
C42	Invalid expression to the right of an equal sign in a statement function.
C43	In an IF, GO TO, or DO statement, a statement number is missing, invalid, incorrectly placed, or is the number of a FORMAT statement.
C44	Syntax error in READ, WRITE, or FIND statement.
C45	*IOCS record missing with a READ or WRITE statement (mainline program only).
C46	FORMAT statement number missing or incorrect in a READ or WRITE statement.
C47	Syntax error in input/output list; or an invalid list element; or, in a FUNCTION subprogram, the input list element is a dummy argument or in COMMON.
C48	Syntax error in GO TO statement.
C49	Index of a computed GO TO is missing, invalid, or not preceded by a comma.
C50	*TRANSFER TRACE or *ARITHMETIC TRACE control record present, with no *IOCS control record in a mainline program.
C51	Incorrect nesting of DO statements; or the terminal statement of the associated DO statement is a GO TO, IF, RETURN, FORMAT, STOP, PAUSE, or DO statement.
C52	More than 25 nested DO statements.
C53	Syntax error in DO statement.
C54	Initial value in DO statement is zero.
C55	In a FUNCTION subprogram the index of DO is a dummy argument or in COMMON.
C56	Syntax error in BACKSPACE statement.
C57	Syntax error in REWIND statement.
C58	Syntax error in END FILE statement.
C59	Syntax error in STOP statement.
C60	Syntax error in PAUSE statement.
C61	Integer constant in STOP or PAUSE statement is greater than 9999.
C62	Last executable statement before END statement is not a STOP, GO TO, IF, CALL LINK, CALL EXIT, or RETURN statement.
C63	Statement contains more than 15 different subscript expressions.
C64	Statement too long to be scanned, because of compiler expansion of subscript expressions or compiler addition of generated temporary storage locations.
C65*	All variables are undefined in an EQUIVALENCE list.

Table 9. FORTRAN Error Codes (continued)

Error Number	Cause of Error
C66 *	Variable made equivalent to an element of an array in such a manner as to cause the array to extend beyond the origin of the COMMON area.
C67 *	Two variables or array elements in COMMON are equated, or the relative locations of two variables or array elements are assigned more than once (directly or indirectly).
C68	Syntax error in an EQUIVALENCE statement; or an illegal variable name in an EQUIVALENCE list.
C69	Subprogram does not contain a RETURN statement, or a mainline program contains a RETURN statement.
C70	No DEFINE FILE statement in a mainline program that has disk READ, WRITE, or FIND statements.
C71	Syntax error in DEFINE FILE statement.
C72	Duplicate DEFINE FILE statement, more than 75 DEFINE FILES, or DEFINE FILE statement in subprogram.
C73	Syntax error in record number of disk READ, WRITE, or FIND statement.
C74	Defined file exceeds disk storage size.
C75	Syntax error in DATA statement.
C76	Names and constants in a DATA statement not in a one to one correspondence.
C77	Mixed mode in DATA statement.
C78	Invalid hollerith constant in a DATA statement.
C79	Invalid hexadecimal specification in a DATA statement.
C80	Variable in a DATA statement not used elsewhere in the program or dummy variable in DATA statement.
C81	COMMON variable loaded with a DATA specification.
C82	DATA statement too long to compile, due to internal buffering.

* The detection of a code 65, 66, or 67 error prevents any subsequent detection of any of these three errors.

DUP MESSAGES AND ERROR CODES

When a DUP function is performed without error, an informational message is printed on the principal printer.

On a DEFINE VOID, one of the following messages is printed.

ASSEMBLER VOIDED
FORTRAN VOIDED
RPG VOIDED

On a DEFINE FIXED AREA, the message is as follows,

CART ID XXXX CYLS FXA XXXX DBS AVAIL XXXX
FLET SECTOR ADDR XXXX

where

CYLS FXA XXXX is the decimal number of cylinders -1 in the Fixed Area. The additional cylinder is used for FLET.

DBS AVAIL XXXX is the hexadecimal number of disk blocks remaining in the Fixed Area following the last program or data file.

FLET SECTOR ADDR XXXX is the hexadecimal sector address of the first cylinder in the Fixed Area, i.e., the sector address of FLET.

On a dump of LET or FLET, the printout is followed by a sign-off message.

END OF DUMPLET/FLET

All other DUP operations are followed by the following message.

CART ID XXXX DB ADDR XXXX DB CNT XXXX

where

DB ADDR XXXX is the hexadecimal starting address of the program or data file.

DB CNT XXXX is the hexadecimal number of disk blocks being deleted, stored, or dumped.

DUP error messages are listed in Table 10.

●Table 10. DUP Error Messages

Error Number and Message		Cause of Error
D01	NAME IS NOT PRIME ENTRY	The primary name of the program in Working Storage does not match the name on the DUP control record.
D02	INVALID HEADER RECORD TYPE	One of the following is detected: a non-DSF program, a mispositioned header, foreign data, or an erroneous subtype.
D03	INVALID HEADER LENGTH	Word six of the DSF header is outside the range of 3-45. The causes are similar to D02, except for subtype.
D05	SECONDARY ENTRY POINT OR NAME ALREADY IN LET	The specified secondary entry point name is already in LET. The name must be deleted before this subprogram can be stored.
D06	ENTRY POINT NAME ALREADY IN LET/FLET	The specified name is already in LET/FLET. The name must be deleted before this program or data file can be stored.
D12	INVALID DISK I/O SPECIFIED	Disk routine code on STORECI control record (column 9) was other than 0, 1, N, Z, or blank.
D13	INVALID FUNCTION FIELD (CC 1-12)	An invalid DUP function is specified in columns 1-12 of the DUP control record.
D14	INVALID FROM FIELD (CC 13-14)	Unacceptable characters are in columns 13 and 14 of the DUP control record. The FROM field specified is not valid with this DUP function.
D15	INVALID TO FIELD (CC 17-18)	Unacceptable characters are in columns 17 and 18 of the DUP control record. The TO field specified is not valid with this DUP function.
D16	INVALID NAME FIELD (CC 21-25)	No name specified and one required, or syntax error in construction of name.
D17	INVALID COUNT FIELD (CC 27-30)	Columns 27 through 30 are blank or include alphabetic characters. The count field requires a decimal number.
D18	INVALID FUNCTION DURING TEMPORARY JOB	This function is not allowed during the JOB T mode.
D19	CARTRIDGE NOT ON SYSTEM	Cartridge specified as TO or FROM cartridge was not specified on JOB record as being used in this job.
D20	CARTRIDGE ID OUTSIDE VALID RANGE (0001-7FFF)	Correct cartridge ID and retry.
D21	INVALID STOREMOD. SIZE OF REPLACEMENT EXCEEDS SIZE OF ORIGINAL	The replacement version of the program or data file is larger than the current version. The old version must be deleted before the replacement can be stored.
D22	PROGRAM NOT IN WORKING STORAGE	The disk block count for the requested program in Working Storage is zero. The program is not in Working Storage.
D23	INVALID SYSTEM OVERLAY SUBTYPE SPECIFIED	The system overlay subtype indicator (column 11) on a STORE control record is not in the range 0-9.
D24	COUNT FIELD TOO LARGE	The count field extends beyond column 30 of a DEFINE FIXED AREA control record or column 31 is not a minus sign.
D25	REQUIRED FORMAT NOT IN W.S.	During a STOREMOD, the format of the LET/FLET entry does not agree with the format in Working Storage.
D26	NAME NOT FOUND IN LET/FLET	The name specified on a DELETE or DUMP control record is not in LET/FLET.
D27	SOURCE NOT IN DSF	The format indicator of the FROM cartridge indicates that Working Storage on this cartridge does not contain a DSF program.
D30	INVALID RECORD TYPE	An invalid type binary record has been read when storing from cards or paper tape.
D31	PROGRAM OR DATA EXCEEDS DESTINATION DISK AREA	The number of disk blocks required to store a program or data exceeds the amount of space available in the specified TO field.
D32	INVALID CORE IMAGE CONVERSION	The Core Load Builder has inhibited the continuation of STORECI. The specific reason has been printed by the Core Load Builder.
D33	LET/FLET OVERFLOW. A CORE DUMP FOLLOWS	A ninth sector of LET/FLET is required (or a seventh sector of LET on a non-system cartridge) for the LET/FLET entry. A deletion of a program with a LET/FLET entry of similar size is required before this program can be stored. A core dump follows this message since the affected cartridge may have to be reloaded. The dump allows the user to locate the condition that caused the error. Use of the affected cartridge is not recommended until the problem has been investigated.
D41	INVALID STORECI CONTROL RECORD	The STORECI control record read was not a LOCAL, NOCAL, FILES or G2250 record; or a mainline name was specified on a G2250 record.
D42	STORECI CONTROL RECORDS INCORRECTLY ORDERED	LOCAL, NOCAL, FILES, and G2250 were intermixed. All records of a given type must be loaded together.
D43	INCORRECT CONTINUATION	A comma at the end of the record indicated that it would be continued; however, it was not.
D44	ILLEGAL CHARACTER IN RECORD	An illegal character, probably a blank, appeared in the record.
D45	ILLEGAL FILE NUMBER	A non-numeric character appears in a file number, or the number is more than five characters long.

●Table 10. DUP Error Messages (continued)

Error Number and Message		Cause of Error
D46	ILLEGAL NAME	A name is more than five characters long, or contains characters other than A-Z, 0-9, or $, or a name contains embedded blanks.
D47	ILLEGAL CARTRIDGE ID	The cartridge ID specified is not in the range/0001-/7FFF, or contains an illegal character.
D48	SCRA BUFFER OVERFLOW	The Supervisor Control Record Area (SCRA) cannot contain all the LOCAL, NOCAL, FILES, or G2250 information.
D50	NON-BLANK CARD READ ENTER BLANK CARDS	A non-blank card has been read during a dump to a 1442-6 or -7. Place blank cards in the hopper and ready the card read punch. Press PROGRAM START.
D70	LAST ENTRY IN LET/FLET NOT 1DUMY	DELETE cannot find the end of LET or FLET. The header for this LET/FLET sector contains the count of unused words in this sector. This count should point to the last 1DUMY entry; however, the entry to which it now points is not a 1DUMY.
D71	1DUMY ENTRY IN LET/FLET IS FOLLOWED BY A SECONDARY ENTRY POINT	The name on the DELETE control record points to a secondary entry point. The first entry in LET/FLET with a non-zero disk block count that precedes the secondary entry is a 1DUMY. The primary entry is not in LET/FLET.
D72	FIRST ENTRY IN LET/FLET SECTOR IS A SECONDARY ENTRY POINT	The LET/FLET table is improperly constructed. The first entry is not a primary entry.
D80	FIXED AREA PRESENT	The FORTRAN Compiler, RPG Compiler, or Assembler cannot be eliminated if a Fixed Area has been previously defined.
D81	ASSEMBLER NOT IN SYSTEM	The Assembler has previously been eliminated from the system.
D82	FORTRAN NOT IN SYSTEM	The FORTRAN Compiler has previously been eliminated from the system.
D83	INCREASE VALUE IN COUNT FIELD (CC 27-30)	The count field was read as a value of zero or one. The first DEFINE requires one cylinder for FLET plus one cylinder of Fixed Area. Thereafter, as little as one cylinder of additional Fixed Area can be defined.
D84	DEFECTIVE SLET	Cartridge must be reloaded.
D85	FIXED AREA NOT PRESENT	The control record specifies a decrease in the Fixed Area, or specifies Fixed Area as the destination, and there is no Fixed Area on the cartridge.
D86	DECREASE VALUE IN COUNT FIELD	There is insufficient Working Storage area available to allow the Fixed Area to be defined or expanded by the amount specified in the count field (cc 27-30). This message is preceded by a count of the number of cylinders available XXXX CYLS AVAILABLE. The count is in decimal.
D87	RPG NOT IN SYSTEM	The RPG Compiler has previously been eliminated from the system.
D90	CHECK SUM ERROR	Checksum error in binary card or paper tape record, or binary cards are out of order.
D92	INVALID DISKZ CALL. A CORE DUMP FOLLOWS	While performing a DUP function, an attempt has been made to read or write sector 0, or to read or write with a negative word count. This is a system error. A core dump follows this message since the affected cartridge may have to be reloaded. The dump allows the user to locate the condition that caused the error. Use of the affected cartridge is not recommended until the problem has been investigated.
D93	CARTRIDGE OVERFLOW	While performing a DUP function, an attempt has been made to read or write a sector beyond 1599 decimal.

SYSTEM LOADER MESSAGES AND ERROR CODES

No informational messages are printed during an initial load. At the completion of a reload, the following message is printed,

END RELOAD

Table 11 lists the System Loader Errors.

Table 11. System Loader Errors

Error Number and Message		Corrective Action
	From Phases 1 and 2	
E01	CHECKSUM ERROR	Follow procedure A or restart initial load.
E02	INVALID RECORD OR BLANK	Follow procedure A or restart initial load.
E03	SEQ ERROR OR MISSING RECORDS	Follow procedure A or restart initial load. The missing record may be end-of-program record. See RELOAD for more information.
E04	ORG BACKWARD	Inspect the deck for records missing or out of sequence. Correct the deck and restart from the record in error.
E05	INITIALIZE THE CARTRIDGE	The cartridge ID cannot be found in DCOM because DCOM is defective or an attempt is being made to initial load a cartridge that has not just been initialized or has been improperly initialized. Initialize and initial load the cartridge.
	From Phase 1 only	
E11	INVALID DRIVE NO.	Set all bit switches off. Set bit switches to select physical drive number and press PROGRAM START. All switches off - Drive 0 Switch 15 on - Drive 1 Switch 14 on - Drive 2 Switches 14, 15 on - Drive 3 Switch 13 on - Drive 4
E12	ID SECTOR DATA INVALID	Initialize using DCIP or DISC and follow with an initial load.
E13	CONFIG DECK ERROR	System configuration deck may be missing, out of place, or may contain error in one or more records. Correct the deck and restart load.
E14	FILE PROTECT ADDR TOO HIGH	This error will occur on a reload only. The last program in the User Area extends into the last two cylinders on the cartridge. These cylinders are required by the System Loader during a reload operation. The file protect address must be lowered before a reload can be accomplished.
E15	PHID RECORD ERROR	Follow procedure A or reload and restart.
E16	INITIAL LOAD THE CARTRIDGE	The ID sector indicates that this cartridge has not been loaded since initialization by DCIP or DISC. Only an initial load may be performed.
E17	ERROR IN LOAD MODE RECORD	Follow procedure A or restart load.
E18	PAPER TAPE ERROR	The paper tape System Loader has found a word count greater than 54. This is probably due to incorrect sequencing of tapes, a faulty tape or a paper tape reader malfunction. Correct error and restart load.
E19	INVALID SLET/RELOAD TABLE CHECKSUM	System Loader will ignore the checksum and continue if PROGRAM START is pressed. However, it is recommended that the cartridge be initialized and an initial load performed.
	From Phase 2 only	
E20	FIXED AREA PRESENT	Programs may not be added to a cartridge with a fixed area defined. Press PROGRAM START to restore the Resident Image and DCOM.
E21	SYSTEM DECK ERROR	A defective record follows the sector break record. Correct the deck and restart the initial load or continue the reload from the preceding sector break record.
E22	SCRA OVERLAY - STOP	The cushion area used for allowing expanded or added phases has been used up. An initial load must be performed to store these phases on the cartridge. Press PROGRAM START to restore the Resident Image and DCOM.
E23	PHASE ID OUT OF SEQUENCE	The Accumulator displays the phase ID that is out of sequence (from last card read). Place the decks in proper order and continue from the sector break record of the correct phase.
E24	PHASE MISSING	Error occurred when phase ID (word 11) of last record read was processed. Inspect Load Mode record, PHID record and phase ID of previously loaded phase to determine which phase is now required. Locate missing phase, place deck in reader starting with sector break record of missing phase and continue.
E25	PHASE ID NOT IN PHID RECORD	The Accumulator displays the extraneous phase ID. To ignore the phase press PROGRAM START. To load the phase correct the PHID record and restart the load.
E26	PHASE ID NOT IN SLET	If the error occurred during Reload Table processing, the Accumulator displays the phase ID sought, and the Extension displays the ID of the phase requesting the SLET search. Press PROGRAM START to place zeros in the entry and process the next. If the Extension displays zeros, a phase is being added, and the phase which should precede it cannot be found. The accumulator displays the phase ID searched for. Press PROGRAM START to restore the Resident Image and DCOM.
E27	DEFECTIVE SLET	SLET is defective. Initialize the cartridge and perform an initial load.
E28	SLET FULL	The Accumulator displays the ID of a phase that may not be added because the SLET table is full. Press PROGRAM START to ignore the phase and continue. An initial load should be performed as SLET is probably defective.
E29	PROGRAM NOT PRESENT	A program or phases of a program defined in the primary PHID record cannot be reloaded unless the program is currently on the cartridge. Press PROGRAM START to ignore the phases of this program.
E30	RELOAD TABLE FULL	If this error occurs before the '81' record is read the Accumulator displays the ID of a phase which may not be loaded because the reload table is full. Press PROGRAM START to ignore the phase and continue.
E31	MISSING PHASE ID DUE TO DEFECTIVE SLET OR RELOAD TABLE	The Accumulator displays the ID of a phase listed in the reload table as a phase requiring SLET information but the phase itself does not appear in SLET. Initialize the cartridge and perform an initial load.
E32	MISSING SYSTEM I/O PHASE	All system I/O subroutines must be on the cartridge and in SLET. Initialize the cartridge and perform an initial load.

Table 11. System Loader Errors (continued)

Procedure A

If cards are being read from a 1442 Card Read Punch:

1. Lift the remaining cards from the hopper and press nonprocess run out (NPRO).
2. Correct the card in error (first card nonprocessed out) and place the two nonprocessed cards ahead of the cards removed from the hopper.
3. Place the deck back in the hopper.
4. Press reader START.
5. Press console PROGRAM START.

If cards are being read from a 2501 Card Reader:

1. Lift the remaining cards from the hopper and press NPRO.
2. a. Correct the card in error (last card in stacker prior to NPRO) and place this card followed by the single nonprocessed card ahead of the cards removed from the hopper or,
   b. If the error occurred after the PHID card was read and before the type 81 card was read the System Loader is in double buffer mode. Correct the card in error (in this case the second from last card in the stacker when the error occurred) and place the last two cards from the stacker and the nonprocessed card ahead of the cards removed from the hopper. Note, however, that the last card in the stacker will be the next card processed since it is already in the double-buffer.
3. Place the deck back in the hopper.
4. Press reader START.
5. Press console PROGRAM START.

Table 12. FORTRAN I/O Errors

Accumulator Display	Cause of Error
F000	No *IOCS card appeared with the mainline program and I/O was attempted in a subroutine.
F001	Logical unit defined incorrectly, or No *IOCS control record for specified I/O device.
F002	Requested record exceeds allocated buffer size.
F003	Illegal character encountered in input record.
F004	Exponent too large or too small in input field.
F005	More than one E encountered in input field.
F006	More than one sign encountered in input field.
F007	More than one decimal point encountered in input field.
F008	Read of output-only device, or Write of input-only device.
F009	Real variable transmitted with an I format specification or integer variable transmitted with an E or F format specification.
F020	Illegal unit reference.*
F021	Read list exceeds length of write list.*
F022	Record does not exist for read list element.*
F023	Maximum length of $$$$$ area on the disk has been exceeded. This error is unrecoverable and result in a call exit.*
F100	File not defined by DEFINE FILE statement.
F101	File record too large, equal to zero, or negative.
F103	Disk FIO (SDFIO) has not been initialized, e.g., there is no *IOCS(DISK) record in the mainline program.
F10A	Subscripting has destroyed the Define File Table. This occurs when a subscript exceeds the specification in a DIMENSION statement.
	*) Can occur in unformatted I/O operations.

FORTRAN I/O ERRORS

When a FORTRAN I/O error occurs, the system WAITs at $PRET with an Fxxx error code displayed in the accumulator. Table 12 lists the FORTRAN I/O errors.

SATELLITE GRAPHIC JOB PROCESSOR ERROR MESSAGES

Table 12A lists the error messages produced by the Satellite Graphic Job Processor (SGJP). The numbered messages appear on the console printer; the messages preceded by IKyxxxz on the 2250 screen. The corrective action is to be performed by the 1130 operator or 2250 user. SGJP is described in detail in the publication IBM System/360 Operating System and 1130 Disk Monitor System: User's Guide for Job Control From an IBM 2250 Display Unit Attached to an IBM 1130 System, Form C27-6938.

Table 12A. SGJP Error Messages

Error Number (if any) and Message	Cause and Corrective Action
G01 INITIALIZATION FAILURE	No contact has been made with SGJP in the System/360 during an attempt to initialize the telecommunications line via the GTNIT data transmission subroutine. Remedy Ensure that the System/360 operator has issued a VARY ON command for the 1130/2250 subsystem on which this error message was printed. Then, using the console keyboard, type in either an R to retry the operation or a C to cancel SGJP.
G02 LINE ERROR	An attempt to transmit data to the System/360 was unsuccessful because of an input/output error; standard retries were unsuccessful. Remedy Using the console keyboard, type in either an R tc retry the operation or a C to cancel SGJP.
G03 SYNCHRONIZATION ERROR	The operation was not completed, either because both the System/360 and the 1130/2250 subsystem were in read mode, or because the System/360 terminated communication. Remedy Using the console keyboard, type in either an R to retry the operation or a C to cancel SGJP.
IKyxxxz message text THE SATELLITE GRAPHIC JOB PROCESSOR MUST RESTART	SGJP was terminated because an internal error occurred. If the error recurs, using the message code (IKyxxxz), refer to the publication IBM System/360 Operating System: Messages and Codes, Form C28-6631, for further explanation of the error condition. Remedy Perform the END function, which will cause the LOG ON frame to reappear. Perform the LOG ON operation again.
IKyxxxz message text THE SATELLITE GRAPHIC JOB PROCESSOR MUST TERMINATE	SGJP must be terminated because an internal error occurred. If the error recurs, using the message code (IKyxxxz), refer to the publication IBM System/360 Operating System: Messages and Codes, Form C28-6631, for further explanation of the error condition. Remedy Perform the END function. This will return SGJP to the state it was in before the initial (CANCEL key) attention.

SUPERVISOR MESSAGES AND ERROR CODES

The monitor Supervisor causes all Monitor system control records to be printed on the principal printer.

During a DCOM update operation (i.e., following each JOB record or user call to SYSUP) the following message is printed.

```
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE

   XXXX        XXXX        XXXX        XXXX

V2 MXX      ACTUAL XXK      CONFIG XXK
```

where

LOG DRIVE is the drive number specified on the JOB card (in the calling sequence of the SYSUP subroutine)

CART SPEC is specified cartridge ID

CART AVAIL is the available cartridge ID

PHY DRIVE is the physical drive number starting with 0.

V2 MXX is the current version and modification level
ACTUAL XXK indicates that XXK is the physical core size
CONFIG XXK indicates that XXK is the configured core size by system generation.

The logical drive may be different from the physical drive, e.g., physical drive 0 may be defined as logical drive 2.

One line is printed for each physical drive on the system.

Tables 13, 14, and 15 list Supervisor errors.

Table 13. Phase 1, Monitor Control Record Analyzer Errors

Error Number and Message		Cause of Error
M11	INVALID MONITOR CONTROL RECORD	A // record was not recognized as a valid Monitor control record.
M12	EXECUTION SUPPRESSED	$NXEQ was set upon detection of an error that would prevent successful execution by the system. Execution is bypassed.
M13	DUP SUPPRESSED	$NDUP was set upon detection of an error that would prevent successful DUP operation. DUP is bypassed.
M14	SYSTEM PROGRAM DETECTED MONITOR CONTROL RECORD	A system program has detected a Monitor control record when none was expected. The control record is passed to the MCRA for processing. This situation often occurs as a result of a missing END statement in an Assembler language program.
M15	ILLEGAL CARTRIDGE ID	A cartridge ID contains an illegal character or is a negative number. The job is aborted.
M16	PROGRAM VOIDED	ASM, FOR or RPG required but the FORTRAN Compiler and/or Assembler and/or RPG compiler was either not loaded by the System Loader or was voided by a DUP DEFINE.

Table 14. Phase 2. System Control Record Program Errors (Phase 2 errors cause execution to be bypassed).

Error Number and Message	Cause of Error
M21 ABOVE RECORD NOT A SUPERVISOR CONTROL RECORD	The last record read is not a LOCAL, NOCAL, G2250 or FILES record.
M22 SUPERVISOR CONTROL RECORDS INCORRECTLY ORDERED	LOCAL, NOCAL, FILES, and G2250 records cannot be intermixed.
M23 INCORRECT CONTINUATION	A comma at the end of the record indicated that the record would be continued; however, it was not.
M24 ILLEGAL CHARACTER IN RECORD	An illegal character, probably a blank, appeared in the record.
M25 ILLEGAL FILE NUMBER	A non-numeric character appears in a file number, or the number is more than five characters long.
M26 ILLEGAL NAME	A name is more than five characters long, or contains characters other than A-Z, 0-9, or $, or a name contains embedded blanks.
M27 ILLEGAL CARTRIDGE ID	The cartridge ID specified is not in the range /0001-/7FFF, or contains an illegal character.
M28 SCRA BUFFER OVERFLOW	The Supervisor Control Record Area (SCRA) cannot contain all the LOCAL, NOCAL, FILES, or G2250 record information.
M29 ILLEGAL DISK SUBROUTINE REQUESTED	A character other than 0, 1, N, Z, or blank appeared in column 19 of the XEQ card.
M30 INVALID CHAR. IN G2250 OPTION COLUMN	A character other than U, N, or blank appeared in column 13, 15, 17, 19 or 21 of the *G2250 control record.
M31 REQUESTED W. S. DR NOT AVAIL.	The requested cartridge has not been specified in the job record.

Table 15. SYSUP - DCOM Update Errors (SYSUP errors are also listed with the System Library Utility Subroutine SYSUP).

Cartridge ID and Message	Cause of Error
XXXX IS NOT AN AVAILABLE CARTRIDGE ID	A requested cartridge ID is not on any cartridge on the system, or the ID is not listed in #CIDN of the DCOM on the cartridge.
XXXX IS A DUPLICATED SPECIFIED CARTRIDGE ID	The cartridge ID was listed as appearing on more than one drive on the JOB card.
XXXX IS A DUPLICATED AVAILABLE CARTRIDGE ID	A specified ID appears on more than one cartridge on the system.
XXXX IS NOT A SYSTEM CARTRIDGE	An attempt has been made to specify a non-system cartridge as the master cartridge (logical 0).

CORE LOAD BUILDER ERRORS

Except for the core load map described in the Programming Tips and Techniques section and messages R41-R45, the Core Load Builder prints no informational messages. Table 16 lists Core Load Builder Error Messages.

RPG COMPILER MESSAGES AND ERROR NOTES

Compiler Messages

Near the end of the compilation, core usage information and literal parameters are printed in the following format. (See Sample Program 3 in Appendix J for an actual printout.) The relative address fiven can be used to compute the actual address of the indicator, field, litera, or routine the program is loaded, as follows: add the relative address to the execution address (as printed on the core map) and subtract hex 11 from the sum. The remainder, then, is the actual address.

INDICATORS

IND DISP ... Indicators through H9 are printed for all programs
(relative address)

FIELD NAMES

FIELD (field name)	DISP	L (field length)	T (field type)	D ... (number of decimal positions)

LITERALS

LITERAL	LENGTH	TYPE	DISP ...

KEY ADDRESSES OF OBJECT PROGRAM

Name of Routine	Hex DISP ...

END OF COMPILATION

Compilation Errors

If Working Storage is exceeded, compilation is terminated and the following message is printed.

WORKING STORAGE EXCEEDED

If terminating errors are detected during compilation the following messages are printed.

ERROR(S) IN COMPILATION
END OF COMPILATION

The program will execute if none of the errors detected were in the uncorrectable class, that is, no error note number was preceded by an asterisk (NOTE *XXX).

Compiler error notes are printed as follows.

1. As each statement is processed it is checked for invalid conditions. When an error is detected, the error note is printed on the line following the line in error in the columns reserved for program ID.

   NOTE XXX (xxx is a 3-digit error number. See Table 15.1).

2. The source program is checked for invalid file references (modified, unreferenced, multidefined) and error notes are printed as required. These notes are printed within or below the source listing in the following form.

   NAME NOTE XXX

3. Following the printout of Indicators, Field Names and Literals, any errors on extended diagnostics are printed in the following format (the sequence number is a 4-digit number assigned to source program statements. Comments cards are not sequence numbered). Some error messages (e.g., 227 and 228), are printed together with the number of the statement following the error because the error cannot be determined until then.

	Seq No.	Error
EXTENDED FILE DEF. EXT. AND/OR INPUT DIAGNOSTICS	XXXX	NOTE XXX
EXTENDED CALCULATION SPECIFICATION DIAGNOSTICS	XXXX	NOTE XXX
EXTENDED OUTPUT SPECIFICATION DIAGNOSTICS	XXXX	NOTE XXX

4. After the extended diagnostics, a summary of all error messages is printed as follows.

DIAGNOSTIC MESSAGE EXPLANATIONS

NOTE XXX Y ERROR MESSAGE
(Y is the specification type.)

or

NOTE *XXX Y ERROR MESSAGE
***UNCORR ERR JOB TERM
(* indicates that error is uncorrectable. The program will not execute. The ley addresses of the program will not be printed.)

All RPG Compiler error notes and their explanations are listed in Table 15.1.

The term "specification is dropped" means that a statement will not be further processed by the compiler; the tern "no immediate action taken" means that the compiler will continue processing a statement, by looking for additional errors.

• Table 15.1 - RPG Compiler Error Notes (Part 1 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
* 1	F	FILE TYPE COL 15 INVALID	File Type entry is not I, O, U or C, or is blank.	I is assumed.
* 2	F	PROC MODE COL 28 INVALID	Mode of Processing entry is not L, R, or blank.	Blank is assumed.
* 3	F	REC ADDR COL 29-30 INVALID	Length of Record Address Field (or key length) entry is invalid or is blank.	08 is assumed.
4	F	REC ADDR TYPE COL 31 INVALID. CORRECT ENTRY ASSUM	Warning only. The correct value for the file type (col 32) is assumed.	Blank is assumed for sequential files K is assumed for ISAM files.
5	F	TABLE FILE COL 16 REQ E COL 39. E ASSUM	Extension Code entry must be E if File Designation entry is T (table file).	E is assumed.
* 6	F	FILE DESIGN INVALID WITH INPUT FILE	File Designation entry col 16 is not P, S, R, C, or T with an input file (I in col 15).	P is assumed.
7	F	OF IND COL 33-34 INVALID BLK ASSUM	Overflow Indicator entry is invalid for the device type specified.	Blanks are assumed.
8	F	FILE TYPE COL 15 INVALID O ASSUM	File Type entry is invalid with a printer device in col 40-46.	O is assumed.
9	F	MULT PRI FILES DEF. SEC ASSUM	Only one primary file (P in col 16) is allowed. Other input files are designated as secondary (S in col 16).	Secondary is assumed for all but first input file.
* 11	F	FILE ORG COL 32 INVALID	File Organization entry is not I, numeric (1-9), or blank; or, two I/O areas are specified for a table File.	Blank is assumed.
12	F	EXT CODE COL 39 NOT BLK. BLK ASSUM	Extension Code must be blank for output files.	Blank is assumed.
13	F	EOF COL 17 INVALID E ASSUM	End of File entry is not E or blank.	E is assumed.
14	F	SEQ COL 18 INVALID A ASSUM	Sequence entry not A, D, or blank.	A is assumed.
15	F	FILE DESIG COL 16 NOT BLK. BLK ASSUM	File Designation entry is not blank for an output file.	Blank is assumed.
* 16	F	C IN FILE TYPE COL 15 INVALID WITH DEVICE	File Type entry C requires card read punch in Device col 40-46.	READ42 is assumed.

•Table 15.1 RPG Compiler Error Notes (Part 2 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
17	F	REC ADDR FILE REQ E COL 39. E ASSUM	File Designation entry R (Record Address File) requires an E in Extension Code.	E is assumed.
18	F	FILE FMT INVALID. F ASSUM	File Format (Col. 19) is not F. 1130 RPG uses fixed length records only.	F is assumed.
19	F	BLOCK LNG COL 20-23 NOT BLK. BLK ASSUM	Block Length must be blank for 1130 RPG.	Blanks are assumed.
20	F	REC LNG COL 24-27 INVALID 80 ASSUM	Record Length is improperly specified or is blank.	0080 is assumed.
* 21	F	U IN FILE TYPE COL 15 INVALID WITH DEVICE	File Type entry U requires disk I/O in Device col 40-46	DISK is assumed.
22	F	COL 17-18 INVALID WITH PRINTER. BLK ASSUM	End-of-File and Sequence entries are invalid with a printer.	Blanks are assumed.
* 24	F	MORE THAN 8 SEC FILES DEF	The number of secondary files (S in col 16) exceeds the maximum allowable 8.	8 is assumed.
25	F	OF IND COL 33-34 INVALID. BLK ASSUM	Overflow indicator not OF or OV.	Blanks are assumed.
27	F	EOF COL 17 NOT BLK WITH OUTPUT. BLK ASSUM	End of File entry must be blank with output files.	Blank is assumed.
29	F	EXT CODE 39 INVALID. E ASSUM	Extension Code entry is not E or blank with input file.	E is assumed.
* 30	E	FROM FILENAME COL 11-18 INVALID	From Filename entry is missing or not left-justified.	Specification is dropped.
* 31	E	FROM FILENAME COL 11-18 INVALID	From Filename entry was not defined on a File Description Specification.	Specification is dropped.
* 32	E	FROM FILENAME COL 11-18 INVALID	From Filename entry requires an E in Extension Code on the File Description Specification.	Specification is dropped.
* 33	E	CHAINING FLD COL 9-10 INVALID	Chaining Field Entry is not C1, C2, or C3 for chaining file (same entry as col 61-62 of input spec).	Specification is dropped.

•Table 15.1 RPG Compiler Error Notes (Part 3 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
* 34	E	SEQ COL 7-8 INVALID	Record Sequence entry must be two alphabetic or two numeric characters for chaining file (same entry as col 15-16 of input spec.).	Specification is dropped.
* 35	E	TO FILENAME COL 19-26 INVALID	To Filename entry is missing or not left-justified on RAF or chaining type specs.	Specification is dropped.
* 36	E	TO FILENAME COL 19-26 INVALID	To Filename entry was not defined on RAF or chaining type specs on a File Description Specification.	Specification is dropped.
* 37	E	TO FILENAME COL 19-26 INVALID	To Filename entry is not the same as the filename defined as a RAF or chaining type spec on a File Description Specification.	Specification is dropped.
38	E	COL 33-57 NOT BLK. BLK ASSUM	Columns 33-57 of the Extension Specification must be blank for all chaining type specifications.	Blanks are assumed.
39	E	COL 7-10 NOT BLK. BLK ASSUM	Columns 7-10 of the Extension Specification must be blank for all RAF type specifications.	Blanks are assumed.
40	E	COL 33-57 NOT BLK. BLK ASSUM	Columns 33-57 of the Extension Specification must be blank for all RAF type specifications.	Blanks are assumed.
41	E	COL 7-10 NOT BLK. BLK ASSUM	Columns 7-10 of the Extension Specification must be blank for all table type specifications.	Blanks are assumed.
* 42	E	TO FILENAME COL 19-26 INVALID	To Filename entry is missing or not left-justified.	Specification is dropped.
* 43	E	TO FILENAME COL 19-26 INVALID	To Filename entry was not defined on a File Description Specification.	Specification is dropped.
* 44	E	TO FILENAME COL 19-26 INVALID	To Filename entry is not defined as an output file on a File Description Specification.	Blanks are assumed.
* 45	E	TBL NAME COL 27-32 OR 46-51 INVALID	Table Name entries missing or not left-justified. 46-51 is required for alternating input formats only.	Specification is dropped.

•Table 15.1 RPG Compiler Error Notes (Part 4 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
* 46	E	COL 27-29 OR 46-48 NOT TAB	First three characters of table names must be TAB. 46-48 is required for alternating input formats only.	TAB is assumed.
* 47	E	NO OF TBL ENTRIES COL 33-35 NOT NUMERIC	Number of table entries per record. These columns must contain a right-justified decimal number.	10 is assumed.
* 48	E	NO OF TBL ENTRIES COL 36-39 NOT NUMERIC	Number of table entries per table. These columns must contain a right-justified decimal number.	100 is assumed.
* 49	E	TBL ENTRY LGN COL 40-42 OR 52-54 NOT NUMERIC	Length of table entry. These columns must contain a right-justified decimal number. 52-54 is required for alternating input formats only.	8 is assumed.
50	E	PACKED ENTRY COL 43 OR 55 INVALID. BLK ASSUM	Packed entry is not P or blank, or invalid for specified device.	Blank is assumed.
* 51	E	NUM DEC POS COL 44 OR 56 INVALID	Decimal Positions is not blank or a number.	Zero is assumed.
52	E	TBL SEQ COL 45 OR 57 INVALID. BLK ASSUM	Sequence entry is not A, D, or blank.	Blank is assumed.
* 53	E	FORM TYPE COL 6 NOT VALID	The next specification should have been an E or I Specification.	Specification is dropped.
56	F	COL 47-65, 67-70 MUST BE BLK FOR 1130 RPG	Specified columns are not used with 1130 RPG except for ISAM load files.	Blanks are assumed.
57	F	ISAM NUMBER OF RECORDS INVALID	The number of records specified for an ISAM load (col. 47-52) is not numberic or left-justified.	99999 is assumed.
60	H	NO RPG CONTROL CARD. BLK ASSUM	Warning only. A compilation and listing will be performed for this run.	Blanks are assumed for all entries.
61	H	COL 11 INVALID. BLK ASSUM	Type of run. This entry should be B, D, or blank.	Blank is assumed.

•Table 15.1 RPG Compiler Error Notes (Part 5 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
63	H	COL 17-20 INVALID. BLK ASSUM	Sterling entries are not blank, 0, 1, or 2, as required.	Blanks are assumed.
64	H	COL 21 INVALID. BLK ASSUM	Inverted print option entry is not I or blank.	Blank is assumed.
65	H	COL 26 INVALID. BLK ASSUM	Alternating collating sequence entry is not A or blank.	Blank is assumed.
67	H	PROG NAME COL 75-80 INV. RPGOBJ ASSUM	Program Name entry on RPG Control Card is invalid.	RPGOBJ is assumed.
* 71	C	RSLT FLD COL 43-48 REQUIRED	Result Field name is required but is missing.	Specification dropped.
72	C	RSLT FLD COL 43-48 MUST BE BLK. BLK ASSUM	Result Field must be blank for COMP, GOTO EXIT TAG, SETOF, SETON, CHAIN, BEGSR, ENDSR, EXSR, and EXCPT	Blanks are assumed.
* 73	C	FACT1 COL 18-27 INVALID	Factor 1 requires a field name, label or literal to be used with the specified operation.	Numeric literal 1 is assumed.
* 74	C	FACT2 COL 33-42 INVALID	Factor 2 requires a field name, label, or literal to be used with the specified operation.	Numeric literal 1 is assumed.
75	C	RSLT IND COL 54-59 INVALID. 00 ASSUM	Resulting Indicator is not 01-99, H1-H9, L1-L9, OF, or OV.	00 is assumed for indicator in error.
76	C	FACT1 COL 18-27 MUST BE BLK. BLK ASSUM	Factor 1 entry must be blank for the operation being performed.	Blanks are assumed.
77	C	FACT2 COL 33-42 MUST BE BLK. BLK ASSUM	Factor 2 entry must be blank for the operation being performed.	Blanks are assumed.
* 78	C	CTRL LEVEL COL 7-8 INVALID	Control Level Col 7 is not L or blank.	Blank is assumed.
* 79	C	DETAIL CALC DOES NOT PRECEDE TOTAL CALC	A detail calc, col 7-8 blank, follows a total calc, col 7-8 L0-L9 or LR.	L0 is assumed.
* 80	C	FACT1 COL 18-27 INVALID	Factor 1 entry is not left-justified.	Numeric literal 1 is assumed.
* 81	C	FACT2 COL 33-42 INVALID	Factor 2 entry is not left-justified.	Numeric literal 1 is assumed.

•Table 15.1 RPG Compiler Error Notes (Part 6 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
* 82	C	FACT1 COL 18-27 INVALID	Factor 1 entry is an improperly stated literal or field name.	Numeric literal 1 is assumed.
* 83	C	FACT2 COL 33-42 INVALID	Factor 2 entry is an improperly stated literal or field name.	Numeric literal 1 is assumed.
* 84	C	FACT1 COL 18-27 INVALID	Factor 1 entry is a field name of more than six characters.	First six characters are assumed.
* 85	C	FACT2 COL 33-42 INVALID	Factor 2 entry is a field name of more than six characters.	First six characters are assumed.
* 86	C	OPER CODE COL 28-32 INVALID	Operation code is missing or unrecognizable.	MOVE op code is assumed.
87	C	CTRL LEV COL 7-8 INVALID. L0 ASSUM	Col 7 is L but column 8 is not 0-9 or R.	L0 is assumed.
* 89	C	RSLT FLD COL 43-48 REQUIRED	Result Field entry is improperly defined.	Specification dropped.
* 94	C	RSLT FLD LNG COL 49-51 INVALID	Field Length entry is blank, not numeric, or not right-justified; OR, Field Length entry contains an embedded blank.	0 is assumed for blank.
* 95	C	DEC POS COL 52 INVALID	Decimal Position entry is not blank or numeric.	0 is assumed.
96	C	HLF ADJ COL 53 INVALID. H ASSUM	Half adjust entry is not H or blank.	H is assumed.
* 97	C	RSLT IND COL 54-59 REQUIRED	A resulting indicator is required for this operation.	Internal indicator is assigned.
* 98	C	IND COL 9-17 INVALID	Indicator entry improperly defined.	Indicator is dropped.
*100		STERL COL 71-74 INVALID	Sterling entry not numeric or sterling not defined on RPQ Control Card. This note can be printed by input or output specifications.	Blanks are assumed.
*101	I	FLD REC RELATION IND COL 63-64 INVALID	Field Record Relation Indicator unrecognizable.	Blanks are assumed.
*102	I	PLUS, MINUS, ZERO/BLK IND COL 65-70 INVALID.	Indicator column 65-70 unrecognizable.	Blanks are assumed.

•Table 15.1 RPG Compiler Error Notes (Part 7 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*109		INPUT OR OUTPUT SPECS MISSING OR INVALID	Input or output specs are required.	Job is terminated.
110	I	FORM TYPE COL 6 INVALID	Form Type is not I, C, or O and column 7 does not contain an *.	Specification is dropped.
*111	I	FILENAME COL 7-14 INVALID	Filename entry is not defined.	Specification is dropped.
*112	I	FILENAME COL 7-14 INVALID	Filename entry is not correctly defined on the File Description sheet.	Specification is dropped.
*113	I	'AND' CD OUT OF SEQ	'AND' card first card in deck, first spec after field name, or invalid file type.	Specification is dropped.
*114	I	NO RECORD ID IN CARD BEFORE 'AND' CARD	Record ID entry col 21-41 of Input Specification required in card before 'AND' card.	Specification is dropped.
*115	I	'OR' CD OUT OF SEQ	'OR' card first card in deck, first spec after field name, or invalid file type.	Specification is dropped.
*116	I	FILENAME COL 7-14 INVALID	Filename entry not left-justified.	Specification is dropped.
*117	I	FILENAME COL 7-14 INVALID	Filename entry begins with a numeric character.	Specification is dropped.
*118	I	FILE AND FLD NAME ARE BOTH ON SAME SPEC	File and field names cannot both appear on same spec.	Filename entry is assumed.
119	I	SEQ COL 15-16 BLK. AA ASSUM	Sequence entry must be two alpha or two numeric characters.	AA is assumed.
*120	I	SEQ COL 15-16 ALPHA SEQ AFTER NUM SEQ	Alpha sequence entries must appear before numeric sequence entries.	Numeric sequence last used is assumed.
*121	I	SEQ COL 15-16 IS INVALID	Ascending numeric sequence is required, or the first entries must begin with 01.	Numeric sequence last used is assumed.
122	I	NUMBER ENTRY COL 17 INVALID. N ASSUM	Sequence is numeric and the number entry column is not N or 1.	N is assumed.

•Table 15.1 RPG Compiler Error Notes (Part 8 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
123	I	OPTION ENTRY COL 18 INVALID. O ASSUM	Sequence is numeric and the option entry column is not O or blank.	O is assumed.
124	I	REC IDENTIFYING IND COL 19-20 INVALID. BLK ASSUM	Record Identifying Indicator entry not 01-99.	Blanks are assumed.
125	I	STKR SEL COL 42 INVALID. BLK ASSUM	Stacker Select entry is: a) not 1, 2, or blank, b) specified with 2 I/O areas, or c) is invalid with the reader specified.	Blank is assumed.
*126	I	INVALID INPUT FILE	Input file has been specified as I, C, or U in col 15 of File Description Specification and no input specs are found for that file. The file was not defined on an Extension Specification.	No immediate action taken.
*127	I	POSITION ENTRY COL 21-24, 28-31, 35-38 INVALID	Position entry contains a non-numeric character.	0 is assumed.
128	I	'NOT' ENTRY COL 25, 32 OR 39 INVALID. N ASSUM	'NOT' entry not N or blank.	N is assumed.
129	I	C/Z/D ENTRY COL 26, 33 OR 40 INVALID. C ASSUM	Combined/Zone/Digit entry is not C, Z, or D.	C is assumed.
*130	I	FIELD NAME SPEC OUT OF SEQ	Field Name Type specification is first in deck, after invalid file name or invalid AND or OR spec.	Specification is dropped.
*131	I	FLD NAME COL 53-58 INVALID	Field Name entry is not left-justified.	Specification is dropped.
*132	I	FLD NAME COL 53-58 INVALID	Field Name entry beings with a non alpha character.	Specification is dropped.
*133	I	FROM OR TO COL 44-51 INVALID	FROM or TO is blank.	0001 is assumed.
*134	I	FROM OR TO COL 44-51 INVALID	FROM or TO entry contains a non-numeric character.	0 is assumed.
*135	I	TO COL 48-51 LESS THAN FROM COL 44-47	Defined field length less than 1.	1 is assumed.

•Table 15.1 RPG Compiler Error Notes (Part 9 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*136	I	PACKED INPUT FLD INVALID	Packed input field length defined by FROM and TO fields is greater than 8, or packed field is invalid for input device.	8 is assumed.
137	I	PACKED ENTRY COL 43 INVALID. P ASSUM	Packed entry is not P or blank.	P is assumed.
*138	I	DEC POS COL 52 INVALID	Decimal Positions is not numeric.	0 is assumed.
*139	I	NUMERIC FLD GT 14	Numeric field length is greater than 14 character.	Field length of 14 is assumed.
*140	I	CTRL LEV COL 59-60 INVALID	Column 59 not L.	L in column 59 is assumed.
*141	I	CTRL LEV COL 59-60 INVALID	Column 60 not numeric.	1 in column 60 is assumed.
*142	I	MATCH OR CHAIN ENTRY COL 61-62 INVALID	Column 61 not M or C.	M in column 61 is assumed.
*143	I	MATCH OR CHAIN ENTRY COL 61-62 INVALID	Column 62 is not numeric.	1 in column 62 is assumed.
*144	I	MATCH ENTRY COL 61-62 NOT M1-M9	Match entry is invalid.	M9 is assumed.
145	I	RSLT IND COL 65-68 SPECIFIED FOR NON-NUM FLD. IND IGN	Plus and Minus indicators cannot be used with an alphameric field.	Indicator is ignored.
*146		ALPHA FLD GT 256	Alphameric field length is more than 256 characters.	Field length of 256 is assumed.
*147	I	STERL FLD INVALID	Sterling field has more than 3 decimal positions specified.	3 is assumed.
*148	I	STERL FLD INVALID	Sterling field has no decimal positions specified.	0 is assumed.
149	I	REC ID SPEC OUT OF SEQ OR NO FIELDS FOR GIVEN REC	Warning only. Record ID spec is out of order, or no fields are indicated for a given record.	No immediate action taken.
*150	I	PACKED FLD MUST BE NUMERIC	Decimal Position entry col 52 is blank.	0 is assumed.
*151	I	FROM TO OR RECORD ID ZERO	FROM, TO, or Position entries are zero.	0001 is assumed.

• Table 15.1 RPG Compiler Error Notes (Part 10 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*155	F	KEY SIZE EXCEEDS REC LNG	Key length col 29-30 (ISAM file) is greater than record length.	No immediate action taken.
*158	F	KEY LNG EXCEEDS 50	Key length col 29-30 (ISAM file) is more than 50 characters.	50 is assumed.
*159		FLD NAME BEGINS WITH 'TAB' BUT IS NOT TBL NAME	Field name beginning with TAB is not a table name. Tables are defined on Extension Specification col 27-32.	Specification is dropped.
*160		FORM TYPE COL 6 INVALID	Next Form Type entry should have been O.	Specification is dropped.
*161	O	INVALID OUTPUT SPEC	Col 6 of spec contains an O but col 7 does not have * or start of filename. There is no H/D/T/E specified in col. 15. The spec is not an AND or OR.	Specification is dropped.
*162	O	FILENAME COL 7-14 INVALID	Filename entry is missing, improperly defined, or undefined.	Specification is dropped.
*163	O	H/D/T/E ENTRY COL 15 OUT OF SEQ	Output lines must be sequenced as follows: H/D/T/E.	Specification is dropped.
*164	O	LINE TYPE COL 15 INVALID	Line Type entry must be H, D, T, or E.	H is assumed.
165	O	IND COL 23-31 MISSING ON 'OR' SPEC, 00 ASSUM	'OR' spec requires conditioning indicators in col 23-31.	Indicator 00 is assumed.
166	O	IND COL 23-31 MISSING ON 'AND' SPEC. SPEC DROPPED	'AND' spec requires conditioning indicators in col 23-31.	Specification is dropped.
167	O	COL 32-70 MUST BE BLK ON LINE SPEC. BLK ASSUM	File ID and CONTROL specification requires col 32-70 blank.	Blanks are assumed.
168	O	FIELD NAME COL 32-37 INVALID. SPEC DROPPED	Field Name entry is not left-justified.	Specification is dropped.
*169	O	IND COL 23-25, 26-28, OR 29-31 INVALID	Output Indicator entry incorrect.	Blanks are assumed.

• Table 15.1 RPG Compiler Error Notes (Part 11 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*170	O	CARD OUT OF ORDER	'OR' or 'AND' card out of sequence.	Specification is dropped.
*171	O	CARD OUT OF ORDER	Field type spec with col 15 blank is not preceded by a valid line type spec.	Specification is dropped.
*172	O	OUTPUT FLD SPEC WITH ENTRIES IN COL 7-22	Output field spec requires col 7-22 blank.	Entries in columns 7-22 are ignored.
173	O	LEAD OR CLOSE QUOTE COL 45-70 MISSING. NO EDIT	Edit word must be enclosed by apostrophes.	No editing is performed.
174	O	EDIT CODE COL 38 INVALID OR USED WITH ALPHA FLD. BLK ASSUM	Edit code used is invalid or an edit code has been specified with an alpha field.	Blank is assumed.
175	O	BLANK AFTER COL 39 INVALID. BLK ASSUM	Blank after entry not B or blank.	Blank is assumed.
176	O	PACKED ENTRY COL 44 INVALID. BLK ASSUM	Packed entry not P or blank, field is not numeric or packed field is invalid.	Blank is assumed.
177	O	COL 17-22 NON-BLK ON 'AND' SPEC. BLK ASSUM	Col 17-22 not blank on 'AND' spec.	Blanks are assumed.
178	O	END POS COL 40-43 INVALID. SPEC DROPPED.	End Position in Output Record entry is blank, alpha, or is incompatible with constant or edit word.	Specification is dropped.
179	O	LEAD OR CLOSE QUOTE COL 45-70 MISSING. SPEC DROPPED	Constant must be enclosed by apostrophes.	Specification is dropped.
*180	C	FLD NAMED COL 43-48 GT 14	On an arithmetic operation, the field named in col 43-48 is longer than 14 characters.	Specification is dropped.
*181	C	MOVE ZONE OPER INVALID	Incorrect alphameric or numeric fields have been specified for this Move Zone operation. Only the low zone of a numeric field can be referred to.	Specification is dropped.
*183	C	FIELD NAME UNDEF.	The field name in Factor 1, Factor 2, or Result Field is undefined.	Specification is dropped.

•Table 15.1 RPG Compiler Error Notes (Part 12 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
184		FLD NAME UNREF	Warning only. Field Name entry is unreferenced field or table name.	No immediate action taken.
*185		FLD NAME MULT-DEF	Field name entry col 53-58 Input Spec, col 43-48 Calc Spec, or col 32-37 Output Spec contains a multi-defined field name. The field name has been defined as alpha and numeric or as same field type with different lengths or as numeric fields with different decimal positions.	No immediate action taken.
*186	C	ARITH OPER SPECIFIED WITH ALPHA FLD	Arithmetic operation specified in operation col 28-32 with an alphameric field specified in Factor 1, Factor 2, or Result field.	Specification is dropped.
*187	C	COMP OPER SPECIFIED WITH ALPHA AND NUM FLD	Alphameric and numeric field being compared, Compare operations are valid only between like fields.	Specification is dropped.
188	C	RSLT FLD LNG COL 49-51 MAY NO BE LARGE ENOUGH	Warning only. The result field may not be long enough to contain the true result.	No immediate action taken.
*189	C	FACT2 OR RSLT FLD NOT TBL NAME	LOKUP requires table names in Factor 2 col 33-42, and Result Field col 43-48 (if specified).	Specification is dropped.
*190	C	EXSR OPER CALLS ITSELF	Name in Factor 2 is the name of the subroutine of which the EXSR operation is a part (a subroutine may not call itself).	Specification is dropped.
*191	C	TESTZ OPER INVALID	Result Field entry col 43-48 is numeric. TESTZ tests for a high-order zone punch of an alpha field.	Specification is dropped.
192	C	GOTO AND TAG OPERS ARE NOT IN SAME CALC SECTION	Label of the TAG operation and the corresponding GOTO are not in Detail or Total calcs.	Specification is dropped.

•Table 15.1 RPG Compiler Error Notes (Part 13 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
193	C	HLF ADJ COL 53 IS IN-COMPATIBLE. BLK ASSUM	The number of positions of the arithmetic result is less than or equal to the specified decimal positions of the result field; therefore, half-adjust cannot be performed.	Blank is assumed.
*194	C	LOKUP OPER INVALID DUE TO UNEQUAL LNGS	Length of Factor 1 col 18-27 and Factor 2 col 33-42 are not equal.	Specification is dropped.
*196	C	MVR OPER NOT PRECEDED BY DIV	There is no remainder to move.	MVR operation is ignored.
*197	C	MVR OPER PRECEDED BY DIV WITH HLF ADJ	Half-adjust effectively removes any remainder.	MVR operation is ignored.
*198	C	LOKUP OPER SPECIFIED WITH ALPHA AND NUM FLD	Factor 1 col 18-27 and Factor 2 col 33-42 must both be alpha or numeric.	Specification is dropped.
*199	C	HIGH AND LOW RSLT IND SPEC FOR LOKUP OPER	High and Low Resulting Indicators are both specified for LOKUP operation.	Low indicator is ignored.
*200	F	NO PRIMARY FILE SPECIFIED	No P in col 16 of File Description spec. One file must be defined as primary.	Job is terminated.
*201		FORM TYPE COL 6 INVALID	Next Form Type entry should have been F, E, or I.	Specification is dropped.
*202	F	FILENAME COL 7-14 INVALID	Filename incorrectly specified.	Specification is dropped.
*203	F	MORE THAN 10 FILENAMES SPEC	Ten is the maximum number of files that can be processed in a program.	Only the first 10 are processed.
204	F	UNREF FILENAME	Warning only. A file defined on the File Description spec has not been used in the program	No immediate action taken.
205	F	O IN COL 15 INVALID WITH READ01	Device entry READ01 requires an I in file type col 15.	Specification is dropped.
*206	F	DEVICE COL 40-46 INVALID	Device name is unrecognizable.	Job is terminated.
207	F	FILENAME COL 7-14 MULT-REF	The filename is specified on the Input or Output-Format Specifications more than once.	No immediate action taken.

•Table 15.1 RPG Compiler Error Notes (Part 14 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*208	F	FILENAME COL 7-14 MULT-DEF	The same filename has been defined on two File Description specs.	Second specification is dropped
*210		NO IND OR ONLY PREDEF IND SPEC FOR INPUT REC	At least one indicator is required on Input Specs.	Job is terminated.
*212		UNDEFINED RESULT IND	Result indicator used but not defined.	No immediate action taken.
213		UNREFERENCED IND	Warning only. Indicator specified but not used.	No immediate action taken.
215	F	FILE DESCR SPEC WITH E COL 39 NOT REF ON EXT SPEC	File Description spec with E in col 39 is not used on an Extension spec.	No immediate action taken.
219	O	FLD NAME COL 32-37 UNDEFINED. SPEC DROPPED	Name must be defined on Input or Calc spec.	Specification is dropped.
*221	I	MATCH FLD LNGS INCOMPATIBLE	Sum of Matching Field lengths must be equal for all record types having matching records specified, or matching fields separated by fields conditioned on Field Record Relation indicators.	No immediate action taken.
*222	E	TBL NAME MULT-DEF	Same name used for two tables, or the table has been defined as alpha and numeric or as same type with two lengths or decimal positions.	No immediate action taken.
*224	I	SPLIT CHAIN FLDS IMPROPER	Split chain fields improperly specified.	No immediate action taken.
*225	I	SPLIT CTRL FLDS IMPROPER	Split control fields improperly specified.	No immediate action taken.
*226	I	SPLIT MATCH FLDS INVALID	Split matching fields are not allowed.	No immediate action taken.
*227	I	MATCH FLD LNGS INCOMPATIBLE	All match fields of the same level must be the same length on all record types.	No immediate action taken.
*228	I	CTRL FLD LNG INCOMPATIBLE	The control field on a given control level must be the same length for all record types.	No immediate action taken.

•Table 15.1 RPG Compiler Error Notes (Part 15 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*229	I	CHAIN FLD LNG INCOMPATIBLE	All fields using the same chaining indicator must be the same length on all record types.	No immediate action taken.
*230	I	CTRL FLD LNG GT 247	The sum of the control fields on all levels used on a record type cannot exceed 247 characters.	No immediate action taken.
*231	I	FLD AREA GT REC SIZE	Input field area size exceeds input record length.	No immediate action taken.
232	O	PRINTER FILE BLK COL 17-22. SPACE 1 AFTER ASSUM	Entry required in col 17-22 for printer carriage control.	Single space after is assumed.
233	O	STKR SEL COL 16 INVALID. BLK ASSUM	Stacker select invalid for output device, or entry is incorrect (not 1 or 2).	Blank is assumed.
234	O	SPACE BEFORE, COL 17, INVALID. 1 ASSUM	There is an entry in column 17 but it is not 0, 1, 2, or 3.	Single space before is assumed.
235	O	SPACE AFTER, COL 18, INVALID. 1 ASSUM	There is an entry in col 18 but it is not 0, 1, 2, or 3.	Single space after is assumed.
236	O	SKIP BEFORE, COL 19-20, INVALID. BLK ASSUM	There is an entry in col 19-20 but it is not 01-12 or with an 1132 Printer the skip is to channel 7, 8, 10, or 11.	Blanks are assumed.
237	O	SKIP AFTER, COL 21-22, INVALID. BLK ASSUM	There is an entry in col 21-22 but it is not 01-12 or with an 1132 Printer the skip is to channel 7, 8, 10, or 11.	Blanks are assumed.
238	O	PACKED FLD COL 44 NOT NUM. BLK ASSUM	Output field is alpha.	Blank is assumed.
239	O	EDIT CODE COL 38 SPECIFIED ON ALPHA FLD. BLK ASSUM	Alpha fields cannot be edited with an edit code.	Blank is assumed.
240	O	STERL SPECIFIED ON NON-NUM FLD. NO STERL ASSUM	Sterling option col 71-74 requested for alpha field.	No sterling is assumed
*241	O	EDIT WD TOO SMALL	Edit word is too small for field.	No immediate action taken.

•Table 15.1 RPG Compiler Error Notes (Part 16 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*242	O	EDIT FLD NOT NUM	Alpha fields cannot be edited.	No immediate action taken.
*243	O	DOLLAR SIGN INVALID	Both fixed and floating dollar sign have been specified.	No immediate action taken.
*244	O	BOTH CR AND - USED	Both CR and - (minus) used for credit.	No immediate action taken.
*245	O	OUTPUT SPEC INVALID	Output specs are missing or are invalid for this program.	Job is terminated.
*246	O	PAGE FLD IS DEF AS ALPHA	PAGE defined on Input spec with no dec pos in col 52 of Input spec.	No immediate action taken.
*247	O	FLD LNG GT END POS COL 40-43.	Output field length is greater than the End Position in Output Record col 40-43 indicates.	No immediate action taken.
*248	F	INDEX SEQ FILE ADDITION COL 66 INVALID	Col 66 must contain an A for ISAM ADD functions.	No immediate action taken.
*250	F	INDEX SEQ KEY LNG COL 29-30 INVALID	Key Length entry col 29-30 is not numeric.	8 is assumed.
251	F	INDEX SEQ KEY START POS COL 35-38 INVALID. 1 ASSUM	Key field must start in position one of record.	0001 is assumed.
*252	O	END POS GT RCD LNG	Output field length is greater than Record length (Col. 40-43).	No immediate action taken.
*254	O	'ADD' COL 16-18 MUST BE SPEC	'ADD' must be specified if records are to be added to an ISAM file.	Specification is dropped.
*255	C	FACT2 COL 33-42 INVALID	Entry in Factor 2 must be filename described as a chained file on the File Description spec.	No immediate action taken.
256	C	CTRL LEV COL 7-8 INVALID. SR ASSUM	Closed subroutine must follow total calcs.	SR is assumed.
*257	C	ERROR IN SEQ OF ENDSR-BEGSR	BEGSR operation must come first.	No immediate action taken.

•Table 15.1 RPG Compiler Error Notes (Part 17 of 17)

NOTE	SPEC TYPE	ERROR MESSAGE	CAUSE OF ERROR	RESULT
*258	C	BEGSR OR EXSR FACTORS INVALID	BEGSR-Subroutine name must appear in Factor 1 col 18-27. EXSR-Subroutine name must appear in Factor 2 col 33-42.	No immediate action taken.
259	C	COL 49-59 MUST BE BLK WITH EXSR OR EXCPT. BLK ASSUM	EXSR or EXCPT op codes require col 49-59 blank.	Blanks are assumed.
260	C	COL 9-17 MUST BE BLK WITH BEGSR. BLK ASSUM	BEGSR op code requires col 9-17 blank.	Blanks are assumed.
262	C	COL 49-53 MUST BE BLK WITH CHAIN. BLK ASSUM	CHAIN op code requires col 49-53 blank.	Blanks are assumed.
263	C	IND COL 56-57 MUST BE THE SAME AS IND COL 54-55. HIGH ASSUM	The same indicator must be specified as high and low indicator.	High indicator assumed for high and low.
264	O	CHAIN SPECIFIED WITH IND IN COL 58-59. BLK ASSUM	Equal indicator cannot be specified on chaining operation.	Blanks are assumed.
*265	C	PAGE FLD INVALID	Page field must be numeric. Field length must be 4 with zero decimal positions.	Field length of 4 and zero decimal positions assumed.
270	O	SKIP INVALID FOR CONSOLE PRINTER. BLK ASSUM	Console Printer has not provisions for forms skipping. Col 19-22 must be blank.	Blanks are assumed.

Table 16. Core Load Builder Error Messages

Error Number and Message		Cause and Corrective Action
R00	LOCALS/SOCALS OVERFLOW WORK STORAGE	There is insufficient Working Storage remaining to accommodate the LOCAL and/or SOCAL overlays required by the core load. Remedy Change the Working Storage ID on the JOB card to the drive with the most available Working Storage or Create more Working Storage on the present drive by deleting subroutines, subprograms, and/or data no longer required.
R01	ORIGIN BELOW 1ST WORD OF MAINLINE	The Core Load Builder has been instructed to load a word into an address lower than the first word of the mainline program. Remedy Remove the ORG statement that is causing the problem or, Origin the mainline program at a lower address.
R02	DEFINE FILE(S) OVERFLOW WORK STORAGE	There is insufficient Working Storage remaining to accommodate even one record of the defined file(s). Remedy See R00.
R03	NO DSF PROGRAM IN WORKING STORAGE	No program in Working Storage. Remedy Place the desired program in Working Storage before calling the Core Load Builder.
R05	INVALID LOADING ADDR FOR ILS02	ILS02 has been loaded into low COMMON. Remedy The mainline should be made longer so that ILS02 will be loaded in a higher address.
R06	FILE(S) TRUNCATED (SEE FILE MAP)	At least one defined file has been truncated, either because the previously defined storage area in the User or Fixed Area was inadequate, or because there is inadequate Working Storage available to store the file. Remedy Redefine the User/Fixed Area file, or change the record count specification in the DEFINE FILE statement.
R07	TOO MANY ENTRIES IN LOAD TABLE	There are references to more than (approximately) 375 different entry points in the core load by CALL and/or LIBF statements. Remedy Divide the core load into two or more links.
R08	CORE LOAD EXCEEDS 32K	The Core Load Builder has been instructed to load a word into a core address that exceeds 32767 (a negative number). The loading process is immediately terminated, since the Core Load Builder cannot process negative addresses. The error was probably caused by bad data being read from the disk.
R09	LIBF TV REQUIRES 82 OR MORE ENTRIES	There are at least 82 different entry points referenced in the core load by LIBF statements. Remedy Divide the core load into two or more links.
R16	XXXXX IS NOT IN LET OR FLET	The program or data file name printed in the message cannot be found in LET or FLET. Remedy Store the program or data file. If the name cannot be explained, the program being loaded has probably been destroyed (bad data was read from the disk).
R17	XXXXX CANNOT BE A LOCAL/NOCAL	The program named in this message is either a type which cannot appear on a *LOCAL control card, or is a LOCAL that has been referenced, directly or indirectly, by another LOCAL.
R18	XXXXX LOADING HAS BEEN TERMINATED	The loading of the mainline program named in this message has been terminated as a result of the errors listed in the messages preceding this one.
R19	XXXXX IS NOT A DATA FILE	The area named in this message does not begin at a sector boundary, which implies that it is not a data file but a DSF program, and thus a possible error. Remedy Choose another area for the storage of this file.
R20	XXXXX COMMON EXCEEDS THAT OF ML	The length of COMMON for the subroutine named in this message is longer than that of the mainline program. Remedy Define more COMMON for the mainline program.
R21	XXXXX PRECISION DIFFERENT FROM ML	The precision for the subroutine named in this message is incompatible with that of the mainline program. Remedy Make the precisions compatible.

Table 16. Core Load Builder Error Messages (continued)

Error Number and Message		Cause and Corrective Action
R22	XXXXX AND ANOTHER VERSION REF'ENCED	At least two different versions of the same ISS have been referenced, e.g., CARDZ and CARD0 (FORTRAN uses CARDZ). If a disk subroutine is named in the message, it is possible that the XEQ record specifies one version (e.g., DISKZ) whereas the program references another (e.g., DISKN). (A blank in column 19 of the XEQ control record causes DISKZ to be used.) Remedy Change the references so that the core load uses only one version of any given I/O subroutine.
R23	XXXXX SHOULD BE IN THE FIXED AREA	The area named in this message is in the User Area. References in DEFINE FILE and DSA statements for *STORECI functions must be to the Fixed Area.
R39	XXXX IS NOT CURRENTLY ON SYSTEM	XXXX is a cartridge ID appearing on an *FILES record. The cartridge was not on-line when the core load was built.
R40	XXXX(HEX) = ADDITIONAL CORE REQUIRED	If the core load was executed, /XXXX is the number of words by which it exceeded core before the Core Load Builder made it fit by creating special overlays (SOCALs). If the core load was not executed, /XXXX equals the number of words still required after the Core Load Builder has attempted to make it fit by using SOCALs. Remedy In the latter case, create more links or LOCALs.
R41	XXXX(HEX) WORDS UNUSED BY CORE LOAD	NOT AN ERROR. /XXXX is the number of words of core storage not used by this core load.
R42	XXXX(HEX) IS THE EXECUTION ADDR	NOT AN ERROR. This message follows every successful conversion from DSF to DCI when a core map has been requested.
R43	XXXX(HEX) = ARITH/FUNC SOCAL WD CNT	NOT AN ERROR. It has been necessary to use special overlays (SOCALs) and /XXXX is the length of the arithmetic/function overlay (see System Overlays).
R44	XXXX(HEX) = FI/O, I/O SOCAL WD CNT	NOT AN ERROR. It has been necessary to use special overlays (SOCALs), and /XXXX is the length of the FORTRAN I/O, I/O, and conversion subroutine overlay (see System Overlays).
R45	XXXX(HEX) = DISK FI/O SOCAL WD CNT	NOT AN ERROR. It has been necessary to use special overlays (SOCALs), and /XXXX is the length of the disk FORTRAN I/O overlay, including the 320 word buffer.
R46	XXXX(HEX) = AN ILLEGAL ML LOAD ADDR	/XXXX is the address at which the Core Load Builder has been requested to start loading the mainline program. However, this address is lower than the highest address occupied by the version of disk I/O requested for this core load. If DISKZ has been requested, this message indicates an absolute mainline program starting at an odd location. An ORG to an even location, followed by a BSS of an odd number of words has the same effect as an ORG to an odd location. Remedy ● Origin the mainline at a higher address, or ● Request a shorter version of disk I/O, or ● Origin the mainline at an even boundary
R47	XXXX(HEX) TOO MANY WDS IN COMMON	The length of COMMON specified in the mainline program plus the length of the core load exceeds core storage by /XXXX words.
R64	XXXXX IS BOTH A LIBF AND A CALL	The subroutine named in this message either been improperly referenced, i.e., CALL instead of LIBF or vice versa, or has been referenced in both CALL and LIBF statements.
R65	XXXXX HAS MORE THAN 14 ENTRY POINTS	This message usually means that the subroutine has been destroyed since a subroutine is not stored if it contains more than 14 entry points.
R66	XXXXX HAS AN INVALID TYPE CODE	The subroutine named in this message has either ● 1) Been designated on an XEQ record and is not a mainline program or, 2) ● Contains a type code other than 3 (subroutine), 4 (function), 5 (ISS), or 6 ILS), in which case, the subroutine has probably been destroyed. This error can also be caused by a DSA statement referencing a DSF program, or a CALL or LIBF referencing a program in DCI or DDF.
R67	XXXX HAS AN INVALID GSB ADDRESS	The subroutine named has a Graphic Short Branch order address that is larger than 8191 after relocation.

AUXILIARY SUPERVISOR ERRORS

The Auxiliary Supervisor prints no informational messages. Table 17 lists Auxiliary Supervisor Error messages.

Table 17. Auxiliary Supervisor Errors

Error Number and Message	Cause of Error
S00 INVALID FUNCTION CODE	The Auxiliary Supervisor received an illegal parameter.
S01 XXXXX IS NOT IN LET/FLET	The Core Image Loader is unable to find the specified name in LET or FLET.
S02 XXXXX IS A DATA FILE	The specified name cannot be executed since it is a data file, not a program.

ISS SUBROUTINE WAITS

A device not ready or illegal function parameter causes a pre-operative WAIT at $PRET. The ISS subroutine WAITs are listed in Table 18.

Table 18. ISS Subroutine WAITs

Device Causing WAIT	Contents of Accumulator	Cause of WAIT
1442 Card Read Punch or 1442 Card Punch	/1000 /1001	Device not ready or last card indicator on for read. Illegal device, device not in system, illegal function, word count over +80, or word count zero or negative.
Keyboard/Console Printer	/2000 /2001 /2002	Device not ready. Device not in system, illegal function, or word count zero or negative. Keyboard input expected (TYPEZ only).
1134/1055 Paper Tape Reader/Punch	/3000 /3001	Device not ready. Illegal device, illegal function, word count zero or negative, or illegal check digit.
2501 Card Reader	/4000 /4001	Device not ready. Illegal function, word count over +80, or word count zero or negative.
Disk	/5000 /5001 /5002 /5003 /5004	Device not ready. Illegal device, device not in system, illegal function, attempt to write in file protected area, word count zero or negative, or starting sector identification over +1599. Write select/power unsafe. Same as /5001 except error caused by a Monitor program (DISK1, (DISKN only). Disk error (DISKZ only).
1132 Printer	/6000 /6001	Device not ready or end of forms. Illegal function, word count over +60, or word count zero or negative.
1627 Plotter	/7000 /7001	Device not ready. Illegal device, device not in system, illegal function, or word count zero or negative.
SCA	/8001 /8002 /8003	Invalid function code or invalid word count (all SCAT subroutines). Invalid sub-function code for some transmit or receive operation (SCAT2 or SCAT3 only). Receive operation not completed or transmit operation not completed (SCAT1 only). Failure to establish synchronization before attempting to perform some transmit or receive operation, or attempting to receive before receiving INQ sequence (SCAT1 only).
1403 Printer	/9000 /9001 /9002	Device not ready or end of forms. Illegal function, word count over +60, or word count zero or negative. Parity check, Scan Check or Sync Check. Press start to proceed.
1231 Optical Mark Page Reader	/A000 /A001 /A002 /A003	Device not ready. Illegal function. Feed check, last document processed. Clear jam, make ready, do not refeed. Feed check, last document processed. Clear jam, make ready, refeed last document. If error was caused by double feed, refeed both documents.

RPG OBJECT PROGRAM ERRORS

RPG object program errors cause the system to WAIT with CXXX in the accumulator. Object program errors are listed in Table 18.1.

The object program errors can be divided into two categories, disk I/O and general purpose. Disk I/O errors are in the range C000 to C05F. All others are between C100 and CFFF. Some of the disk I/O errors should not occur during normal processing. However, if incorrect object code is generated, or the object program is erroneously modified at object time, these errors may occur. These errors are identified with an asterisk to the right of the CXXX number in Table 18.1.

When an RPG object program error occurs, the system operator must take specific action. Generally this amounts to aborting the job by turning all console entry switches off and pressing console START. Certain errors however allow the operator to ignore the error or retry the operation. This is accomplished by setting console entry switch 15 on, all others off, and pressing console START. In the case of a retry, the card in error must be placed back in the hopper before continuing.

Incorrect operator action will cause the error WAIT to reoccur.

•Table 18.1 (Part 1 of 5)

Accumulator Display	Type of Processing	Meaning	Action Required	Console Entry Switch Settings
C000	Sequential file: random processing.	Record number is not within the assigned limits of the file.	Terminate the job; or, bypass the record and continue processing; or, if chaining, correct the card and reinsert it in the input stream, or bypass the chaining record and read the next card.	All off. Press console START. 15 on, all others off. Press console START. 15 on, all others off. Press console START.
C001*	Sequential file: random processing.	Record size not within limits (maximum 640 characters).	Terminate the job.	All off. Press console START.
C002*	Sequential file: random processing.	Records per sector not maximum.	Terminate the job.	All off. Press console START.
C003	Sequential file: random processing.	No record found. The record number is not a positive number.	Terminate the job; or, bypass the record and continue processing; or, if chaining, correct the card and reinsert it in the input stream, or bypass the chaining record and read the next card.	All off. Press console START. 15 on, all others off. Press console START. 15 on, all others off. Press console START.
C004	Sequential file: random processing.	Write before read on an update file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C005*	Sequential file: random processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C006*	Sequential file: random processing.	I/O buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C010	Sequential file: sequential processing.	Disk file is full.	Terminate the job.	All off. Press console START.

•Table 18.1 (Part 2 of 5)

Accumulator Display	Type of Processing	Meaning	Action Required	Console Entry Switch Settings
C011*	Sequential file: sequential processing.	A write was requested on an input file.	Terminate the job.	All off. Press console START.
C012*	Sequential file: sequential processing.	A read was requested on an output file.	Terminate the job.	All off. Press console START.
C013*	Sequential file: sequential processing.	Record size not within limits (maximum 640 characters).	Terminate the job.	All off. Press console START.
C014*	Sequential file: sequential processing.	Number of records per sector not maximum.	Terminate the job.	All off. Press console START.
C015*	Sequential file: sequential processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C016*	Sequential file: sequential processing.	I/O buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C017	Sequential file: sequential processing.	Write before read on an update file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on. All others off. Press console START.
C020	ISAM load processing.	Invalid type of processing on load function.	Terminate the job.	All off. Press console START.
C021*	ISAM load processing.	Record size not within limits (maximum 636 characters) or, number of records per sector not maximum.	Terminate the job.	All off. Press console START.
C022*	ISAM load processing.	Key length greater than maximum	Terminate the job.	All off. Press console START.
C023*	ISAM load processing.	Index entry length not same as length computed from key length.	Terminate the job.	All off. Press console START.

•Table 18.1 (Part 3 of 5)

Accumulator Display	Type of Processing	Meaning	Action Required	Console Entry Switch Settings
C024*	ISAM load processing.	Number of index entries per sector does not permit maximum number of records per sector.	Terminate the job.	All off. Press console START.
C025	ISAM load processing.	Prime data area is full.	Terminate the job.	All off. Press console START.
C026	ISAM load processing.	Index area is full.	Terminate the job.	All off. Press console START.
C027*	ISAM load processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C028*	ISAM load processing	Index buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C029*	ISAM load processing.	Prime data buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C02A	ISAM load processing.	Input record out of sequence.	Terminate the job; or, correct the card and reinsert it in the input stream, or bypass the record by reading another card.	All off. Press console START. 15 on, all others off. Press console START.
C030*	ISAM add processing.	Invalid type of processing on add function.	Terminate the job.	All off. Press console START.
C031*	ISAM add processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C032	ISAM add processing.	Key length for this job not same as key length in file.	Terminate the job.	All off. Press console START.
C033	ISAM add processing.	Record length for this job not same as record length in file.	Terminate the job.	All off. Press console START.
C034	ISAM add processing.	Attempt was made to add record already on file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.

•Table 18.1 (Part 4 of 5/

Accumulator Display	Type of Processing	Meaning	Action Required	Console Entry Switch Settings
C035	ISAM add processing.	Overflow area is full.	Terminate the job.	All off. Press console START.
C036*	ISAM add processing.	Index buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C040*	ISAM file: sequential processing.	Invalid type of processing on retrieve or update function.	Terminate the job.	All off. Press console START.
C041*	ISAM file: sequential processing.	Index buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C042*	ISAM file: sequential processing.	Prime data buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C043	ISAM file: sequential processing.	Key length for this job not same as key length in file.	Terminate the job.	All off. Press console START.
C044	ISAM file: sequential processing.	Record length for this job not same as record length in file.	Terminate the job.	All off. Press console START.
C045*	ISAM file: sequential processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C046	ISAM file: sequential processing.	Write before read on update file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C050*	ISAM file: random processing.	Invalid type of processing on retrieve or update function.	Terminate the job.	All off. Press console START.
C051*	ISAM file: random processing.	Index buffer not on even-word boundary.	Terminate the job.	All off. Press console START.
C052*	ISAM file: random processing.	Prime data buffer not on oven-word boundary.	Terminate the job.	All off. Press console START.
C053	ISAM file: random processing.	Key length for this job not same as key length in file.	Terminate the job.	All off. Press console START.

•Table 18.1 (Part 5 of 5)

Accumulator Display	Type of Processing	Meaning	Action Required	Console Entry Switch Settings
C054	ISAM file: random processing.	Record length for this job not same as record length in file.	Terminate the job.	All off. Press console START.
C055*	ISAM file: random processing.	File accessed when not open.	Terminate the job.	All off. Press console START.
C056	ISAM file: random processing.	Write before read on update.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C057	ISAM file: random processing.	Record not on file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C111		Numeric records or matching fields out of sequence, or record is an undefined type.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C12n		Halt switch set by object program (n=1-9)	Terminate the job; or, set the halt switches off and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C400		Write before read on combined file.	Terminate the job; or, bypass the record and continue processing.	All off. Press console START. 15 on, all others off. Press console START.
C430		Attempt to divide by zero.	Terminate the job; or, continue processing. The quotient will be set to zero.	All off. Press console START. 15 on, all others off. Press console START.
C998		Table fields out of sequence.	Terminate the job.	All off. Press console START.

Ref No.	EBCDIC Binary 0123	EBCDIC Binary 4567	EBCDIC Hex	IBM Card Code Rows 12	11	0	9	8	7-1	IBM Card Code Hex	Graphics and Control Names	1132 Printer EBCDIC Subset Hex	PTTC/8 Hex U-Upper Case L-Lower Case	Console Printer Hex Notes	1403 Printer Hex
0	0000	0000	00	12		0	9	8	1	B030	NUL				
1		0001	01	12			9		1	9010					
2		0010	02	12			9		2	8810					
3		0011	03	12			9		3	8410					
4		0100	04	12			9		4	8210	PF Punch Off				
5*		0101	05	12			9		5	8110	HT Horiz.Tab		6D (U/L)	41 ①	
6*		0110	06	12			9		6	8090	LC Lower Case		6E (U/L)		
7*		0111	07	12			9		7	8050	DEL Delete		7F (U/L)		
8		1000	08	12			9	8		8030					
9		1001	09	12			9	8	1	9030					
10		1010	0A	12			9	8	2	8830					
11		1011	0B	12			9	8	3	8430					
12		1100	0C	12			9	8	4	8230					
13		1101	0D	12			9	8	5	8130					
14		1110	0E	12			9	8	6	80B0					
15		1111	0F	12			9	8	7	8070					
16	0001	0000	10	12	11		9	8	1	D030					
17		0001	11		11		9		1	5010					
18		0010	12		11		9		2	4810					
19		0011	13		11		9		3	4410					
20*		0100	14		11		9		4	4210	RES Restore		4C (U/L)	05 ②	
21*		0101	15		11		9		5	4110	NL New Line		DD(U/L)	81 ③	
22*		0110	16		11		9		6	4090	BS Backspace		5E (U/L)	11	
23		0111	17		11		9		7	4050	IDL Idle				
24		1000	18		11		9	8		4030					
25		1001	19		11		9	8	1	5030					
26		1010	1A		11		9	8	2	4830					
27		1011	1B		11		9	8	3	4430					
28		1100	1C		11		9	8	4	4230					
29		1101	1D		11		9	8	5	4130					
30		1110	1E		11		9	8	6	40B0					
31		1111	1F		11		9	8	7	4070					
32	0010	0000	20		11	0	9	8	1	7030					
33		0001	21			0	9		1	3010					
34		0010	22			0	9		2	2810					
35		0011	23			0	9		3	2410					
36		0100	24			0	9		4	2210	BYP Bypass				
37*		0101	25			0	9		5	2110	LF Line Feed		3D (U/L)	03	
38*		0110	26			0	9		6	2090	EOB End of Block		3E (U/L)		
39		0111	27			0	9		7	2050	PRE Prefix				
40		1000	28			0	9	8		2030					
41		1001	29			0	9	8	1	3030					
42		1010	2A			0	9	8	2	2830					
43		1011	2B			0	9	8	3	2430					
44		1100	2C			0	9	8	4	2230					
45		1101	2D			0	9	8	5	2130					
46		1110	2E			0	9	8	6	20B0					
47		1111	2F			0	9	8	7	2070					
48	0011	0000	30	12	11	0	9	8	1	F030					
49		0001	31				9		1	1010					
50		0010	32				9		2	0810					
51		0011	33				9		3	0410					
52		0100	34				9		4	0210	PN Punch On				
53*		0101	35				9		5	0110	RS Reader Stop		0D(U/L)	09 ④	
54*		0110	36				9		6	0090	UC Upper Case		0E (U/L)		
55		0111	37				9		7	0050	EOT End of Trans.				
56		1000	38				9	8		0030					
57		1001	39				9	8	1	1030					
58		1010	3A				9	8	2	0830					
59		1011	3B				9	8	3	0430					
60		1100	3C				9	8	4	0230					
61		1101	3D				9	8	5	0130					
62		1110	3E				9	8	6	00B0					
63		1111	3F				9	8	7	0070					

NOTES: Typewriter Output

① Tabulate
② Shift to black
③ Carrier Return
④ Shift to red

* Recognized by all Conversion subroutines
Codes that are not asterisked are recognized only by the SPEED subroutine

Ref No.	EBCDIC Binary 0123	EBCDIC Binary 4567	EBCDIC Hex	IBM Card Code Rows 12	11	0	9	8	7-1	IBM Card Code Hex	Graphics and Control Names	1132 Printer EBCDIC Subset Hex	PTTC/8 Hex U-Upper Case L-Lower Case	Console Printer Hex	1403 Printer Hex
64*	0100	0000	40	no punches						0000	blank	40	10 (U/L)	21	7F
65		0001	41	12		0	9		1	B010					
66		0010	42	12		0	9		2	A810					
67		0011	43	12		0	9		3	A410					
68		0100	44	12		0	9		4	A210					
69		0101	45	12		0	9		5	A110					
70		0110	46	12		0	9		6	A090					
71		0111	47	12		0	9		7	A050					
72		1000	48	12		0	9	8		A030					
73		1001	49	12				8	1	9020					
74*		1010	4A	12				8	2	8820	¢		20 (U)	02	
75*		1011	4B	12				8	3	8420	. (period)	4B	6B (L)	00	6E
76*		1100	4C	12				8	4	8220	<		02 (U)	DE	
77*		1101	4D	12				8	5	8120	(	4D	19 (U)	FE	57
78*		1110	4E	12				8	6	80A0	+	4E	70 (U)	DA	6D
79*		1111	4F	12				8	7	8060	| (logical OR)		3B (U)	C6	
80*	0101	0000	50	12						8000	&	50	70 (L)	44	15
81		0001	51	12	11		9		1	D010					
82		0010	52	12	11		9		2	C810					
83		0011	53	12	11		9		3	C410					
84		0100	54	12	11		9		4	C210					
85		0101	55	12	11		9		5	C110					
86		0110	56	12	11		9		6	C090					
87		0111	57	12	11		9		7	C050					
88		1000	58	12	11		9	8		C030					
89		1001	59		11			8	1	5020					
90*		1010	5A		11			8	2	4820	!		5B (U)	42	
91*		1011	5B		11			8	3	4420	$	5B	5B (L)	40	62
92*		1100	5C		11			8	4	4220	*	5C	08 (U)	D6	23
93*		1101	5D		11			8	5	4120	)	5D	1A (U)	F6	2F
94*		1110	5E		11			8	6	40A0	;		13 (U)	D2	
95*		1111	5F		11			8	7	4060	¬ (logical NOT)		6B (U)	F2	
96*	0110	0000	60		11					4000	- (dash)	60	40 (L)	84	61
97*		0001	61			0			1	3000	/	61	31 (L)	BC	4C
98		0010	62		11	0	9		2	6810					
99		0011	63		11	0	9		3	6410					
100		0100	64		11	0	9		4	6210					
101		0101	65		11	0	9		5	6110					
102		0110	66		11	0	9		6	6090					
103		0111	67		11	0	9		7	6050					
104		1000	68		11	0	9	8		6030					
105		1001	69			0		8	1	3020					
106		1010	6A	12	11					C000					
107*		1011	6B			0		8	3	2420	, (comma)	6B	3B (L)	80	16
108*		1100	6C			0		8	4	2220	%		15 (U)	06	
109*		1101	6D			0		8	5	2120	_ (underscore)		40 (U)	BE	
110*		1110	6E			0		8	6	20A0	>		07 (U)	46	
111*		1111	6F			0		8	7	2060	?		31 (U)	86	
112	0111	0000	70	12	11	0				E000					
113		0001	71	12	11	0	9		1	F010					
114		0010	72	12	11	0	9		2	E810					
115		0011	73	12	11	0	9		3	E410					
116		0100	74	12	11	0	9		4	E210					
117		0101	75	12	11	0	9		5	E110					
118		0110	76	12	11	0	9		6	E090					
119		0111	77	12	11	0	9		7	E050					
120		1000	78	12	11	0	9	8		E030					
121		1001	79					8	1	1020					
122*		1010	7A					8	2	0820	:		04 (U)	82	
123*		1011	7B					8	3	0420	#		0B (L)	C0	
124*		1100	7C					8	4	0220	@		20 (L)	04	
125*		1101	7D					8	5	0120	' (apostrophe)	7D	16 (U)	E6	0B
126*		1110	7E					8	6	00A0	=	7E	01 (U)	C2	4A
127*		1111	7F					8	7	0060	"		0B (U)	E2	

Ref No.	EBCDIC Binary 0123	EBCDIC Binary 4567	EBCDIC Hex	IBM Card Code Rows 12	11	0	9	8	7-1	IBM Card Code Hex	Graphics and Control Names	1132 Printer EBCDIC Subset Hex	PTTC/8 Hex U-Upper Case L-Lower Case	Console Printer Hex	1403 Printer Hex
128	1000	0000	80	12		0		8	1	B020					
129		0001	81	12		0			1	B000	a				
130		0010	82	12		0			2	A800	b				
131		0011	83	12		0			3	A400	c				
132		0100	84	12		0			4	A200	d				
133		0101	85	12		0			5	A100	e				
134		0110	86	12		0			6	A080	f				
135		0111	87	12		0			7	A040	g				
136		1000	88	12		0		8		A020	h				
137		1001	89	12		0	9			A010	i				
138		1010	8A	12		0		8	2	A820					
139		1011	8B	12		0		8	3	A420					
140		1100	8C	12		0		8	4	A220					
141		1101	8D	12		0		8	5	A120					
142		1110	8E	12		0		8	6	A0A0					
143		1111	8F	12		0		8	7	A060					
144	1001	0000	90	12	11			8	1	D020					
145		0001	91	12	11				1	D000	j				
146		0010	92	12	11				2	C800	k				
147		0011	93	12	11				3	C400	l				
148		0100	94	12	11				4	C200	m				
149		0101	95	12	11				5	C100	n				
150		0110	96	12	11				6	C080	o				
151		0111	97	12	11				7	C040	p				
152		1000	98	12	11			8		C020	q				
153		1001	99	12	11		9			C010	r				
154		1010	9A	12	11			8	2	C820					
155		1011	9B	12	11			8	3	C420					
156		1100	9C	12	11			8	4	C220					
157		1101	9D	12	11			8	5	C120					
158		1110	9E	12	11			8	6	C0A0					
159		1111	9F	12	11			8	7	C060					
160	1010	0000	A0		11	0		8	1	7020					
161		0001	A1		11	0			1	7000					
162		0010	A2		11	0			2	6800	s				
163		0011	A3		11	0			3	6400	t				
164		0100	A4		11	0			4	6200	u				
165		0101	A5		11	0			5	6100	v				
166		0110	A6		11	0			6	6080	w				
167		0111	A7		11	0			7	6040	x				
168		1000	A8		11	0		8		6020	y				
169		1001	A9		11	0	9			6010	z				
170		1010	AA		11	0		8	2	6820					
171		1011	AB		11	0		8	3	6420					
172		1100	AC		11	0		8	4	6220					
173		1101	AD		11	0		8	5	6120					
174		1110	AE		11	0		8	6	60A0					
175		1111	AF		11	0		8	7	6060					
176	1011	0000	B0	12	11	0		8	1	F020					
177		0001	B1	12	11	0			1	F000					
178		0010	B2	12	11	0			2	E800					
179		0011	B3	12	11	0			3	E400					
180		0100	B4	12	11	0			4	E200					
181		0101	B5	12	11	0			5	E100					
182		0110	B6	12	11	0			6	E080					
183		0111	B7	12	11	0			7	E040					
184		1000	B8	12	11	0		8		E020					
185		1001	B9	12	11	0	9			E010					
186		1010	BA	12	11	0		8	2	E820					
187		1011	BB	12	11	0		8	3	E420					
188		1100	BC	12	11	0		8	4	E220					
189		1101	BD	12	11	0		8	5	E120					
190		1110	BE	12	11	0		8	6	E0A0					
191		1111	BF	12	11	0		8	7	E060					

Ref No.	EBCDIC Binary 0123	EBCDIC Binary 4567	EBCDIC Hex	IBM Card Code Rows 12	11	0	9	8	7-1	IBM Card Code Hex	Graphics and Control Names	1132 Printer EBCDIC Subset Hex	PTTC/8 Hex U-Upper Case L-Lower Case	Console Printer Hex	1403 Printer Hex
192	1100	0000	C0	12		0				A000	(+ zero)				
193*		0001	C1	12					1	9000	A	C1	61 (U)	3C or 3E	64
194*		0010	C2	12					2	8800	B	C2	62 (U)	18 or 1A	25
195*		0011	C3	12					3	8400	C	C3	73 (U)	1C or 1E	26
196*		0100	C4	12					4	8200	D	C4	64 (U)	30 or 32	67
197*		0101	C5	12					5	8100	E	C5	75 (U)	34 or 36	68
198*		0110	C6	12					6	8080	F	C6	76 (U)	10 or 12	29
199*		0111	C7	12					7	8040	G	C7	67 (U)	14 or 16	2A
200*		1000	C8	12				8		8020	H	C8	68 (U)	24 or 26	6B
201*		1001	C9	12			9			8010	I	C9	79 (U)	20 or 22	2C
202		1010	CA	12		0	9	8	2	A830					
203		1011	CB	12		0	9	8	3	A430					
204		1100	CC	12		0	9	8	4	A230					
205		1101	CD	12		0	9	8	5	A130					
206		1110	CE	12		0	9	8	6	A0B0					
207		1111	CF	12		0	9	8	7	A070					
208	1101	0000	D0		11	0				6000	(- zero)				
209*		0001	D1		11				1	5000	J	D1	51 (U)	7C or 7E	58
210*		0010	D2		11				2	4800	K	D2	52 (U)	58 or 5A	19
211*		0011	D3		11				3	4400	L	D3	43 (U)	5C or 5E	1A
212*		0100	D4		11				4	4200	M	D4	54 (U)	70 or 72	5B
213*		0101	D5		11				5	4100	N	D5	45 (U)	74 or 76	1C
214*		0110	D6		11				6	4080	O	D6	46 (U)	50 or 52	5D
215*		0111	D7		11				7	4040	P	D7	57 (U)	54 or 56	5E
216*		1000	D8		11			8		4020	Q	D8	58 (U)	64 or 66	1F
217*		1001	D9		11		9			4010	R	D9	49 (U)	60 or 62	20
218		1010	DA	12	11		9	8	2	C830					
219		1011	DB	12	11		9	8	3	C430					
220		1100	DC	12	11		9	8	4	C230					
221		1101	DD	12	11		9	8	5	C130					
222		1110	DE	12	11		9	8	6	C0B0					
223		1111	DF	12	11		9	8	7	C070					
224	1110	0000	E0			0		8	2	2820					
225		0001	E1		11	0	9		1	7010					
226*		0010	E2			0			2	2800	S	E2	32 (U)	98 or 9A	0D
227*		0011	E3			0			3	2400	T	E3	23 (U)	9C or 9E	0E
228*		0100	E4			0			4	2200	U	E4	34 (U)	B0 or B2	4F
229*		0101	E5			0			5	2100	V	E5	25 (U)	B4 or B6	10
230*		0110	E6			0			6	2080	W	E6	26 (U)	90 or 92	51
231*		0111	E7			0			7	2040	X	E7	37 (U)	94 or 96	52
232*		1000	E8			0		8		2020	Y	E8	38 (U)	A4 or A6	13
233*		1001	E9			0	9			2010	Z	E9	29 (U)	A0 or A2	54
234		1010	EA		11	0	9	8	2	6830					
235		1011	EB		11	0	9	8	3	6430					
236		1100	EC		11	0	9	8	4	6230					
237		1101	ED		11	0	9	8	5	6130					
238		1110	EE		11	0	9	8	6	60B0					
239		1111	EF		11	0	9	8	7	6070					
240*	1111	0000	F0			0				2000	0	F0	1A (L)	C4	49
241*		0001	F1						1	1000	1	F1	01 (L)	FC	40
242*		0010	F2						2	0800	2	F2	02 (L)	D8	01
243*		0011	F3						3	0400	3	F3	13 (L)	DC	02
244*		0100	F4						4	0200	4	F4	04 (L)	F0	43
245*		0101	F5						5	0100	5	F5	15 (L)	F4	04
246*		0110	F6						6	0080	6	F6	16 (L)	D0	45
247*		0111	F7						7	0040	7	F7	07 (L)	D4	46
248*		1000	F8					8		0020	8	F8	08 (L)	E4	07
249*		1001	F9				9			0010	9	F9	19 (L)	E0	08
250		1010	FA	12	11	0	9	8	2	E830					
251		1011	FB	12	11	0	9	8	3	E430					
252		1100	FC	12	11	0	9	8	4	E230					
253		1101	FD	12	11	0	9	8	5	E130					
254		1110	FE	12	11	0	9	8	6	E0B0					
255		1111	FF	12	11	0	9	8	7	E070					

DISK FORMATS

DISK SYSTEM FORMAT (DSF)

Disk system format is the format in which absolute and relocatable programs (mainlines and subprograms) are stored on disk. Disk system format is shown in Figure 16.

Program Header

The format of words 1-12 of the program header is the same for all program types (see Program Types below). These words contain the following information:

Word	Contents
1	Zero
2	Checksum, if the source was cards; otherwise, zero.
3	Program type (bits 4-7), subtype (bits 0-3), and precision (bits 8-15)
4	Effective program length, i.e., the terminal address in the program
5	Length of COMMON (in words)
6	Length of the program header (in words) minus 9
7	Zero
8	Length of the program, including the program header (in disk blocks)
9	FORTRAN indicator (bits 0-7), number of files defined (bits 8-15).
10-11	Name of entry point 1 (in name code)
12	Address of entry point 1 (absolute for type 1 programs, relative for all others)

The format of words 13-54 of the program header varies according to the program type. For program types 1 and 2, the program header consists of words 1-12 only.

For program types 3 and 4, the program header, in addition to words 1-12, contains the following information:

Word	Contents
13-14	Name of entry point 2 (in name code)
15	Relative address of entry point 2
16-17	Name of entry point 3 (in name code)
18	Relative address of entry point 3
19-51	Names and relative addresses of entry points 4 through 14, as required, in the format shown above. The program header ends following the relative address of the last entry point defined; hence, it is of variable length.

For program types 5 and 6, the program header, in addition to words 1-12, contains the following information:

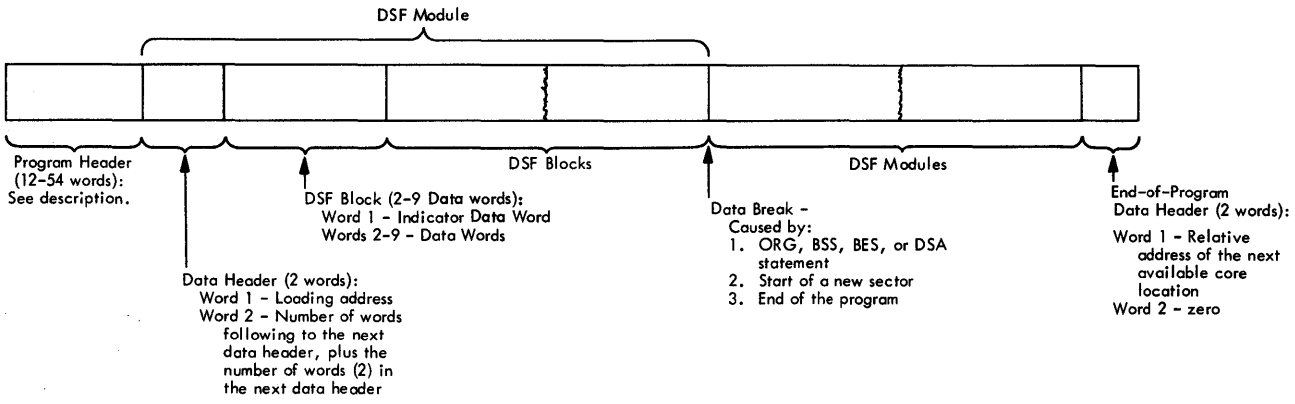


Figure 16. Disk System Format

Word	Contents
13	ISS number plus 50
14	ISS number
15	Number of interrupt levels required*
16	Interrupt level number associated with the primary interrupt*
17	Interrupt level number associated with the secondary interrupt*

*The 1442 Card Read Punch is the only device requiring more than one interrupt level.

For type 7 programs, the program header, in addition to words 1-12, contains the associated interrupt level number in word 13.

Program Types

The program types are defined as follows:

Type	Type of Program
1	Mainline (absolute)
2	Mainline (relocatable)
3	Subprogram, not an ISS, referenced by a LIBF statement
4	Subprogram, not an ISS, referenced by a CALL statement
5	Interrupt service subroutine (ISS) referenced by a LIBF statement
6	Interrupt service subroutine (ISS) referenced by a CALL statement
7	Interrupt level subroutine (ILS)

Program Subtypes

Subtypes are defined for program types 3, 4, 5, and 7 only. When not used, the subtype indicator in the program header contains a zero.

The program subtypes are defined as follows:

Subtype	Type	Description
0	3,4	In-core subprograms
1	3	Disk FORTRAN I/O subroutines
2	3	Arithmetic subroutines
3	3	Non-disk FORTRAN I/O and "Z" conversion subroutines
3	5	"Z" device subroutines
8	4	Function subprogram
1	7	Dummy ILS02, ILS04

DISK DATA FORMAT (DDF)

Disk data format is the format in which data files are stored on the disk. Disk data format consists of 320 binary words per sector. There are no headers, trailers, indicator words, etc.

DISK CORE IMAGE FORMAT (DCI)

Disk core image format is the format in which a core image program is stored on disk. A core image program consists of the Core Image Header, the mainline program, all subprograms referenced in the mainline program or other subprograms (except the disk I/O subroutine), the Transfer Vector, and any LOCALs and SOCALs required. Figure 5 (see STORECI under Disk Utility Programs) shows the layout of a core image program stored on disk.

Core Image Header

The Core Image Header contains the following information:

Word Symbol	Word Relative Address	Contents	
@XEQ1	1	Execution address of the core load	
@CMON	2	Length of COMMON (in words)	
@DREQ	3	Disk I/O subroutine indicator -- /FFFF for DISKZ, /0000 for DISK1, /0001 for DISKN	
@FILE	4	Number of files defined	
@HWET	5	Length of the Core Image Header (in words)	
@LSCT	6	Sector count of files in System WS	
@LDAD	7	Loading address of the core load	
@XCTL	8	Exit control address for DISK1/N	
@TVWC	9	Length of the transfer vector (in words)	
@WCNT	10	Length of the core load (in words)	
@XR3X	11	Setting for index register 3 during execution of the core load	
@ITVX	12	Contents of word 8 during execution	ITV
	13	Contents of word 9 during execution	
	14	Contents of word 10 during execution	
	15	Contents of word 11 during execution	
	16	Contents of word 12 during execution	
	17	Contents of word 13 during execution	
	18-20	Reserved	IBT for ILS04
	21	Interrupt entry to 1231 ISS	
	22	Interrupt entry to 1403 ISS	
	23	Interrupt entry to 2501 ISS	
	24	Interrupt entry to 1442 ISS	
	25	Interrupt entry to Keyboard/ Console Printer ISS	
	26	Interrupt entry to 1134/1055 ISS	
@OVSW	27	Sector count of LOCALs/SOCALs	
@CORE	28	Core size of system on which core load built	
	29-30	Define File Table checksum work area.	

CARD FORMATS

CARD SYSTEM FORMAT (CDS)

Card system format is the format in which absolute and relocatable programs (mainlines and subprograms) are punched into cards. Each deck in card system format consists of (1) a header card, (2) data cards, and (3) an end-of-program card.

Mainline Header Card

The mainline header card is the first card of every type 1 or 2 program in card system format. It contains the following information:

Word	Contents
1	Reserved
2	Checksum
3	Type code (first 8 bits): 0000 0001 -- absolute 0000 0010 -- relocatable Precision code (last 8 bits): 0000 0001 -- standard 0000 0010 -- extended 0000 0000 -- undefined
4	Reserved
5	Length of COMMON, in words (FORTRAN mainline program only)
6	0000 0000 0000 0011
7	Length of the work area required, in words (FORTRAN only)
8	Reserved
9	Define File Count
10-11	Name
12	Relative Entry Point
13-54	Reserved

Subprogram Header Card

The subprogram header card is the first card of every type 3 or 4 program in card system format. It contains the following information:

Word	Contents
1	Reserved
2	Checksum
3	Type code (first 8 bits): 0000 0011 -- to be called by a LIBF statement only 0000 0100 -- to be called by a CALL statement only Precision code (last 8 bits): 0000 0001 -- standard 0000 0010 -- extended 0000 0000 -- undefined
4-5	Reserved
6	Number of entry points times three
7-9	Reserved
10-11	Name of entry point 1 (in name code)
12	Relative address of entry point 1
13-51	Names and relative addresses of entry points 2 through 14, as required
52-54	Reserved

ISS Header Card

The ISS header card is the first card of every type 5 or 6 program in card system format. It contains the following information:

Word	Contents
1	Reserved
2	Checksum
3	Type code (first 8 bits): 0000 0101 -- to be called by a LIBF statement only 0000 0110 -- to be called by a CALL statement only Precision code (last 8 bits): 0000 0001 -- standard 0000 0010 -- extended 0000 0000 -- undefined
4-5	Reserved
6	Number of interrupt levels required plus 6
7-9	Reserved
10-11	Subroutine name (in name code)
12	Relative entry point address
13-14	Reserved for parameters used by the 1130 Card/Paper Tape System
15	Number of interrupt levels required*
16	Interrupt level number associated with the primary interrupt*
17	Interrupt level associated with the secondary interrupt level*
18-29	Reserved
30	One
31-54	Reserved

*The 1442 Card Read Punch is the only device requiring more than one interrupt level.

ILS Header Card

The ILS header card is the first card of every type 7 program in card system format. It contains the following information:

Word	Contents
1	Reserved
2	Checksum
3	Type code (first 8 bits): 0000 0111 Reserved (last 8 bits)
4-5	Reserved
6	0000 0000 0000 0100
7-9	Reserved
10-12	Reserved
13	Interrupt level number
14-54	Reserved

Data Cards

In all types of programs, data cards contain the instructions and data that constitute the machine language program. The format of each data card is as follows:

Word	Contents
1	The loading address of the first data word in the card. Succeeding words go into higher-numbered core locations. The relocation factor must be added to this address to obtain the actual load address. For an absolute program the relocation factor is zero.
2	Checksum
3	Type code (first 8 bits); 0000 1010 Count of data words, excluding indicator data words, in this card (last 8 bits)
4-9	Relocation indicator data words (2 bits for each following data word): 00 -- absolute 01 -- relocatable 10 -- LIBF 11 -- CALL
10	Data word 7
11-54	Data words 8 through 51

End-of-Program (EOP) Card

The end-of-program card is the last card of all programs in card system format. It contains the following information:

Word	Contents
1	Effective length of the program. This number is always even and is assigned by the Assembler, FORTRAN Compiler, or RPG Compiler.
2	Checksum
3	Type code (first 8 bits): 0000 1111 Last 8 bits: 0000 0000
4	Execution address (mainline program only)
5-54	Reserved

Sector Break Cards

Sector break cards are binary cards used by the System Loader to cause programs or phases of programs to start loading at the beginning of a sector. The Monitor system uses Type 1 header cards as sector break cards. The sector break cards are not checksummed. Columns 5-72 of the sector break cards may contain information identifying the program phase being loaded. The card sequence number appears in columns 73-80. Columns 5-80 are punched in IBM Card Code.

Type 1 cards are identified by a 1 punch in column 4 (binary word 3). A Type 1 card indicates to the System Loader that it should check word 11 of the first data card that follows. For the Resident Image, Cold Start Program, and Phase 1 or the System Loader, word 11 contains an absolute starting sector address. For all other Monitor programs or phases word 11 contains the phase ID. Recognition of a phase ID during initial load causes the System Loader to load the program or phase starting at the next sequential sector. During a reload, the phase ID is matched with the ID in SLET and the phase is loaded to the sector address indicated in SLET.

On an initial load, phase 1 of DUP starts loading at sector 8.

A type 2 (relocatable starting sector address sector break card is processed by the Monitor system as a Type 1 sector break card.

CARD DATA FORMAT (CDD)

Card data format is the format in which data files are punched into cards. Card Data format consists of 54 binary words per card. Each binary word occupies 1-1/3 columns. There are no headers, trailers, indicator words, etc.

Card Data format is illustrated in Figure 17.

CARD CORE IMAGE FORMAT (CDC)

Card core image format is the format in which core image programs are punched into cards. Card core image format is identical to card data format; that is, one binary word occupies 1-1/3 columns and 54 binary words can be punched per card.

PAPER TAPE FORMATS

The paper tape formats -- paper tape system format (PTS), paper tape data format (PTD), and paper tape core image format (PTC) -- are analogous to the corresponding card formats (see above).

Two frames in paper tape (data or core image) format contain one binary word and are equivalent to 1-1/3 columns in card (data or core image) format. A data record in paper tape (data or core image) format differs from a data record in card (data or core image) format in that 2 special frames precede the data record; the first contains $7F_{16}$, and the second contains the word count, one-half the number of frames in this data record. A data record in paper tape (data or core image) format contains a maximum of 108 frames (54 binary words) plus the 2 special frames.

Information that would appear in columns 73-80 in card format must not appear in paper tape format.

PRINT FORMAT

PRINT DATA FORMAT (PRD)

Print data format is the format in which DUP prints a DSF program, core image program, or data file on a print device (1403, 1132, or Console Printer).

The Address which precedes each printed line is the core address of Word 1 on that line if a core image program is being printed. If a DSF program or data file is being printed, the Address is the address of Word 1 on that line relative to the start of the DSF program or data file. Each Word printed consists of four hexadecimal characters and represents one binary word. Figure 18 illustrates the DSF and core image print format.

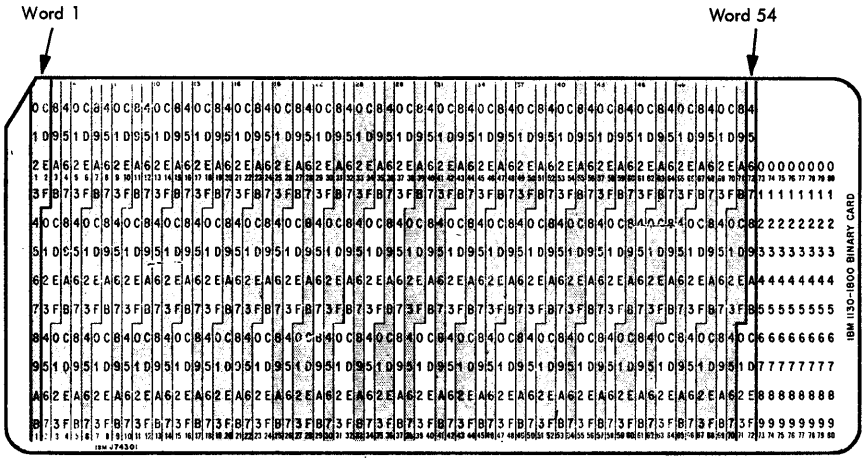


Figure 17. Card Data Format

DSF

```
*DUMP        UA  PR  SYSUP
ADDR    ***0  ***1  ***2  ***3    ***4  ***5  ***6  ***7    ***8  ***9  ***A  ***B    ***C  ***D  ***E  ***F

0000    CCC0  B043  0400  C000    0000  0003  0000  0000    0000  22A2  2917  0000    0000  0035  0444  0000
0010    DC00  0242  2C00  01BC    6D00  01BE  6E00  4700    01C0  6F00  01C2  0689    3155  0099  4408  0028
0020    C101  068A  88A4  D400    010E  6500  00A4  6D00    02E1  04C0  1810  D400    00DB  C400  009F  1890
0030    C400  002C  1101  DC00    023E  C400  023D  D400    002C  C480  C000  1100    4C28  0035  84C0  00BE
0040    DC02  002D  0035  0040    61F8  C500  C000  D500    00C5  7101  70FA  700C    0110  1001  1801  D400
0050    00BF  C400  00D8  1890    44C0  0004  C0F2  7400    00EE  70FD  C07E  4C20    0051  C400  4000  010D
0060    1890  4400  C0F2  7400    00EE  70FD  C400  4101    03D1  D071  D400  00DB    10A0  6200  C600  00C0
0070    11C0  4C18  005C  4400    0244  7201  005A  0035    0001  1000  70F7  61FB    6200  C400  000A  D400
0080    00DF  1010  C400  C0DE    D400  000A  C500  00E5    D400  009F  0000  E871    D067  E870  D067  0863
0090    0864  1003  4C28  4100    007B  1810  D500  00CB    701E  0000  085B  4CC0    4000  0077  3000  C062
00A0    D400  C00A  C054  D400    009A  0000  C056  1890    4400  00F2  7400  C087    0035  0041  00EE  70FD
00B0    C047  D5C0  00CB  C041    D600  C0E5  0410  C03F    D600  00E6  C03D  D600    00E7  7203  1000  0000
00C0    7101  70C9  C045  D400    000A  C480  0006  4420    4044  036E  C053  1890    C400  033C  4480  010E
00D0    7400  0044  0036  70FD    6100  6600  C0E5  6E00    02F0  6200  1100  C600    00C6  4C28  01D5  4C08
00E0    00B4  0010  4440  00B9    9500  00C0  4C18  010F    7201  1000  7403  4000    02F0  70F1  0005  0000
00F0    00C5  C004  0000  C000    00CB  0006  CC00  FFFF    0004  0000  00D2  0018    4005  00D6  CCC0  0292
0100    0CC0  0001  0000  C0BE    00CC  4040  03FE  0000    0600  0701  0077  0000    2000  8800  0000  9000
0110    9800  A000  00F4  0035    4000  00F5  0017  D3D6    C740  C4D9  C9E5  C540    4040  0000  C3C1  D9E3
0120    40E2  D7C5  C340  4040    C3C1  D9E3  C000  40C1    E5C1  C9D3  4040  D7C8    E840  C4D9  C9E5  1040
0130    C540  0398  0000  4400    02D1  C0E2  1890  C600    4411  00C6  4400  0244    C0C3  4C20  012F  C400

0140    010D  0000  1890  4400    00F2  7400  C0EE  0121    0035  0401  70FD  C400    03A2  D400  00E0  629C
0150    C600  03FE  1010  D600    0464  7201  70FA  4400    0285  62FB  C600  4404    00CB  D600  03D1  7201
≈                                                                                                           ≈

0410    0001  03FE  0005  C0C0    0064  CC00  0478  C000    0000  0000  0000  0000    0000  0000  C000  0000
0420    0000  0000  0000  0000
CART ID 3333  DB ADDR 19DB    DB CNT 0035
```

Disk block on sector. For Data Files, this position will always be 0 (Data Files must start on a sector boundary).

Sector

Core Image (note that the actual starting address is 2409)

```
*DUMP        FX  PR  CIMGE
ADDR    ***0  ***1  ***2  ***3    ***4  ***5  ***6  ***7    ***8  ***9  ***A  ***B    ***C  ***D  ***E  ***F

2400                                                        4222  8024  0408    4018  1082  09C0  2409
2410    0CC0  C000  8408  2000    0000  2209  0000  0000    8209  0024  0900  08C0    0000  0000  1002  0000
2420    0000  C000  0000  0000    C000  0000  CC00  0000    0000  0C00  0000  0000    0000  0000  0000  0000
2430    00C0  CC00  C000  0000    C000  0000  C000  0000    0000  0000  C000  0000    0000  0000  0000  C000
2440    FDFF  02C0  C000  C000    0000  0000  0C00  0000    0000  0C00  C000  0000    0000  0000  00C0  0000
2450    CCC0  0000  C000  0000    0000  0000  0000  0000    0000  0C00  0000  00C0    0000  0000  0000  0000
2460    0C00  0000  0000  0000    0000  0000  0C00  0000    0000  0000  0000  0000    0000  0000  0000  0000
2470    0000  CC00  00C0  C000    0000  0000  DB21  0A03    0000  0C00  C000  0000    0000  0000  7DD4  D6C4
2480    C640  CCC0  00C0  00C0    0000  CC00  0C00  0000    0000  0C00  0000  0000    0000  0000  0000  0000
2490    CCC0  C000  C000  0000    0000  C000  0000  0000    0000  0C00  0000  0000    0000  0000  0000  0000
24A0    0000  0000  0000  C000    C000  0000  CC00  0000    0000  0000  0000  0004    F6A4  0A2D  1110  030C
24B0    10C0  C001  C441  0440    7000  01F4  7000  0110    7000  0111  7001  0000    C8F7  4400  00F2  0689
24C0    3155  0099  068A  88A4    D400  00AF  C400  00EE    4820  70FC  C8EB  4400    00F2  C400  00EE  4820
24D0    70FC  C076  D4C0  0111    C8E3  4480  00AF  4480    00AF  C06D  4480  00AF    C06A  4480  00AF  6500
24E0    0138  0031  8BE6  0A2D    1044  0440  4111  0010    0110  0400  6D00  000B    C8D6  4480  00AF  6500
24F0    0179  69D2  C8D0  4480    00AF  6500  01B7  69CC    C8CA  4480  00AF  C054    4480  00AF  66C0  01F6

2500    6700  013C  6B24  6500    FE54  C200  4C18  008D    9049  4818  685E  C200    9500  0622  4C18  0059
2510    7104  70F9  63FA  C500    0623  18D0  1010  005E    EF85  0A2D  0410  1004    0000  0000  0001  0000
2520    1088  D7C0  C110  F04B    4C18  0095  1010  1088    D700  0111  7101  7302    70EF  4400  00E2  6700
2530    CCC0  C838  4056  C82C    4054  C82C  4052  C828    405C  C82A  404E  C824    404C  C828  404A  C820
2540    4C48  C826  4046  6BEC    C0EB  9026  4C18  00B4    C817  403F  C815  403D    6BE3  008B  B2EF  0A12
2550    0010  0000  C000  C000    0000  0000  7201  708D    61EA  C01C  D500  0110    7101  70FB  7203  70D8
2560    C018  70C8  7002  C028    00A1  0009  0039  00C0    0000  C000  0000  0000    0C00  0000  0000  C000
2570    CCC0  C0C0  C0C0  C0C0    0000  CC00  0000  0000    0000  0CC0  0000  0000    0000  0000  0000  0000
2580    0C00  C000  0000  009E    8429  0A2D  1111  1140    0001  1110  0100  0000    0002  00AB  0002  00FC
2590    0004  00FE  0004  0102    0004  0106  C005  010A    0178  0040  0040  007C    007D  0000  0000  0000
25A0    0C00  CCC0  6700  013B    6F00  CC0B  CCC0  000A    4480  00AF  C400  0036    4820  70FC  74C0  00B0
25B0    7002  7201  7082  4CC0    0038  0000  D0E9  18D0    D0E8  00CB  970F  0A2D    1040  0100  1104  0000
25C0    0C40  0400  7400  00B1    7006  C480  00B3  1008    D300  68DE  7007  C300    EC80  00B3  D300  1010
25D0    D0D7  7301  7401  00B3    74FF  00B2  70EB  4C80    00C7  0C00  690F  61F0    63FC  C200  18D0  1010
25E0    1084  90AF  4808  80AE    80AE  D500  010A  7101    7004  6500  0000  4C80    00E2  7301  70F0  00F8
25F0    1211  CA02  CCC0  C000    0000  0000  0000  0000    7201  70EB  0000  0000    0000  0000  0000  0000

2600    0000  0000  0000  0000    0000  C000  CC00  0000    0000  0000  0C00  0000    0000  0000  0000  0000
2610    0000  C000  0000  00C0    0000  0000  0000  0000    0000  0C00  0000  0000    0000  0000  0000  0000
2620    0CC0  CC00  0C00  C000    0000  9500  0F63  4C18    09B6  70CA  7112  4528    F000  0000  0000  0000
2630    0FB1  D851  9832  4C20    05F3  1090  4C18  06BD    0000  CC00  G000  6AFD    C0DF  90FB  D0FA  4C10
2640    09CB  C400  0A48  D400    CA43  C07F  1890  4400    00F2  7400  00EE  70FD    6600  0139  6AEA  6203
2650    C0DD  D6C0  CA48  C0D6    D6C0  0A46  C0D4  D6C0    0A47  C0DE  D400  0A48    C0DC  D400  0A47  C500
2660    0F63  D5C0  0F5E  C061    1890  C400  0864  4400    00F2  74C0  00EE  70FD    C500  0032  E0BD  4C08
2670    0A13  EC00  0F35  D400    0F98  C400  0C56  1890    4400  00F2  7400  00EE    70FD  62FB  C500  0F72
2680    9600  CFD8  4C18  0A05    7201  70F8  C400  0698    4C00
CART ID 3333  DB ADDR 1220    DB CNT 0020
```

Figure 18. Dump of DSF and Core Image Program

APPENDIX D. DISK STORAGE UNIT CONVERSION FACTORS

No. Of / Per:	Word	Disk Block	Sector	Track	Cylinder	Disk
Bits	16	320	5,112	20,480	40,960	8,192,000
Data Words		20	320*	1,280	2,560	512,000
Disk Block			16	64	128	25,600
Sectors				4	8	1,600
Tracks					2	400
Cylinders						200

*These follow the first actual word of each sector, which is used for the address.

APPENDIX E. DECIMAL AND HEXADECIMAL DISK ADDRESSES

SECTOR ADDRESS BASE 10	SECTOR ADDRESS BASE 16	CYLINDER ADDRESS BASE 10	CYLINDER ADDRESS BASE 16	SECTOR ADDRESS BASE 10	SECTOR ADDRESS BASE 16	CYLINDER ADDRESS BASE 10	CYLINDER ADDRESS BASE 16
+00000	0000	+00000	0000	+00800	0320	+00100	0064
+00008	0008	+00001	0001	+00808	0328	+00101	0065
+00016	0010	+00002	0002	+00816	0330	+00102	0066
+00024	0018	+00003	0003	+00824	0338	+00103	0067
+00032	0020	+00004	0004	+00832	0340	+00104	0068
+00040	0028	+00005	0005	+00840	0348	+00105	0069
+00048	0030	+00006	0006	+00848	0350	+00106	006A
+00056	0038	+00007	0007	+00856	0358	+00107	006B
+00064	0040	+00008	0008	+00864	0360	+00108	006C
+00072	0048	+00009	0009	+00872	0368	+00109	006D
+00080	0050	+00010	000A	+00880	0370	+00110	006E
+00088	0058	+00011	000B	+00888	0378	+00111	006F
+00096	0060	+00012	000C	+00896	0380	+00112	0070
+00104	0068	+00013	000D	+00904	0388	+00113	0071
+00112	0070	+00014	000E	+00912	0390	+00114	0072
+00120	0078	+00015	000F	+00920	0398	+00115	0073
+00128	0080	+00016	0010	+00928	03A0	+00116	0074
+00136	0088	+00017	0011	+00936	03A8	+00117	0075
+00144	0090	+00018	0012	+00944	03B0	+00118	0076
+00152	0098	+00019	0013	+00952	03B8	+00119	0077
+00160	00A0	+00020	0014	+00960	03C0	+00120	0078
+00168	00A8	+00021	0015	+00968	03C8	+00121	0079
+00176	00B0	+00022	0016	+00976	03D0	+00122	007A
+00184	00B8	+00023	0017	+00984	03D8	+00123	007B
+00192	00C0	+00024	0018	+00992	03E0	+00124	007C
+00200	00C8	+00025	0019	+01000	03E8	+00125	007D
+00208	00D0	+00026	001A	+01008	03F0	+00126	007E
+00216	00D8	+00027	001B	+01016	03F8	+00127	007F
+00224	00E0	+00028	001C	+01024	0400	+00128	0080
+00232	00E8	+00029	001D	+01032	0408	+00129	0081
+00240	00F0	+00030	001E	+01040	0410	+00130	0082
+00248	00F8	+00031	001F	+01048	0418	+00131	0083
+00256	0100	+00032	0020	+01056	0420	+00132	0084
+00264	0108	+00033	0021	+01064	0428	+00133	0085
+00272	0110	+00034	0022	+01072	0430	+00134	0086
+00280	0118	+00035	0023	+01080	0438	+00135	0087
+00288	0120	+00036	0024	+01088	0440	+00136	0088
+00296	0128	+00037	0025	+01096	0448	+00137	0089
+00304	0130	+00038	0026	+01104	0450	+00138	008A
+00312	0138	+00039	0027	+01112	0458	+00139	008B
+00320	0140	+00040	0028	+01120	0460	+00140	008C
+00328	0148	+00041	0029	+01128	0468	+00141	008D
+00336	0150	+00042	002A	+01136	0470	+00142	008E
+00344	0158	+00043	002B	+01144	0478	+00143	008F
+00352	0160	+00044	002C	+01152	0480	+00144	0090
+00360	0168	+00045	002D	+01160	0488	+00145	0091
+00368	0170	+00046	002E	+01168	0490	+00146	0092
+00376	0178	+00047	002F	+01176	0498	+00147	0093
+00384	0180	+00048	0030	+01184	04A0	+00148	0094
+00392	0188	+00049	0031	+01192	04A8	+00149	0095
+00400	0190	+00050	0032	+01200	04B0	+00150	0096
+00408	0198	+00051	0033	+01208	04B8	+00151	0097
+00416	01A0	+00052	0034	+01216	04C0	+00152	0098
+00424	01A8	+00053	0035	+01224	04C8	+00153	0099
+00432	01B0	+00054	0036	+01232	04D0	+00154	009A
+00440	01B8	+00055	0037	+01240	04D8	+00155	009B
+00448	01C0	+00056	0038	+01248	04E0	+00156	009C
+00456	01C8	+00057	0039	+01256	04E8	+00157	009D
+00464	01D0	+00058	003A	+01264	04F0	+00158	009E
+00472	01D8	+00059	003B	+01272	04F8	+00159	009F
+00480	01E0	+00060	003C	+01280	0500	+00160	00A0
+00488	01E8	+00061	003D	+01288	0508	+00161	00A1
+00496	01F0	+00062	003E	+01296	0510	+00162	00A2
+00504	01F8	+00063	003F	+01304	0518	+00163	00A3
+00512	0200	+00064	0040	+01312	0520	+00164	00A4
+00520	0208	+00065	0041	+01320	0528	+00165	00A5
+00528	0210	+00066	0042	+01328	0530	+00166	00A6
+00536	0218	+00067	0043	+01336	0538	+00167	00A7
+00544	0220	+00068	0044	+01344	0540	+00168	00A8
+00552	0228	+00069	0045	+01352	0548	+00169	00A9
+00560	0230	+00070	0046	+01360	0550	+00170	00AA
+00568	0238	+00071	0047	+01368	0558	+00171	00AB
+00576	0240	+00072	0048	+01376	0560	+00172	00AC
+00584	0248	+00073	0049	+01384	0568	+00173	00AD
+00592	0250	+00074	004A	+01392	0570	+00174	00AE
+00600	0258	+00075	004B	+01400	0578	+00175	00AF
+00608	0260	+00076	004C	+01408	0580	+00176	00B0
+00616	0268	+00077	004D	+01416	0588	+00177	00B1
+00624	0270	+00078	004E	+01424	0590	+00178	00B2
+00632	0278	+00079	004F	+01432	0598	+00179	00B3
+00640	0280	+00080	0050	+01440	05A0	+00180	00B4
+00648	0288	+00081	0051	+01448	05A8	+00181	00B5
+00656	0290	+00082	0052	+01456	05B0	+00182	00B6
+00664	0298	+00083	0053	+01464	05B8	+00183	00B7
+00672	02A0	+00084	0054	+01472	05C0	+00184	00B8
+00680	02A8	+00085	0055	+01480	05C8	+00185	00B9
+00688	02B0	+00086	0056	+01488	05D0	+00186	00BA
+00696	02B8	+00087	0057	+01496	05D8	+00187	00BB
+00704	02C0	+00088	0058	+01504	05E0	+00188	00BC
+00712	02C8	+00089	0059	+01512	05E8	+00189	00BD
+00720	02D0	+00090	005A	+01520	05F0	+00190	00BE
+00728	02D8	+00091	005B	+01528	05F8	+00191	00BF
+00736	02E0	+00092	005C	+01536	0600	+00192	00C0
+00744	02E8	+00093	005D	+01544	0608	+00193	00C1
+00752	02F0	+00094	005E	+01552	0610	+00194	00C2
+00760	02F8	+00095	005F	+01560	0618	+00195	00C3
+00768	0300	+00096	0060	+01568	0620	+00196	00C4
+00776	0308	+00097	0061	+01576	0628	+00197	00C5
+00784	0310	+00098	0062	+01584	0630	+00198	00C6
+00792	0318	+00099	0063	+01592	0638	+00199	00C7

Type and sub-type are defined in Appendix C.

System Library Programs	Names	Type and Sub-type	Subroutines Required	ID Field (cc 73 - 75)
MAINLINES				
Disk Maintenance Programs				
Disk Initialization	DISC	2,0	SYSUP, RDREC, DISKZ	U6C
Print Cartridge ID	IDENT	2,0	CALPR, DISKZ	U6F
Change Cartridge ID	ID	2,0	RDREC, CALPR, DISKZ	U6G
Disk Copy	COPY	2,0	RDREC, DISKZ	U6B
Writer Sector Addresses in WS	ADRWS (cannot be called)	2	Linked From DUP DWADR	U6A
Delete CIB	DLCIB	2,0	RDREC, DISKZ	U6D
Dump System Location Equivalence Table	DSLET	2,0	FSLEN, DISKZ	U6E
System Maintenance	MODIF	-	DISKZ	U6H
Paper Tape Utility				
Keyboard or 1134 Input/Console Printer or 1055 Output	PTUTL	--		U6I
SUBROUTINES				
Utility Calls				
Selective Dump on Console Printer	DMTD0, DMTX0	4,0	WRTY0	U5B
Selective Dump on 1132 Printer	DMPD1, DMPX1	4,0	PRNT1	U5C
Dump 80 Subroutine	DMP80	4,0	None	U5A
Update DCOM	SYSUP	4,0	FSLEN, FSYSU	U5E
Call System Print	CALPR	4,0	FSLEN	U7A
Read *ID Record	RDREC	4,0	FSLEN	U7C
Fetch Phase IDs or, Fetch System Subroutine	FSLEN, FSYSU	4,0	DISKZ	U7B
Common FORTRAN Calls				
Test Data Entry Switches	DATSW	4,8	None	T3A
Divide Check Test	DVCHK	4,8	None	T3B
Functional Error Test	FCTST	4,8	None	T3C
Overflow Test	OVERF	4,8	None	T3E
Sense Light Control and Test	SLITE, SLITT	4,8	None	T3G
FORTRAN Trace Stop	TSTOP	4,8	TSET	T3H
FORTRAN Trace Start	TSTRT	4,8	TSET	T3I
Integer Transfer of Sign	ISIGN	4,8	None	T3D
Extended Arith/Funct Calls				
Extended Precision Hyperbolic Tangent	ETANH, ETNH	4,8	EEXP, ELD/ESTO, EADD, EDIV, EGETP	S2I
Extended Precision A**B Function	EAXB, EAXBX	4,8	EEXP, ELN, EMPY	S2C
Extended Precision Natural Logarithm	ELN, EALOG	4,8	XMD, EADD, EMPY, EDIV, NORM, EGETP	S2E
Extended Precision Exponential	EEXP, EXPN	4,8	XMD, FARC, EGETP	S2D
Extended Precision Square Root	ESQR, ESQRT	4,8	ELD/ESTO, EADD, EMPY, EDIV, EGETP	S2H
Extended Precision Sine-Cosine	ESIN, ESINE, ECOS, ECOSN	4,8	EADD, EMPY, NORM, XMD, EGETP	S2G
Extended Precision Arctangent	EATN, EATAN	4,8	EADD, EMPY, EDIV, XMD, EGETP, NORM	S2B
Extended Precision Absolute Value Function	EABS, EAVL	4,8	EGETP	S2A
FORTRAN Sign Transfer Calls				
Extended Precision Transfer of Sign	ESIGN	4,8	ESUB, ELD	S2F
Standard Precision Transfer of Sign	FSIGN	4,8	FSUB, FLD	R2F
Standard Arith/Funct Calls				
Standard Precision Hyperbolic Tangent	FTANH, FTNH	4,8	FEXP, FLD/FSTO, FADD, FDIV, FGETP	R2I
Standard Precision A**B Function	FAXB, FAXBX	4,8	FEXP, FLN, FMPY	R2C
Standard Precision Natural Logarithm	FLN, FALOG	4,8	FSTO, XMDS, FADD, FMPY, FDIV, NORM, FGETP	R2E
Standard Precision Exponential	FEXP, FXPN	4,8	XMDS, FARC, FGETP	R2D
Standard Precision Square Root	FSQR, FSQRT	4,8	FLD/FSTO, FADD, FMPY, FDIV, FGETP	R2H
Standard Precision Sine-Cosine	FSIN, FSINE, FCOS, FCOSN	4,8	FADD, FMPY, NORM, XMDS, FSTO, FGETP	R2G
Standard Precision Arctangent	FATN, FATAN	4,8	FADD, FMPY, FDIV, XMDS, FSTO, FGETP	R2B
Standard Precision Absolute Value Function	FABS, FAVL	4,8	FGETP	R2A
Common Arith/Funct Calls				
Fixed Point (Fractional) Square Root	XSQR	4,8	None	T1C
Integer Absolute Function	IABS	4,8	None	T1B
Floating Binary/EBC Decimal Conversions	FBTD (BIN. TO DEC.) FDTB (DEC. TO BIN.)	4,0	None	T1A T1A
Flipper for LOCAL SOCAL Subprograms				
FORTRAN Trace Subroutines	FLIPP	4,0	DISKZ, DISK1, or DISKN	U5D
Extended Floating Variable Trace	SEAR, SEARX	3,0	ESTO, TTEST, SWRT, SIOF, SCOMP	S2J
Fixed Variable Trace	SIAR, SIARX	3,0	TTEST, SWRT, SIOI, SCOMP	T6B
Standard Floating IF Trace	SFIF	3,0	FSTO, TTEST, SWRT, SIOF, SCOMP	R2K
Extended Floating IF Trace	SEIF	3,0	FSTO, TTEST, SWRT, SIOF, SCOMP	S2K
Fixed IF Trace	SIIF	3,0	TTEST, SWRT, SIOI, SCOMP	T6C
Standard Floating Variable Trace	SFAR, SFARX	3,0	FSTO, TTEST, SWRT, SIOF, SCOMP	R2J
GO TO Trace	SGOTO	3,0	TTEST, SWRT, SIOI, SCOMP	T6A

System Library Programs	Names	Type and Sub-type	Subroutines Required	ID Field (cc 73 - 75)
Non-Disk FORTRAN Format I/O				
FORTRAN Format Subroutine	SFIO, SIOI, SIOAI, SIOF, SIOAF, SIOFX, SCOMP, SWRT, SRED, SIOIX	3,3	FLOAT, ELD/ESTO or FLD/FSTO, IFIX	T4C
FORTRAN Find Subroutine	SDFND	3,1	DISKZ, DISK1 or DISKN	T4B
Disk FORTRAN I/O	SDFIO, SDRED, SDWRT, SDCOM, SDAF, SDF, SDI, SDIX, SDFX, SDAI	3,1	DISKZ, DISK1 or DISKN	T4A
FORTRAN Unformatted Disk I/O	SUFIO	3,1	DISKZ, DISK1 or DISKN	T4D
FORTRAN Common LIBFs				
FORTRAN Pause	PAUSE	3,2	None	T2A
FORTRAN Stop	STOP	3,2	None	T2B
FORTRAN Subscript Displacement Calculation	SUBSC	3,0	None	T2D
FORTRAN Subroutine Initialization	SUBIN	3,0	None	T2C
FORTRAN Trace Test and Set	TTEST, TSET	3,0	None	T2E
FORTRAN I/O and Conversion Subroutines				
FORTRAN 1442 Input/Output Subroutine	CARDZ	5,3	HOLEZ, GETAD, EBCTB, HOLTB, ILS00, ILS04	T5A
FORTRAN 1442 Output Subroutine	PNCHZ	5,3	HOLEZ, GETAD, EBCTB, HOLTB, ILS00, ILS04	T5G
FORTRAN 2501 Input Subroutine	READZ	5,3	HOLEZ, GETAD, EBCTB, HOLTB, ILS04	T5J
Disk I/O Routine (Part of Supervisor)	DISKZ	-	ILS02	--
FORTRAN Paper Tape Subroutine	PAPTZ	5,3	ILS04	T5F
FORTRAN 1132 Printer Subroutine	PRNTZ	5,3	ILS01	T5H
FORTRAN 1403 Printer Subroutine	PRNZ	5,3	ILS04	T5I
FORTRAN Keyboard-Typewriter Subroutine	TYPEZ	5,3	GETAD, EBCTB, HOLEZ, ILS04	T5K
FORTRAN Typewriter Subroutine	WRTYZ	5,3	GETAD, EBCTB, ILS04	T5L
FORTRAN 1627 Plotter Subroutine	PLOTX	5,0	ILS03	V1L
FORTRAN Hollerith to EBCDIC Conversion	HOLEZ	3,3	GETAD, EBCTB, HOLTB	T5D
FORTRAN Get Address Routine	GETAD	3,3	None	T5C
FORTRAN EBCDIC Table	EBCTB	3,3	None	T5B
FORTRAN Hollerith Table	HOLTB	3,3	None	T5E
Extended Arith/Funct LIBFs				
Extended Precision Get Parameter Subroutine	EGETP	3,2	ELD	S1I
Extended Precision A**I Function	EAXI, EAXIX	3,2	ELD/ESTO, EMPY, EDVR	S1B
Extended Precision Divide Reverse	EDVR, EDVRX	3,2	ELD/ESTO, EDIV	S1D
Extended Precision Float Divide	EDIV, EDIVX	3,2	XDD, FARC	S1C
Extended Precision Float Multiply	EMPY, EMPYX	3,2	XMD, FARC	S1G
Extended Precision Subtract Reverse	ESBR, EXBRX	3,2	EADD	S1H
Extended Add-Subtract	EADD, ESUB, EADDX, ESUBX	3,2	FARC, NORM	S1A
Extended Load-Store	ELD, ELDX, ESTO, ESTOX	3,0	None	S1F
Standard Arith/Funct LIBFs				
Standard Precision Get Parameter Subroutine	FGETP	3,2	FLD	R1E
Standard Precision A**I Function	FAXI, FAXIX	3,2	FLD/FSTO, FMPY, FDVR	R1B
Standard Precision Divide Reverse	FDVR, FDVRX	3,2	FLD/FSTO, FDIV	R1D
Standard Precision Float Divide	FDIV, FDIVX	3,2	FARC	R1C
Standard Precision Float Multiply	FMPY, FMPYX	3,2	XMDS, FARC	R1G
Standard Precision Subtract Reverse	FSBR, FSBRX	3,2	FADD	R1H
Standard Add-Subtract	FADD, FSUB, FADDX, FSUBX	3,2	NORM, FARC	R1A
Standard Load-Store	FLD, FLDX, FSTO, FSTOX	3,0	None	R1F
Standard Precision Fractional Multiply	XMDS	3,2	None	S3I
Common Arith/Funct LIBFs				
Fixed Point (Fractional) Double Divide	XDD	3,2	XMD	S3G
Fixed Point (Fractional) Double Multiply	XMD	3,2	None	S3H
Sign Reversal Function	SNR	3,2	None	S3F
Integer to Floating Point Function	FLOAT	3,0	NORM	S3C
Floating Point to Integer Function	IFIX	3,0	None	S3D
I**J Integer Function	FIXI, FIXIX	3,2	None	S3B
Normalize Subroutine	NORM	3,0	None	S3E
Floating Accumulator Range Check Subroutine	FARC	3,2	None	S3A
Interrupt Service Subroutines				
1442 Card Read Punch Input/Output (No Error Parameter)	CARD0	5,0	ILS00, ILS04	U2A
1442 Card Read Punch Input/Output (Error Parameter)	CARD1	5,0	ILS00, ILS04	U2B
2501 Card Read Input (No Error Parameter)	READ0	5,0	ILS04	U2L
2501 Card Read Input (Error Parameter)	READ1	5,0	ILS04	U2M
1442 Card Punch Output (No Error Parameter)	PNCH0	5,0	ILS00, ILS04	U2H
1442 Card Punch Output (Error Parameter)	PNCH1	5,0	ILS00, ILS04	U2T
Multiple Sector Disk Input/Output (Part of Supervisor)	DISK1	-	ILS02	--
High Speed Multiple Sector Disk Input/Output (Part of Supervisor)	DISKN	-	ILS02	--
Synchronous Communications Adaptor (SCA) STR Mode	SCAT1	5,0	ILS01	W1F

System Library Programs	Names	Type and Sub-type	Subroutines Required	ID Field (cc 73 - 75)
Interrupt Service Subroutines (Cont'd)				
SCA(BSC, Point-to-Point Mode)	SCAT2	5,0	ILS01	
SCA(BSC, Multi-Point Mode)	SCAT3	5,0	ILS01	
Paper Tape Input/Output	PAPT1	5,0	ILS04	U2D
Simultaneous Paper Tape Input/Output	PAPTN	5,0	ILS04	U2E
Character/Word Count Paper Tape Input/Output	PAPTX	5,0	ILS04	U2F
Plotter Output Subroutine	PLOT1	5,0	ILS03	U2G
1132 Printer Output Subroutine	PRNT1	5,0	ILS01	U2J
1132-SCA Print With Overlap	PRNT2	5,0	ILS01	W1E
1403 Printer Output Subroutine	PRNT3	5,0	ILS04	U2K
Keyboard/Console Printer Input/Output	TYPE0	5,0	HOL, PRTY, ILS04	U2N
Console Printer Output Subroutine	WRTY0	5,0	ILS04	U2O
1231 Optical Mark Page Reader Input Subroutine	OMPR1	5,0	ILS04	U2C
Conversion Subroutines				
Binary Word to 6 Decimal Characters (Card Code)	BINDC	3,0	None	U4B
Binary Word to 4 Hexadecimal Characters (Card Code)	BINHX	3,0	None	U4C
6 Decimal Characters (Card Code) to Binary Word	DCBIN	3,0	None	U4G
EBCDIC to Console Printer Output Code	EBPRT	3,0	EBPA, PRTY	U3A
Card Code to EBCDIC-EBCDIC to Card Code	HOLEB	3,0	EBPA, HOLL	U3B
Card Code to Console Printer Output Code	HOLPR	3,0	HOLL, PRTY	U3C
4 Hexadecimal Characters (Card Code) to Binary Word	HXBIN	3,0	None	U3D
PTTC/8 to EBCDIC-EBCDIC to PTTC/8	PAPEB	3,0	EBPA	U3E
PTTC/8 to Card Code-Card Code to PTTC/8	PAPHL	3,0	EBPA, HOLL	U3F
PTTC/8 to Console Printer Output Code	PAPPR	3,0	EBPA, PRTY	U3G
Card Code to EBCDIC-EBCDIC to Card Code	SPEED	3,0	None	U3H
4 of 8 Code to EBCDIC, EBCDIC to 4 of 8 Code	EBC48	3,0	HXCV, STRTB	W1A
4 of 8 Code to IBM Card Code, IBM Card Code to 4 of 8 Code	HOL48	3,0	HXCV, HOLCA, STRTB	W1B
4 of 8 Code to Table of Displacements	HXCV	3,0	None	W1D
32-Bit Binary Value to IBM Card Code Decimal Value	BIDEC	3,0	None	U4A
IBM Card Code Decimal Value to 32-Bit Binary Value	DECBI	3,0	None	U4H
Supplement to All Standard Conversions Except Those Involving PTTC/8	ZIPCO	3,0	Any ZIPCO Conversion Table	U3I
Conversion Tables				
EBCDIC and PTTC/8	EBPA	3,0	None	U4K
Card Code Table	HOLL	3,0	None	U4P
Console Printer Output Code Table	PRTY	3,0	None	U4Q
Table of IBM Card Codes	HOLCA	3,0	None	W1C
Table of 4 of 8 and EBCDIC Codes	STRTB	3,0	None	W1G
ZIPCO Conversion Tables				
EBCDIC to Console Printer Code	EBCCP	3,0	None	U4I
EBCDIC to IBM Card Code	EBHOL	3,0	None	U4J
EBCDIC to 1403 Printer Code	EBPT3	3,0	None	U4L
Console Printer Code to EBCDIC	CPEBC	3,0	None	U4D
Console Printer Code to IBM Card Code	CPHOL	3,0	None	U4E
Console Printer Code to 1403 Printer Code	CPPT3	3,0	None	U4F
IBM Card Code to EBCDIC	HLEBC	3,0	None	U4M
IBM Card Code to Console Printer Code	HOLCP	3,0	None	U4O
IBM Card Code to 1403 Printer Code	HLPT3	3,0	None	U4N
1403 Printer Code to EBCDIC	PT3EB	3,0	None	U4S
1403 Printer Code to Console Printer Code	PT3CP	3,0	None	U4R
1403 Printer Code to IBM Card Code	PTHOL	3,0	None	U4T
Interrupt Level Subroutines				
Interrupt Level Zero Subroutine	ILS00	7,0	None	U1A
Interrupt Level One Subroutine	ILS01	7,0	None	U1B
Interrupt Level Two Subroutine (Part of Supervisor)	ILS02	7,1	None	U1C
Interrupt Level Three Subroutine	ILS03	7,0	None	U1D
Interrupt Level Four Subroutine (Part of Supervisor)	ILS04	7,1	None	U1E
Standard Plot Calls				
Standard Precision Character	FCHAR	4,0	FSIN, FCOS, FPLOT, FCHRX, FLD, FSTOX, FSTO	V1F
Standard Precision Scale	SCALF	4,0	FRULE	V1O
Standard Precision Grid	FGRID	4,0	FPLOT, POINT, FADD, FLD, FSTO, SNR	V1H
Standard Precision Plot	FPLOT	4,0	FMOVE, XYPLT, PLOTI	V1I

System Library Programs	Names	Type and Sub-type	Subroutines Required	ID Field (cc 73-75)
Extended Plot Calls				
Extended Precision Character	ECHAR	4,0	ESIN, ECOS, EPLOT, ECHRX, ELD, ESTO, ESTOX	V1A
Extended Precision Scale	SCALE	4,0	ERULE	V1N
Extended Precision Grid	EGRID	4,0	EPLOT, POINT, EADD, ELD, ESTO, SNR	V1C
Extended Precision Plot	EPLOT	4,0	EMOVE, XYPLT, PLOTI	V1D
Common Plot Call				
Point Characters	POINT	4,0	PLOTI	V1M
Standard Plot LIBFs				
Standard Precision Annotation	FCHRX, FCHRI, WCHRI	3,0	FLOAT, FMPY, IFIX, FADD, FLDX, FINC, XYPLT, PLOTI, FSTOX, FLD	V1G
Standard Precision Plot Scaler	FRULE, FMOVE, FINC	3,0	FLDX, FSUBX, FMPYX, FLD, FSTOX, FMPY, IFIX, FADD	V1J
Extended Plot LIBFs				
Extended Precision Annotation	ECHRX, ECHRI, YCHRI	3,0	FLOAT, EMPY, IFIX, EADD, ELDX, EINC, XYPLT, PLOTI, ESTOX, ELD	V1B
Extended Precision Plot Scaler	ERULE, EMOVE, EINC	3,0	ELDX, ESUBX, EMPYX, ELD, ESTOX, EMPY, IFIX, EADD, ESTO	V1E
Common Plot LIBFs				
Pen Mover	XYPLT	3,2	PLOTI	V1P
Interface	PLOTI	3,2	PLOTX	V1K
Interrupt Service	PLOTX	5,0	ILS03	V1L
Disk I/O				
Sequential Access	SEQOP, SEQIO, SEQCL	3,0	DISKZ	W3F
Direct Access	DAOPN, DAIO, DACLS	3,0	DISKZ	W3E
ISAM Load	ISLDO, ISLD, ISLDC	3,0	DISKZ	W3D
ISAM Add	ISADO, ISAD, ISADC	3,0	DISKZ	W3C
ISAM Sequential	ISEQO, ISETL, ISEQ, ISEQC	3,0	DISKZ	W3B
ISAM Random	ISRDO, ISRD, ISRDC	3,0	DISKZ	W3A
RPG Decimal Arithmetic				
Add, Subtract and Numeric Compare	RGADD, RGSUB, RGNCP	3,0	None	W2T
Multiply	RGMLT	3,0	RGBTD, RGDTB	W2S
Divide	RGDIV	3,0	None	W2R
Move Remainder	RGMVR	3,0	RGBTD	W2Q
Binary Conversion	RGBTD, RGDTB	3,0	None	W2P
RPG Sterling and Edit				
Sterling Input Conversion	RGSTI	3,0	RGBTD, RGDTB	W4B
Sterling Output Conversion	RGSTO	3,0	RGBTD, RGDTB	W4A
Edit	RGEDT	3,0	None	W2O
RPG Move				
From I/O Buffer to Core	RGMV1, RGMV5	3,0	None	W2N
From Core to I/O Buffer	RGMV2	3,0	None	W2M
MOVE Operation	RGMV3	3,0	None	W2L
MOVEL Operation	RGMV4	3,0	None	W2K
RPG Compare				
Alphameric	RGCMP	3,0	None	W2J
RPG Indicators				
Test	RGSI1	3,0	None	W2I
Set Resulting On	RGSI2	3,0	None	W2H
Set On	RGSI3	3,0	None	W2G
Set Off	RGSI4	3,0	None	W2F
Test for 0 or Blank	RGSI5	3,0	None	W2E
RPG Miscellaneous				
Test Zone	RGTSZ	3,0	None	W2D
Convert to Binary	RGCVB	3,0	None	W2C
Object Time Error	RGERR	3,0	None	W2B
Blank After	RGBLK	3,0	None	W2A
Alternating Sequence	ALTSE	- -		

The Location Equivalence Table (LET) contains the name and disk block count of all programs and data files stored in the User Area, including the System Library. The Fixed Location Equivalence Table (FLET) contains the names of all programs and data files stored in the Fixed Area.

Each cartridge must have a LET. FLET is optionally defined on each cartridge by the DUP control record DEFINE FIXED AREA.

LET/FLET DISK FORMAT

All entries are three words long and consist of a name and disk block count. In addition, each sector of LET/FLET contains a five word sector header.

LET/FLET Entries

Entries	DSF (LET only)	CI	Data
Word 1, bits 0-1	00	10	11
Word 1, bits 2-15 plus Word 2	Program or data file name in name code		
Word 3	DB count of program or data file		

Sometimes unused disk space occurs because data files and programs in core image format are stored on sector boundaries. Such spaces are represented by a 1DUMY entry.

A 1DUMY entry is always inserted to precede a DDF or DCI entry when the last entry is in DSF format, even when the preceding program ends on a sector boundary. This case will generate a 1DUMY with a DB count of zero (blank). This is done because a DELETE operation in the future might call for a 1DUMY "padding", and under certain circumstances there may be no room to insert a 1DUMY entry.

The format is the following:

Word 1, bits 0-1	Reserved
Word 1, bits 2-15 plus Word 2	Name code for 1DUMY
Word 3	DB count of entry

The last entry of LET is a 1DUMY entry reflecting the current size of WS available.

LET/FLET Sector Header

Word	Entry
1	Relative sector number for this cartridge only
2	Sector address of the UA (sector address of FX if FLET)
3	Reserved
4	Number of words available in this LET/FLET sector.
5	Sector address for the next LET/FLET sector of this cartridge. This entry is zero if this is the last LET/FLET sector on this cartridge.

LET/FLET DUMP FORMAT

The DUP control records DUMPLET or DUMPFLET are used to dump LET/FLET on the principal printer. One sector of LET/FLET is printed per page. The page is headed with the word LET or FLET, whichever is applicable. Each sector of LET/FLET dumped is preceded by two lines of header information. The first header line contains the contents of the following locations from COMMA/DCOM:

#CIDN -- Cartridge ID, Logical Drive 0, 1, 2, 3, or 4
$FPAD -- COMMA File Protect Address, Logical Drive 0, 1, 2, 3, or 4
#FPAD -- DCOM File Protect Address, Logical Drive 0, 1, 2, 3, or 4
#CIBA -- CIB Address, Logical Drive 0, 1, 2, 3, or 4
#ULET -- LET Address, Logical Drive 0, 1, 2, 3, or 4
#FLET -- FLET Address, Logical Drive 0, 1, 2, 3, or 4

Following this line will be a second header line which reflects information concerning the LET/FLET sector being dumped:

Relative Sector Number (SCTR NO.)
User Area/Fixed Area (UA/FXA)
Word Available (WORDS AVAIL)
Chain Address (CHAIN ADR)

Following these two header lines are the LET/FLET entries. Twenty one lines of entries are printed, five entries per line and sequenced by column. Each entry is formatted as follows:

Name -- 5 print positions + blank
Type code -- DSF, DCI, or DDF -- 3 print positions + blank, 4 blanks if 1DUMY or secondary entry point
DB Count -- 4 print positions + blank
DB Addr -- 4 print positions + 5 blanks

Only the name is printed for each secondary entry.

Examples of DUMPLET and DUMPFLET follow.

```
// JOB    3333

LOG DRIVE   CART SPEC   CART AVAIL   PHY DRIVE
  0000        3333        3333         0002
                          2222         0001
                          C00F         0003
                          0A88         0004

// DUP

*DEFINE FIXED AREA           5
CART ID 3333  CYLS FXA  0004  DBS AVAIL 0200  FLET SECTOR ADDR  0118

*STOREDATA  CD  FX  DATA    10
CART ID 3333   DB ADDR  1200   DB CNT   0020

*STOREDATACICD  FX  CIMGE   10
CART ID 3333   DB ADDR  1220   CB CNT   0020



*STOREDATA  CD  UA  DATA2   10
CART ID 3333   DB ADDR  1C40   CB CNT   0020

*DUMPLET


                                  LET


=CIDN   $FPAD   =FPAD   =CIBA   =ULET   =FLET
3333    01D6    01D6    0140    C150    C118

SCTR NO.  UA/FXA.   WORDS AVAIL.  CHAIN ADDR.
CC00      0158          0C00        0151


 PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB
 NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR

 FADD  DSF 0008 1580     FATN  DSF 000A 15AA     ESUB                     EATAN                    FIXIX
 FSUB                    FATAN                    EADDX                    EAXB  DSF 0005 161D      FLOAT DSF 0002 165E
 FADDX                   FAXB  DSF 0005 15B4     ESUBX                    EAXBX                    IFIX  DSF 0004 1660
 FSUBX                   FAXBX                    EAXI  DSF 0007 15F2      EEXP  DSF 000A 1622      NORM  DSF 0004 1664
 FAXI  DSF 0006 1588     FEXP  DSF C009 15B9      EAXIX                    EXPN                     SNR   DSF 0002 1668
 FAXIX                   FXPN                     EDIV  DSF 0006 15F9      ELN   DSF 000B 162C      XDD   DSF 0006 166A
 FDIV  DSF 0007 158E     FLN   DSF 000A 15C2      EDIVX                    EALOG                    XMD   DSF 0005 1670
 FDIVX                   FALOG                    EDVR  DSF 0003 15FF      ESIGN DSF 0003 1637      XMDS  DSF 0003 1675
 FDVR  DSF 0003 1595     FSIGN DSF C003 15CC      EDVRX                    ESIN  DSF 000B 163A      FBTD  DSF 0019 1678
 FDVRX                   FSIN  DSF 0009 15CF      EGETP DSF 0003 1602      ESINE                    FDTB
 FGETP DSF 0003 1598     FSINE                    ELD   DSF 0004 1605      ECOSN                    IABS  DSF 0002 1691
 FLD   DSF 0005 159B     FCOS                     ELDX                     ECOS                     XSQR  DSF 0004 1693
 FLDX                    FCOSN                    ESTO                     ESQR  DSF 0006 1645      PAUSE DSF 0003 1697
 FSTO                    FSQR  DSF 0006 15D8      ESTOX                    ESQRT                    STOP  DSF 0002 169A
 FSTOX                   FSQRT                    EMPY  DSF 0004 1609      ETANH DSF 0004 164B      SUBIN DSF 0003 169C
 FMPY  DSF 0005 15A0     FTANH DSF C005 15DE      EMPYX                    ETNH                     SUBSC DSF 0003 169F
 FMPYX                   FTNH                     ESBR  DSF 0003 160D      SEAR  DSF 0004 164F      TTEST DSF 0002 16A2
 FSBR  DSF 0003 15A5     SFAR  DSF 0004 15E3      ESBRX                    SEARX                    TSET
 FSBRX                   SFARX                    EABS  DSF 0002 1610      SEIF  DSF CC03 1653      DATSW DSF 0003 16A4
 FABS  DSF 0C02 15A8     SFIF  DSF 0003 15E7      EAVL                     FARC  DSF 0003 1656      DVCHK DSF 0002 16A7
 FAVL                    EADD  DSF 0008 15EA      EATN  DSF 000B 1612      FIXI  DSF 0005 1659      FCTST DSF 0003 16A9


=CIDN   $FPAD   =FPAD   =CIBA   =ULET   =FLET
3333    01D6    01D6    0140    0150    0118

SCTR NO.  UA/FXA.   WORDS AVAIL.  CHAIN ADDR.
C001      0158          0000        0152


 PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB
 NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR

 ISIGN DSF 0003 16AC     SIOF                     PAPTZ DSF 000F 1748      CARD1 DSF 0010 17BE      SPEED DSF 0015 18DA
 OVERF DSF 0002 16AF     SICAF                    PNCHZ DSF 0006 1757      OMPR1 DSF 0011 17CE      ZIPCO DSF 000A 18EF
 SLITE DSF 0005 16B1     SIOFX                    PRNTZ DSF 000D 175D      PAPT1 DSF 0011 17DF      BIDEC DSF 0006 18F9
 SLITT                   SCOMP                    PRNZ  DSF 000D 176A      PAPTN DSF 0013 17F0      BINDC DSF 0005 18FF
 PDUMP DSF 0009 16B6     SWRT                     READZ DSF 0005 1777      PAPTX DSF 0015 1803      BINHX DSF 0004 1904
 ISTOP DSF 0002 16BF     SRED                     TYPEZ DSF 0007 177C      PLOT1 DSF 000E 1818      CPEBC DSF 0009 1908
 ISTRT DSF 0002 16C1     SIOIX                    WRTYZ DSF 0005 1783      PNCH0 DSF 000E 1826      CPHOL DSF 0009 1911
 SDFIO DSF 0017 16C3     UFIO  DSF 001C 1720      EBCTB DSF 0005 1788      PNCH1 DSF 000E 1834      CPPT3 DSF 0009 191A
 SDAF                    URED                     GETAD DSF 0002 178D      PRNT1 DSF 0018 1842      DCBIN DSF 0006 1923
```

```
  SDAI                      UWRT                      HOLEZ DSF 0005 178F     PRNT3 DSF 000D 185A     DECBI DSF 0009 1929
  SDCOM                     UIOI                      HOLTB DSF 0004 1794     READO DSF 0007 1867     EBCCP DSF 0009 1932
  SDF                       UIOF                      SGOTO DSF 0003 1798     READ1 DSF 0008 186E     EBHOL DSF 0009 193B
  SDFX                      UIOAI                     SIAR  DSF 0004 179B     TYPEO DSF 0012 1876     EBPA  DSF 0006 1944
  SDI                       UIOAF                     SIARX                   WRTYO DSF 0009 1888     EBPT3 DSF 0009 194A
  SDIX                      UIOFX                     SIIF  DSF 0003 179F     EBPRT DSF 0007 1891     HLEBC DSF 0009 1953
  SDRED                     UIOIX                     ILSOO DSF 0003 17A2     HOLEB DSF 0009 1898     HLPT3 DSF 0009 195C
  SDWRT                     UCOMP                     ILSO1 DSF 0003 17A5     HOLPR DSF 0007 18A1     HOLCP DSF 0009 1965
  SDFND DSF 0006 16DA       BCKSP                     ILSO2 DSF 0001 17A8     HXBIN DSF 0005 18A8     HOLL  DSF 0006 196E
  SFIO  DSF 0040 16E0       EOF                       ILSO3 DSF 0003 17A9     PAPEB DSF 0010 18AD     PRTY  DSF 0006 1974
  SIOI                      REWND                     ILSO4 DSF 0002 17AC     PAPHL DSF 0010 18BD     PTHOL DSF 0009 197A
  SIOAI                     CARDZ DSF 000C 173C       CARDO DSF 0010 17AE     PAPPR DSF 000D 18CD     PT3CP DSF 0009 1983


=CIDN   $FPAD   =FPAD   =CIBA   =ULET   =FLET
3333    01D6    01D6    0140     0150    0118

SCTR NO.  UA/FXA.   WORDS AVAIL.  CHAIN ADDR.
C002      0158          0087        0118


  PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB
  NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR

  PT3EB DSF 0009 198C     ECHAR DSF 0005 1B75     POINT DSF 0008 1C05
  DMTDO DSF 001A 1995     ECHRX DSF 0026 1B7A     SCALE DSF 0002 1C0D
  DMTXO                   ECHRI                   SCALF DSF 0002 1C0F
  DMPD1 DSF 001E 19AF     VCHRI                   XYPLT DSF 0007 1C11
  DMPX1                   EGRID DSF 0008 1BA0     EBC48 DSF 0009 1C18
  DMP80 DSF 0007 19CD     ERULE DSF 000A 1BA8     HOLCA DSF 0006 1C21
  FLIPR DSF 0007 19D4     EMOVE                   HOL48 DSF 0008 1C27
  SYSUP DSF 0035 19DB     EINC                    HXCV  DSF 0004 1C2F
  ADRWS DSF 0010 1A10     EPLOT DSF 0005 1BB2     PRNT2 DSF 001E 1C33
  COPY  DSF 001D 1A20     FCHAR DSF 0005 1BB7     SCAT1 DSF 0041 1C51
  DISC  DSF 0036 1A3D     FCHRX DSF 0025 1BBC     STRTB DSF 0006 1C92
  DLCIB DSF 001D 1A73     FCHRI                   SCAT3 DSF 004B 1C98
  DSLET DSF 0036 1A90     WCHRI                   KG3   DSF 0006 1CE3
  ID    DSF 001A 1AC6     FGRID DSF 0008 1BE1     KG4   DSF 0009 1CE9
  IDENT DSF 000C 1AE0     FRULE DSF 0009 1BE9     SCAT2 DSF 0040 1CF2
  MODIF DSF 0059 1AEC     FMOVE                   1DUMY       000E 1D32
  PTLTL DSF 0009 1B45     FINC                    DATA2 DDF 0020 1D40
  CALPR DSF 0007 1B4E     FPLOT DSF 0004 1BF2     1DUMY       46A0 1D60
  FSLEN DSF 000B 1B55     PLOTI DSF 0003 1BF6
  FSYSU                   PLOTS
  RDREC DSF 0015 1B60     PLOTX DSF 000C 1BF9


                                  FLET


=CIDN   $FPAD   =FPAD   =CIBA   =ULET   =FLET
3333    01D6    01D6    0140     0150    0118

SCTR NO.  UA/FXA.   WORDS AVAIL.  CHAIN ADDR.
0010      0120          0132        C000


  PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB
  NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR

  DATA  DDF 0020 1200
  CIMGE DCI 0020 1220
  1DUMY       01C0 1240


  END OF DUMPLET/FLET

*DUMPFLET


=CIDN   $FPAD   =FPAD   =CIBA   =ULET   =FLET
3333    01D6    01D6    0140     0150    0118

SCTR NO.  UA/FXA.   WORDS AVAIL.  CHAIN ADDR.
0010      0120          0132        0000


  PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB       PROG  FOR DB   DB
  NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR     NAME  MAT CNT  ADDR

  DATA  DDF 0020 1200
  CIMGE DCI 0020 1220
  1DUMY       01C0 1240
```

APPENDIX H. RESIDENT MONITOR (INCLUDING TABLE OF EQUIVALENCES)

The contents of this appendix are not to be construed as an external specification, i.e., the location in this listing may be changed. $PRET, $IREQ, $EXIT, $LINK, and $DUMP are the only locations that are guaranteed.

Note that = is equivalent to # and ' (apostrophe) is equivalent to @.

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                   0001    * RLTV ADDR* SYMBOL* DESCRIPTION                         PMN00010
                   0002    *          *        *                                    PMN00020
                   0003    * 0-3      *        * RESERVED FOR EVEN BOUNDARIES       PMN00030
                   0004    * 4-5      * #NAME  * NAME OF PROGRAM/CORE LOAD          PMN00040
                   0005    * 6        * #DBCT  * BLOCK COUNT OF PROG/CORE LOAD      PMN00050
                   0006    * 7        * #FCNT  * #FILES SWITCH--ZERO MEANS NO       PMN00060
                   0007    *          *        * FILES HAVE BEEN EQUATED            PMN00070
                   0008    * 8        * #SYSC  * SYS/NON-SYS CARTRIDGE INDR        PMN00080
                   0009    * 9        * #JBSW  * JOBT SWITCH-- NON-ZERO MEANS      PMN00090
                   0010    *          *        * TEMPORARY MODE                     PMN00100
                   0011    * 10       * #CBSW  * CLB-RETURN-TO-DUP SWITCH--        PMN00110
                   0012    *          *        * ZERO=CLB RETURN TO SUPV            PMN00120
                   0013    * 11       * #LCNT  * NO. OF LOCALS                     PMN00130
                   0014    * 12       * #MPSW  * CORE MAP SWITCH--ZERO MEANS       PMN00140
                   0015    *          *        * DO NOT PRINT A CORE MAP            PMN00150
                   0016    * 13       * #MDF1  * NO. DUP CTRL RECDS (MODIF)        PMN00160
                   0017    * 14       * #MDF2  * ADDR OF MODIF BUFFER              PMN00170
                   0018    * 15       * #NCNT  * NO. OF NOCALS                     PMN00180
                   0019    * 16       * #ENTY  * RLTV ENTRY ADDR OF PROGRAM        PMN00190
                   0020    * 17       * #RP67  * 1442-5 SW (0#1442-5 ON SYSTEM     PMN00200
                   0021    * 18       * #TODR  * 'TO' WORKING STG DRIVE CODE       PMN00210
                   0022    * 19       * #FRDR  * 'FROM' WORKING STG DRIVE CODE     PMN00220
                   0023    * 20       * #FHOL  * ADDR OF LARGEST HOLE IN FXA       PMN00230
                   0024    * 21       * #FSZE  * BLK CNT OF LARGEST HOLE IN FXA    PMN00240
                   0025    * 22       * #UHOL  * ADDR OF LARGEST HOLE IN UA        PMN00250
                   0026    * 23       * #USZE  * BLK CNT OF LARGEST HOLE IN UA     PMN00260
                   0027    * 24       * #DCSW  * DUP CALL SW--NON-ZERO#DUP CALL    PMN00270
                   0028    * 25       * #PIOD  * PRINCIPAL I/O DEVICE INDICATOR    PMN00280
                   0029    * 26       * #PPTR  * PRINC. PRINT DEVICE INDICATOR     PMN00290
                   0030    * 27       * #CIAD  * RLTV ADDR IN 'STRT OF CIL ADDR    PMN00300
                   0031    * 28       * #ACIN  * AVAILABLE CARTRIDGE INDICAT2-2    PMN00310
                   0032    * 29       * #GRPH  * 2250 INDICATOR              2G2   PMN00320
                   0033    * 30       * #GCNT  * NO. G2250 RECORDS           2G2   PMN00330
                   0034    * 31       * #LOSW  * LOCAL-CANNOT-CALL-LOCAL SW  2-2   PMN00340
                   0035    * 32       * #X3SW  * SPECIAL ILS SWITCH          2-2   PMN00350
                   0036    * 33       * #ECNT  * NO. OF *EQUAT RCDS          2-4   PMN00355
                   0037    * 33-34    *        * RESERVED FOR FUTURE USE     2-2   PMN00360
                   0038    * 35       * #ANDU  * 1+BLOCK ADDR OF END OF USER       PMN00370
                   0039    *          *        * AREA (ADJUSTED) LOGICAL DR 0       PMN00380
                   0040    * 36       *        * 1+BLOCK ADDR OF END OF USER        PMN00390
                   0041    *          *        * AREA (ADJUSTED) LOGICAL DR 1       PMN00400
                   0042    * 37       *        * 1+BLOCK ADDR OF END OF USER        PMN00410
                   0043    *          *        * AREA (ADJUSTED) LOGICAL DR 2       PMN00420
                   0044    * 38       *        * 1+BLOCK ADDR OF END OF USER        PMN00430
                   0045    *          *        * AREA (ADJUSTED) LOGICAL DR 3       PMN00440
                   0046    * 39       *        * 1+BLOCK ADDR OF END OF USER        PMN00450
                   0047    *          *        * AREA (ADJUSTED) LOGICAL DR 4       PMN00460
                   0048    * 40       * #BNDU  * 1+BLOCK ADDR OF END OF USER       PMN00470
                   0049    *          *        * AREA (BASE) LOGICAL DRIVE  0       PMN00480
                   0050    * 41       *        * 1+BLOCK ADDR OF END OF USER        PMN00490
                   0051    *          *        * AREA (BASE) LOGICAL DRIVE  1       PMN00500
                   0052    * 42       *        * 1+BLOCK ADDR OF END OF USER        PMN00510
                   0053    *          *        * AREA (BASE) LOGICAL DRIVE  2       PMN00520
                   0054    * 43       *        * 1+BLOCK ADDR OF END OF USER        PMN00530
                   0055    *          *        * AREA (BASE) LOGICAL DRIVE  3       PMN00540
                   0056    * 44       *        * 1+BLOCK ADDR OF END OF USER        PMN00550
                   0057    *          *        * AREA (BASE) LOGICAL DRIVE  4       PMN00560
                   0058    * 45       * #FPAD  * FILE PROTECT ADDR, LOGICAL        PMN00570
                   0059    *          *        * DRIVE 0 (BASE)                     PMN00580
                   0060    * 46       *        * FILE PROTECT ADDR, LOGICAL        PMN00590
                   0061    *          *        * DRIVE 1 (BASE)                     PMN00600
                   0062    * 47       *        * FILE PROTECT ADDR, LOGICAL        PMN00610
                   0063    *          *        * DRIVE 2 (BASE)                     PMN00620
                   0064    * 48       *        * FILE PROTECT ADDR, LOGICAL        PMN00630
                   0065    *          *        * DRIVE 3 (BASE)                     PMN00640
                   0066    * 49       *        * FILE PROTECT ADDR,LOGICAL         PMN00650
                   0067    *          *        * DRIVE 4 (BASE)                     PMN00660
                   0068    * 50       * #PCID  * CARTRIDGE ID, PHYSICAL DRIVE 0    PMN00670
                   0069    * 51       *        * CARTRIDGE ID, PHYSICAL DRIVE 1    PMN00680
                   0070    * 52       *        * CARTRIDGE ID, PHYSICAL DRIVE 2    PMN00690
                   0071    * 53       *        * CARTRIDGE ID, PHYSICAL DRIVE 3    PMN00700
                   0072    * 54       *        * CARTRIDGE ID, PHYSICAL DRIVE 4    PMN00710
                   0073    * 55       * #CIDN  * CARTRIDGE ID, LOGICAL DRIVE 0     PMN00720
                   0074    * 56       *        * CARTRIDGE ID, LOGICAL DRIVE 1     PMN00730
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO
                  0075     * 57       *       * CARTRIDGE ID, LOGICAL DRIVE 2  PMN00740
                  0076     * 58       *       * CARTRIDGE ID, LOGICAL DRIVE 3  PMN00750
                  0077     * 59       *       * CARTRIDGE ID, LOGICAL DRIVE 4  PMN00760
                  0078     * 60       * #CIBA * SCTR ADDR OF CIB, LOGICAL DR 0 PMN00770
                  0079     * 61       *       * SCTR ADDR OF CIB, LOGICAL DR 1 PMN00780
                  0080     * 62       *       * SCTR ADDR OF CIB, LOGICAL DR 2 PMN00790
                  0081     * 63       *       * SCTR ADDR OF CIB, LOGICAL DR 3 PMN00800
                  0082     * 64       *       * SCTR ADDR OF CIB, LOGICAL DR 4 PMN00810
                  0083     * 65       * #SCRA * SCRA, LOGICAL DRIVE 0          PMN00820
                  0084     * 66       *       * SCRA, LOGICAL DRIVE 1          PMN00830
                  0085     * 67       *       * SCRA, LOGICAL DRIVE 2          PMN00840
                  0086     * 68       *       * SCRA, LOGICAL DRIVE 3          PMN00850
                  0087     * 69       *       * SCRA, LOGICAL DRIVE 4          PMN00860
                  0088     * 70       * #FMAT * FORMAT OF PROG IN WS, DRIVE 0  PMN00870
                  0089     * 71       *       * FORMAT OF PROG IN WS, DRIVE 1  PMN00880
                  0090     * 72       *       * FORMAT OF PROG IN WS, DRIVE 2  PMN00890
                  0091     * 73       *       * FORMAT OF PROG IN WS, DRIVE 3  PMN00900
                  0092     * 74       *       * FORMAT OF PROG IN WS, DRIVE 4  PMN00910
                  0093     * 75       * #FLET * FLET SCTR ADDR, LOGICAL DR 0   PMN00920
                  0094     * 76       *       * FLET SCTR ADDR, LOGICAL DR 1   PMN00930
                  0095     * 77       *       * FLET SCTR ADDR, LOGICAL DR 2   PMN00940
                  0096     * 78       *       * FLET SCTR ADDR, LOGICAL DR 3   PMN00950
                  0097     * 79       *       * FLET SCTR ADDR, LOGICAL DR 4   PMN00960
                  0098     * 80       * #ULET * LET SCTR ADDR, LOGICAL DR 0    PMN00970
                  0099     * 81       *       * LET SCTR ADDR, LOGICAL DR 1    PMN00980
                  0100     * 82       *       * LET SCTR ADDR, LOGICAL DR 2    PMN00990
                  0101     * 83       *       * LET SCTR ADDR, LOGICAL DR 3    PMN01000
                  0102     * 84       *       * LET SCTR ADDR, LOGICAL DR 4    PMN01010
                  0103     * 85       * #WSCT * BLK CNT OF PROG IN WS, DRIVE 0 PMN01020
                  0104     * 86       *       * BLK CNT OF PROG IN WS, DRIVE 1 PMN01030
                  0105     * 87       *       * BLK CNT OF PROG IN WS, DRIVE 2 PMN01040
                  0106     * 88       *       * BLK CNT OF PROG IN WS, DRIVE 3 PMN01050
                  0107     * 89       *       * BLK CNT OF PROG IN WS, DRIVE 4 PMN01060
                  0108     * 90       * #CSHN * SCTR CNT CUSHION,LOGICAL DR 0  PMN01070
                  0109     * 91       *       * SCTR CNT CUSHION,LOGICAL DR 1  PMN01080
                  0110     * 92       *       * SCTR CNT CUSHION,LOGICAL DR 2  PMN01090
                  0111     * 93       *       * SCTR CNT CUSHION,LOGICAL DR 3  PMN01100
                  0112     * 94       *       * SCTR CNT CUSHION,LOGICAL DR 4  PMN01110
                  0113     * 95-319   *       * RESERVED FOR FUTURE USE        PMN01120

RESIDENT MONITOR

ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                  0115     ************************************************** PMN01140
                  0116     *                                                * PMN01150
                  0117     *STATUS-VERSION 2, MODIFICATION 4                * PMN01160
                  0118     *                                                * PMN01170
                  0119     *FUNCTION/OPERATION-                             * PMN01180
                  0120     *   THIS SECTION ALWAYS REMAINS IN CORE. IT      * PMN01190
                  0121     *   IS COMPRISED OF THE COMMUNICATIONS           * PMN01200
                  0122     *   AREA (COMMA), THE SKELETON SUPERVISOR, AND   * PMN01210
                  0123     *   A DISK I/O SUBROUTINE, NOMINALLY DISKZ. (THE * PMN01220
                  0124     *   FIRST TWO OF THESE SECTIONS ARE INTERMIXED.) * PMN01230
                  0125     *   COMMA CONTAINS THE SYSTEM PARAMETERS REQUIR- * PMN01240
                  0126     *   ED TO FETCH A CORE LOAD IN CORE IMAGE FOR-   * PMN01250
                  0127     *   MAT. THE SKELETON SUPERVISOR PROVIDES IN-    * PMN01260
                  0128     *   STRUCTIONS FOR INITIATING A CALL EXIT, A     * PMN01270
                  0129     *   CALL LINK, A DUMP-TO-PRINTER OR A CALL TO THE * PMN01280
                  0130     *   AUXILIARY SUPERVISOR.  IN ADDITION, THE SKELE-* PMN01290
                  0131     *   TON SUPERVISOR CONTAINS SEVERAL TRAPS FOR CER-* PMN01300
                  0132     *   TAIN I/O FUNCTIONS/CONDITIONS. THE DISK I/O  * PMN01310
                  0133     *   SECTION CONSISTS OF A SUBROUTINE FOR READING * PMN01320
                  0134     *   FROM OR WRITING ON A DISK CARTRIDGE ON A     * PMN01330
                  0135     *   GIVEN LOGICAL DISK DRIVE.                    * PMN01340
                  0136     *                                                * PMN01350
                  0137     *ENTRY POINTS-                                   * PMN01360
                  0138     *   * $PRET-A TRAP FOR PREOPERATIVE I/O ERRORS.  * PMN01370
                  0139     *           THE CALLING SEQUENCE IS              * PMN01380
                  0140     *              BSI   L  $PRET                    * PMN01390
                  0141     *   * $PSTX-A POSTOPERATIVE ERROR TRAP FOR I/O   * PMN01400
                  0142     *           DEVICES ON LEVEL X (X#1,2,3,OR 4).   * PMN01410
                  0143     *           THE CALLING SEQUENCE IS              * PMN01420
                  0144     *              BSI  L  $PSTX                     * PMN01430
                  0145     *   * $STOP-THE PROGRAM STOP KEY TRAP.           * PMN01440
                  0146     *   * $EXIT-THE ENTRY POINT FOR THE EXIT/CALL    * PMN01450
                  0147     *         EXIT STATEMENT. THE CALLING SEQUENCE IS* PMN01460
                  0148     *            LDX   0 $EXIT                       * PMN01470
                  0149     *   * $LINK-THE ENTRY POINT FOR THE LINK/CALL    * PMN01480
                  0150     *         LINK STATEMENT. THE CALLING SEQUENCE IS* PMN01490
                  0151     *            BSI  L  $LINK                       * PMN01500
                  0152     *   * $DUMP-THE ENTRY POINT FOR THE DUMP/PDMP    * PMN01510
                  0153     *         STATEMENT. THE CALLING SEQUENCE IS     * PMN01520
                  0154     *            BSI  L  $DUMP                       * PMN01530
                  0155     *            DC      FORMAT                      * PMN01540
                  0156     *            DC      LIMIT1                      * PMN01550
                  0157     *            DC      LIMIT2                      * PMN01560
                  0158     *         WHERE LIMIT1 AND LIMIT2 ARE THE LIMITS * PMN01570
                  0159     *         BETWEEN WHICH THE DUMP IS TO OCCUR, AND* PMN01580
                  0160     *         FORMAT IS A CODE INDICATING THE FORMAT * PMN01590
                  0161     *         OF THE DUMP. IF FORMAT IS NEGATIVE,    * PMN01600
                  0162     *         THE AUXILIARY SUPERVISOR IS FETCHED    * PMN01610
                  0163     *         AND CONTROL PASSED TO IT.              * PMN01620
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                  ID/SEQNO

                    0164   *    * DZ000-ENTERED WHEN THE CALLER WISHES TO      * PMN01630
                    0165   *           PERFORM A DISK I/O OPERATION. THE       * PMN01640
                    0166   *           CALLING SEQUENCE VARIES WITH THE        * PMN01650
                    0167   *           VERSION OF THE DISK I/O SUBROUTINE.     * PMN01660
                    0168   *    * $I200/$I400-ENTERED WHEN THE OPERATION-      * PMN01670
                    0169   *                  COMPLETE INTERRUPT OCCURS ON     * PMN01680
                    0170   *                  LEVEL 2/4.                       * PMN01690
                    0171   *                                                   * PMN01700
                    0172   *INPUT-N/A                                          * PMN01710
                    0173   *                                                   * PMN01720
                    0174   *OUTPUT-WORDS 6-4090 SAVED ON THE CIB ON A CALL     * PMN01730
                    0175   *       DUMP                                        * PMN01740
                    0176   *                                                   * PMN01750
                    0177   *EXTERNAL REFERENCES-N/A                            * PMN01760
                    0178   *                                                   * PMN01770
                    0179   *EXITS-                                             * PMN01780
                    0180   *    * NORMAL                                       * PMN01790
                    0181   *       *THE EXITS FROM THE SUBROUTINES AT $PRET    * PMN01800
                    0182   *        $PST1, $PST2, $PST3, $PST4, AND $STOP      * PMN01810
                    0183   *        ARE BRANCH INSTRUCTIONS FOLLOWING A        * PMN01820
                    0184   *        WAIT INSTRUCTION. $STOP TURNS OFF IN-      * PMN01830
                    0185   *        TERRUPT LEVEL 5 AFTER THE START KEY IS     * PMN01840
                    0186   *        DEPRESSED.                                 * PMN01850
                    0187   *       *THE EXITS FROM $EXIT,$LINK,AND $DUMP ARE   * PMN01860
                    0188   *        TO THE CORE IMAGE LOADER, PHASE 1,         * PMN01870
                    0189   *        AFTER THAT PHASE HAS BEEN FETCHED.         * PMN01880
                    0190   *       *THE EXIT FROM DZ000 IS BACK TO THE         * PMN01890
                    0191   *        CALLER AFTER THE REQUESTED DISK OPERA-     * PMN01900
                    0192   *        TION HAS BEEN INITIATED.                   * PMN01910
                    0193   *       *THE EXITS FROM $I200/$I400 ARE BACK TO     * PMN01920
                    0194   *        THE ADDRESSES FROM WHICH THE DISK OP-      * PMN01930
                    0195   *        ERATION COMPLETE INTERRUPT OCCURED         * PMN01940
                    0196   *        AFTER THE INTERRUPT HAS BEEN SERVICED      * PMN01950
                    0197   *        BY THE APPROPRIATE ISS.                    * PMN01960
                    0198   *    * ERROR-N/A                                    * PMN01970
                    0199   *                                                   * PMN01980
                    0200   *TABLES/WORK AREAS-                                 * PMN01990
                    0201   *    * $ACDE                                        * PMN02000
                    0202   *    * $CH12                                        * PMN02010
                    0203   *    * $CILA                                        * PMN02020
                    0204   *    * $CLSW                                        * PMN02030
                    0205   *    * $COMN                                        * PMN02040
                    0206   *    * $CORE                                        * PMN02050
                    0207   *    * $CTSW                                        * PMN02060
                    0208   *    * $CXR1                                        * PMN02070
                    0209   *    * $CYLN                                        * PMN02080
                    0210   *    * $DADR                                        * PMN02100
                    0211   *    * $DBSY                                        * PMN02110
                    0212   *    * $DCYL                                        * PMN02120
                    0213   *    * $DMPF                                        * PMN02130
                    0214   *    * $DREQ                                        * PMN02140
                    0215   *    * $FPAD                                        * PMN02150
                    0216   *    * $GCOM                                2G2     * PMN02160
                    0217   *    * $GRIN                                2G2     * PMN02170
                    0218   *    * $HASH                                        * PMN02180
                    0219   *    * $IBT2                                        * PMN02190
                    0220   *    * $IBT4                                        * PMN02200
                    0221   *    * $IBSY                                        * PMN02210
                    0222   *    * $IOCT                                        * PMN02220
                    0223   *    * $KCSW                                        * PMN02230
                    0224   *    * $LAST                                        * PMN02240
                    0225   *    * $NDUP                                        * PMN02250
                    0226   *    * $NXEQ                                        * PMN02260
                    0227   *    * $PBSY                                        * PMN02270
                    0228   *    * $PGCT                                        * PMN02280
                    0229   *    * $PHSE                                        * PMN02290
                    0230   *    * $RMSW                                        * PMN02300
                    0231   *    * $SCAT                                2-4     * PMN02305
                    0232   *    * $SNLT                                        * PMN02310
                    0233   *    * $UFIO                                        * PMN02320
                    0234   *    * $ULET                                        * PMN02330
                    0235   *    * $WRD1                                        * PMN02340
                    0236   *    * $WSDR                                        * PMN02350
                    0237   *    * $XR3X                                2-2     * PMN02360
                    0238   *                                                   * PMN02370
                    0239   *ATTRIBUTES-REUSABLE                               * PMN02380
                    0240   *                                                   * PMN02390
                    0241   *NOTES-                                             * PMN02400
                    0242   *    THERE ARE WAIT INSTRUCTIONS AT $PRET+1,        * PMN02410
                    0243   *    $STOP+1, AND $PSTX+1. DEPRESSING THE START     * PMN02420
                    0244   *    KEY WILL RETURN CONTROL TO THE CALLER IN ALL   * PMN02430
                    0245   *    CASES.                                         * PMN02440
                    0246   ************************************************** PMN02450
```

```
ADDR REL OBJECT   ST.NO.  LABEL OPCD FT OPERANDS                                    ID/SEQNO

                  0248    * PROVIDE PARAMETERS FOR SYSTEM LOADER                   PMN02470
                  0249    *                                                        PMN02480
                  0250          ABS                                                PMN02490
0280              0251          ORG     4                                          PMN02500
0004 0  0FFA      0252          DC      4095-* WD CNT FOR WRITING CORE ON CIB      PMN02510
0005 0  0000      0253    $CIBA DC      *-*  SCTR ADDR OF THE CIB                  PMN02520
0006 0  0000      0254    $CH12 DC      *-*  ADDR OF CHANNEL 12 INDICATOR          PMN02530
0007 0  0000      0255    $COMN DC      *-*  LENGTH OF COMMON (IN WORDS)           PMN02540
                  0256    *                                                        PMN02550
                  0257    * ULTIMATE RESIDENCE OF THE INTERRUPT TV                 PMN02560
                  0258    *                                                        PMN02570
0008 0  0000      0259    $LEV0 DC      *-*   LEVEL 0 BRANCH ADDRESS               PMN02580
0009 0  0000      0260    $LEV1 DC      *-*   LEVEL 1 BRANCH ADDRESS               PMN02590
000A 0  00B3      0261    $LEV2 DC      $I200 LEVEL 2 BRANCH ADDR                  PMN02600
000B 0  0000      0262    $LEV3 DC      *-*   LEVEL 3 BRANCH ADDRESS               PMN02610
000C 0  00C4      0263    $LEV4 DC      $I400 LEVEL 4 BRANCH ADDR                  PMN02620
000D 0  0091      0264    $LEV5 DC      $STOP LEVEL 5 BRANCH ADDR                  PMN02630
                  0265    *                                                        PMN02640
                  0266    *                                                        PMN02650
000E 0  0000      0267    $CORE DC      *-*  SIZE OF CORE, E.G., 4096=4K          PMN02660
000F 0  0000      0268    $CTSW DC      *-*  CONTROL RECORD TRAP SWITCH            PMN02670
0010 0  0000      0269    $DADR DC      *-*  SCTR ADDR OF PROG TO BE LOADED        PMN02680
0011 0  0000      0270    $SCAT DC      *-*  NON ZERO=SCA INTRPT PNDNG     2-4     PMN02690
0012 0  0000      0271    $DREQ DC      *-*  IND. FOR REQUESTED VERSION DKI/O      PMN02700
0013 0  0000      0272    $IBSY DC      *-*  NON-ZERO IF CD/PAP TP DEV. BUSY       PMN02710
0014      000C    0273    $HASH BSS  E  12   WORK AREA                             PMN02720
                  0274    *                                                        PMN02730
                  0275    *                                                        PMN02740
0020      0008    0276    $SCAN BSS     8    1132 SCAN AREA                 32     PMN02750
                  0277    *                                                        PMN02760
                  0278    *                                                        PMN02770
                  0279    *                                                        PMN02780
                  0280    * TRAP FOR PREOPERATIVE I/O ERRORS                       PMN02790
                  0281    *                                                        PMN02800
0028 0  0000      0282    $PRET DC      *-*       ENTRY POINT                      PMN02810
0029 0  3000      0283          WAIT              WAIT TIL START KEY PUSHED        PMN02820
002A 00 4C800028  0284          BSC  I  $PRET     RETURN TO CALLER                 PMN02830
                  0285    *                                                        PMN02840
                  0286    *                                                        PMN02850
002C 0  0000      0287    $IREQ DC      *-*  ADDR OF INT REQUEST SUBROUTINE       PMN02860
002D 0  0000      0288    $ULET DC      *-*  ADDR OF LET, LOGICAL DR 0             PMN02870
002E 0  0000      0289          DC      *-*  ADDR OF LET, LOGICAL DR 1             PMN02880
002F 0  0000      0290          DC      *-*  ADDR OF LET, LOGICAL DR 2             PMN02890
0030 0  0000      0291          DC      *-*  ADDR OF LET, LOGICAL DR 3             PMN02900
0031 0  0000      0292          DC      *-*  ADDR OF LET, LOGICAL DR 4             PMN02910
0032 0  0000      0293    $IOCT DC      *-*  ZERO IF NO I/O IN PROGRESS     50      PMN02920
0033 0  0000      0294    $LAST DC      *-*  NON-ZERO WHEN LAST CARD SENSED        PMN02930
0034 0  0000      0295    $NDUP DC      *-*  DO NOT DUP IF NON-ZERO                PMN02940
0035 0  0000      0296    $NXEQ DC      *-*  DO NOT EXECUTE IF NON-ZERO            PMN02950
0036 0  0000      0297    $PBSY DC      *-*  NON-ZERO WHEN PRINTER BUSY            PMN02960
0037 0  0000      0298    $PGCT DC      *-*  PAGE NO. FOR HEADINGS                 PMN02970
                  0299    *                                                        PMN02980
                  0300    * CALL EXIT ENTRY POINT TO SKELETON SUPERVISOR           PMN02990
                  0301    *                                                        PMN03000
0038 0  7019      0302    $EXIT MDX     $S000     BR TO FETCH CIL, PHASE 1 56      PMN03010
                  0303    *                                                        PMN03020
                  0304    *** CALL LINK ENTRY POINT                                PMN03030
                  0305    *                                                        PMN03040
0039 0  0000      0306    $LINK DC      *-*       ENTRY POINT              57      PMN03050
003A 0  1810      0307          SRA     16                                         PMN03060
003B 0  7017      0308          MDX     $S100     BR TO FETCH CIL, PHASE 1         PMN03070
003C      0000    0309          BSS  E  0                                          PMN03080
003C 0  0001      0310    $S900 DC      1    DISK PARAMETERS FOR SAVING CORE       PMN03090
003D 0  0004      0311          DC      $CIBA-1 *IN CONNECTION WITH DUMP          PMN03100
003E 0  FFFF      0312    $S910 DC      -1   CALL EXIT INDICATOR                  PMN03110
                  0313    *                                                        PMN03120
                  0314    *** SAVE 1ST 4K OF CORE ON THE CIB                       PMN03130
                  0315    *                                                        PMN03140
003F 0  0000      0316    $DUMP DC      *-*       ENTRY POINT              63      PMN03150
0040 0  D8D9      0317          STD     $ACEX     SAVE ACCUMULATOR, EXTENSION      PMN03180
0041 0  4009      0318          BSI     $S250     CHK PNDNG INTRPT         2-4     PMN03185
0042 0  69D4      0319          STX   1 $CXR1     SAVE XR1                         PMN03190
0043 00 C480003F  0320          LD   I  $DUMP                                      PMN03200
0045 0  D0D3      0321          STO     $DMPF     SAVE DUMP FORMAT CODE            PMN03210
0046 0  C8F5      0322          LDD     $S900                                      PMN03220
0047 00 440000F2  0323          BSI  L  DZ000     SAVE WDS 6-4095 ON CIB           PMN03230
0049 0  C0F2      0324          LD      $S900                                      PMN03240
004A 0  7008      0325          MDX     $S100     BR TO FETCH CIL, PHASE 1         PMN03250
                  0326    *                                                        PMN03251
                  0327    *** SUBR TO  CHECK IF ANY INTRPT IS PENDING              PMN03252
                  0328    *                                                        PMN03253
```

```
ADDR RFL OBJECT   ST.NO.  LABEL OPCD FT OPERANDS                                  ID/SEQNO

0048 0  0000      0329    $S250 DC      *-*       ENTRY POINT                     PMN03254
004C 0  C0E5      0330    $S300 LD      $IOCT     IS THERE INTRPT PNDNG           PMN03255
004D 0  E8C3      0331          OR      $SCAT     *OR SCA INTRPT PNDNG            PMN03256
004E 00 4C20004C  0332          BSC  L  $S300,Z   *THEN BR, IF ALL INTRPT         PMN03257
0050 00 4C80004B  0333          BSC  I  $S250     *IS SERVICED-RETURN             PMN03258
                  0335    *                                                       PMN03270
                  0336    *** FETCH CORE IMAGE LOADER, PHASE 1                    PMN03280
                  0337    *                                                       PMN03290
0052 0  C0EB      0338    $S000 LD      $S910                                     PMN03300
0053 0  D0C2      0339    $S100 STO     $RMSW     SAVE EXIT-LINK-DUMP SWITCH      PMN03310
0054 00 65800039  0340          LDX  I1 $LINK     LINK ADDR TO XR1                PMN03350
0056 0  C101      0341          LD    1 1         FETCH 2ND WD OF LINK NAME       PMN03360
0057 0  18D0      0342          RTE     16                                        PMN03370
0058 0  C100      0343          LD    1 0         FETCH 1ST WD OF LINK NAME       PMN03380
                  0344    * $S150+1 CONTAINS ADDR LAST WD OF DISK I/O MINUS 3     PMN03400
0059 00 65000000  0345    $S150 LDX  L1 *-*       ADDR END OF DKI/O-1 TO XR1      PMN03410
005B 0  D888      0346          STD     $LKNM     SAVE LINK NAME                  PMN03415
005C 0  40EE      0347          BSI     $S250     CHK ANY PNDNG INTRPT     2-4    PMN03417
005D 0  C0FC      0348          LD      $CILA                                     PMN03420
005E 0  1890      0349    $S200 SRT     16                                        PMN03430
005F 00 440000F2  0350          BSI  L  DZ000     FETCH CI LOADER, PHASE 1        PMN03440
0061 0  40E9      0351          BSI     $S250     CHK DISK OP FINISHED     2-4    PMN03460
0062 0  4102      0352          BSI   1 2         BR TO CI LOADER, PHASE 1        PMN03470
                  0353    *                                                       PMN03480
0063 0  0000      0354    $GCON DC      *-*  GRAPHIC SUBR PACKAGE INDR     2G2    PMN03490
0064 0  0000      0355    $GRIN DC      *-*  GRAPHIC INITLZN PROGRAM INDR  2G2    PMN03500
                  0356    *                                                  2G2    PMN03510
0065      0003    0357          BSS     3         RESERVED FOR 2250       2G2    PMN03520
0068      0009    0358          BSS     9         PATCH AREA                       PMN03530
                  0359    *                                                       PMN03540
0071 0  0000      0360    $FLSH DC      *-*  FLUSH-TO-NEXT-JOB SWITCH 1#FLUSH     PMN03550
0072      0000    0361          BSS  E  0                                         PMN03560
0072 0  0000      0362    $CWCT DC      *-*  WORD COUNT AND SECTOR ADDRESS        PMN03570
0073 0  0000      0363          DC      *-*  *FOR SAVING/RESTORING COMMON         PMN03580
0074 0  0000      0364    $CCAD DC      *-*  ADDR FOR SAVING/RESTORING COMMON     PMN03590
0075 0  0000      0365    $LSAD DC      *-*  SCTR ADDR OF 1ST LOCAL/SOCAL         PMN03600
0076 0  0000      0366    $DZIN DC      *-*  DISKZ/1/N INDICATOR (-1,0,+1)        PMN03610
0077 0  0000      0367    $DCDE DC      *-*  LOGICAL DRIVE CODE FOR PROGRAM       PMN03620
0078 0  0000      0368    $PHSE DC      *-*  NO. OF PHASE NOW IN CORE             PMN03630
0079 0  0000      0369    $UFIO DC      *-*  UNFORMATTED I/O RECORD NO.           PMN03640
007A 0  0000      0370    $WSDR DC      *-*  WORKING STORAGE DRIVE CODE           PMN03650
007B 0  0000      0371    $WRD1 DC      *-*  LOADING ADDR OF THE CORE LOAD        PMN03660
007C 0  0000      0372    $KCSW DC      *-*  1 IF KB,CP BOTH UTILIZED             PMN03670
007D 0  0000      0373    $UFDR DC      *-*  UNFORMATTED I/O DRIVE CODE           PMN03680
007E 0  0000      0374    $CPTR DC      *-*  CHANNEL 12 INDICATOR FOR CP          PMN03690
007F 0  0000      0375    $1132 DC      *-*  CHANNEL 12 INDICATOR FOR 1132        PMN03700
0080 0  0000      0376    $1403 DC      *-*  CHANNEL 12 INDICATOR FOR 1403        PMN03710
                  0378    * TRAP FOR POSTOPERATIVE I/O ERRORS ON LEVEL 1          PMN03730
                  0379    *                                                       PMN03740
0081 0  0000      0380    $PST1 DC      *-*       ENTRY POINT                     PMN03750
0082 0  3000      0381          WAIT                                              PMN03760
0083 00 4C800081  0382          BSC  I  $PST1     RETURN TO DEVICE SUBROUTINE     PMN03770
                  0383    *                                                       PMN03780
                  0384    * TRAP FOR POSTOPERATIVE I/O ERRORS ON LEVEL 2          PMN03790
                  0385    *                                                       PMN03800
0085 0  0000      0386    $PST2 DC      *-*       ENTRY POINT                     PMN03810
0086 0  3000      0387          WAIT                                              PMN03820
0087 00 4C800085  0388          BSC  I  $PST2     RETURN TO DEVICE SUBROUTINE     PMN03830
                  0389    *                                                       PMN03840
                  0390    * TRAP FOR POSTOPERATIVE I/O ERRORS ON LEVEL 3          PMN03850
                  0391    *                                                       PMN03860
0089 0  0000      0392    $PST3 DC      *-*       ENTRY POINT                     PMN03870
008A 0  3000      0393          WAIT                                              PMN03880
008B 00 4C800089  0394          BSC  I  $PST3     RETURN TO DEVICE SUBROUTINE     PMN03890
                  0395    *                                                       PMN03900
                  0396    * TRAP FOR POSTOPERATIVE I/O ERRORS ON LEVEL 4          PMN03910
                  0397    *                                                       PMN03920
008D 0  0000      0398    $PST4 DC      *-*       ENTRY POINT                     PMN03930
008E 0  3000      0399          WAIT                                              PMN03940
008F 00 4C80008D  0400          BSC  I  $PST4     RETURN TO DEVICE SUBROUTINE     PMN03950
                  0401    *                                                       PMN03960
                  0402    *                                                       PMN03970
                  0403    * PROGRAM STOP KEY TRAP                                 PMN03980
                  0404    *                                                       PMN03990
0091 0  0000      0405    $STOP DC      *-*       ENTRY POINT                     PMN04000
0092 0  3000      0406          WAIT              WAIT TIL START KEY PUSHED       PMN04010
0093 00 4CC00091  0407          BOSC I  $STOP     RETURN TO CALLER                PMN04020
```

```
ADDR REL OBJECT    ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                   0409     *                                                         PMN04040
                   0410     * PARAMETERS USED BY THE DISK I/O SUBROUTINES. THE       PMN04050
                   0411     * LOGICAL DRIVE CODE IS FOUND IN BITS 1-3 FOR ALL       PMN04060
                   0412     * BUT THE AREA CODE. BIT 0 WILL ALWAYS BE ZERO.         PMN04070
                   0413     *                                                         PMN04080
                   0414     *                                                         PMN04090
                   0415     *** DISK1 AND DISKN WILL NOT WRITE BELOW THE            PMN04100
                   0416     *** FOLLOWING SCTR ADDRESSES (EXCEPT WRITE IMMED).      PMN04110
                   0417     *                                                         PMN04120
0095 0  0000       0418     $FPAD DC      *-*  FILE PROTECT ADDR, LOGICAL DR 0      PMN04130
0096 0  0000       0419           DC      *-*  FILE PROTECT ADDR, LOGICAL DR 1      PMN04140
0097 0  0000       0420           DC      *-*  FILE PROTECT ADDR, LOGICAL DR 2      PMN04150
0098 0  0000       0421           DC      *-*  FILE PROTECT ADDR, LOGICAL DR 3      PMN04160
0099 0  0000       0422           DC      *-*  FILE PROTECT ADDR, LOGICAL DR 4      PMN04170
                   0423     *                                                         PMN04180
                   0424     *** THE ARM POSITION IS UPDATED WHENEVER A SEEK         PMN04190
                   0425     *** OCCURS.                                             PMN04200
                   0426     *                                                         PMN04210
009A 0  0000       0427     $CYLN DC      0    ARM POSITION FOR LOGICAL DRIVE 0    PMN04220
009B 0  0000       0428           DC      0    ARM POSITION FOR LOGICAL DRIVE 1    PMN04230
009C 0  0000       0429           DC      0    ARM POSITION FOR LOGICAL DRIVE 2    PMN04240
009D 0  0000       0430           DC      0    ARM POSITION FOR LOGICAL DRIVE 3    PMN04250
009E 0  0000       0431           DC      0    ARM POSITION FOR LOGICAL DRIVE 4    PMN04260
                   0432     *                                                         PMN04270
                   0433     *** BELOW ARE THE DISK AREA CODES. A ZERO               PMN04280
                   0434     *** INDICATES THE CORRESPONDING DRIVE IS NOT            PMN04290
                   0435     *** ON THE SYSTEM                                       PMN04300
                   0436     *                                                         PMN04310
009F 0  0000       0437     $ACDE DC      *-*  AREA  CODE FOR LOGICAL DRIVE 0     PMN04320
00A0 0  0000       0438           DC      *-*  AREA  CODE FOR LOGICAL DRIVE 1     PMN04330
00A1 0  0000       0439           DC      *-*  AREA  CODE FOR LOGICAL DRIVE 2     PMN04340
00A2 0  0000       0440           DC      *-*  AREA  CODE FOR LOGICAL DRIVE 3     PMN04350
00A3 0  0000       0441           DC      *-*  AREA  CODE FOR LOGICAL DRIVE 4     PMN04360
                   0442     *                                                         PMN04370
                   0443     *** THE ADR OF THE CYLINDER IN WHICH A DEFECT OC-       PMN04380
                   0444     *** CURS, IF ANY, IS STORED IN THE 1ST, 2ND, OR 3RD     PMN04390
                   0445     *** WORD BELOW, DEPENDING ON WHETHER IT IS THE 1ST,     PMN04400
                   0446     *** 2ND, OR 3RD DEFECT ON THE CARTRIDGE.                PMN04410
                   0447     *                                                         PMN04420
00A4 0  0000       0448     $DCYL DC      *-*  DEFECTIVE CYLINDER ADDRESSES   1    PMN04430
00A5 0  0000       0449           DC      *-*  *FOR LOGICAL DRIVE 0            2    PMN04440
00A6 0  0000       0450           DC      *-*                                  3    PMN04450
00A7 0  0000       0451           DC      *-*  DEFECTIVE CYLINDER ADDRESSES   1    PMN04460
00A8 0  0000       0452           DC      *-*  *FOR LOGICAL DRIVE 1            2    PMN04470
00A9 0  0000       0453           DC      *-*                                  3    PMN04480
00AA 0  0000       0454           DC      *-*  DEFECTIVE CYLINDER ADDRESSES   1    PMN04490
00AB 0  0000       0455           DC      *-*  *FOR LOGICAL DRIVE 2            2    PMN04500
00AC 0  0000       0456           DC      *-*                                  3    PMN04510
00AD 0  0000       0457           DC      *-*  DEFECTIVE CYLINDER ADDRESSES   1    PMN04520
00AE 0  0000       0458           DC      *-*  *FOR LOGICAL DRIVE 3            2    PMN04530
00AF 0  0000       0459           DC      *-*                                  3    PMN04540
00B0 0  0000       0460           DC      *-*  DEFECTIVE CYLINDER ADDRESSES   1    PMN04550
00B1 0  0000       0461           DC      *-*  *FOR LOGICAL DRIVE 4            2    PMN04560
00B2 0  0000       0462           DC      *-*                                  3    PMN04570
                   0464     *                                                         PMN04590
                   0465     * ILS02--THIS SUBROUTINE SAVES XR1, XR2, STATUS,       PMN04600
                   0466     *         AND THE ACCUMULATOR AND ITS EXTENSION.        PMN04610
                   0467     *         THE ADDRESS OF THE INTERRUPT SERVICE ROU-     PMN04620
                   0468     *         TINE IS STORED IN $I205 BY PHASE 2 OF         PMN04630
                   0469     *         THE CORE IMAGE LOADER. WORD 10 ALWAYS         PMN04640
                   0470     *         CONTAINS THE ADDRESS OF $I200.                PMN04650
                   0471     *                                                         PMN04660
                   0472     *                                                         PMN04670
                   0473     *                                                         PMN04680
00B3 0  0000       0474     $I200 DC      *-*       ENTRY PT (LEVEL 2 INTRUPT)    PMN04690
00B4 0  6906       0475           STX     1 $I210+1   SAVE XR1                     PMN04700
00B5 0  6A07       0476           STX     2 $I210+3   SAVE XR2                     PMN04710
00B6 0  2807       0477           STS       $I210+4   STORE STATUS                PMN04720
00B7 0  D80A       0478           STD       $I290     SAVE ACCUMULATOR,EXTENSION    PMN04730
                   0479     * $I205+1 CONTAINS ADDR INTERRUPT ENTRY PT TO DKI/O    PMN04740
00B8 00 44000000   0480     $I205 BSI  L  *-*       BR TO SERVICE THE INTERRUPT   PMN04750
00BA 00 65000000   0481     $I210 LDX  L1 *-*       RESTORE XR1                   PMN04760
00BC 00 66000000   0482           LDX  L2 *-*       RESTORE XR2                   PMN04770
00BE 0  2000       0483           LDS     0         RESTORE STATUS                PMN04780
00BF 0  C802       0484           LDD       $I290   RESTORE ACCUMULATOR,EXT       PMN04790
00C0 00 4CC000B3   0485           BOSC I    $I200   RETURN FROM INTERRUPT         PMN04800
00C2    0000       0486     $I290 BSS  E  0                                       PMN04810
00C2 0  0000       0487           DC      *-*       CONTENTS OF ACCUMULATOR AND   PMN04820
00C3 0  0000       0488           DC      *-*       *EXTENTION                    PMN04830
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                  0490     *                                                        PMN04850
                  0491     * ILS04--THIS SUBROUTINE SAVES XR1, XR2, STATUS,        PMN04860
                  0492     *         AND THE ACCUMULATOR AND ITS EXTENSION.         PMN04870
                  0493     *         IF THE INTERRUPT IS FOR A KEYBOARD REQ-   *    PMN04880
                  0494     *         UEST,AND IF A MONITOR PROGRAM IS IN CON-  *    PMN04890
                  0495     *         TROL, CONTROL IS PASSED TO DUMP. OTHER-   *    PMN04900
                  0496     *         WISE, CONTROL IS PASSED TO THE KEYBOARD/  *    PMN04910
                  0497     *         CONSOLE PRINTER SUBROUTINE. WORD 12 AL-   *    PMN04920
                  0498     *         WAYS CONTAINS THE ADDRESS OF $I400.       *    PMN04930
                  0499     *                                                        PMN04940
                  0500     * THE TABLE BELOW CONTAINS THE ADDRESSES OF THE         PMN04950
                  0501     * INTERRUPT SERVICE ROUTINES FOR ALL THE DEVICES       PMN04960
                  0502     * ON LEVEL 4.                                           PMN04970
                  0503     *                                                        PMN04980
                  0504     *                                                        PMN04990
                  0505     *                                                        PMN05000
00C4 0  0000      0506     $I400 DC      *-*         ENTRY POINT                  PMN05010
00C5 0  0818      0507           STD     $I490       SAVE ACCUMULATOR, EXTENSION  PMN05020
00C6 0  280E      0508           STS     $I410       SAVE STATUS                  PMN05030
00C7 0  690F      0509           STX   1 $I410+2     SAVE XR1                     PMN05040
00C8 0  6A10      0510           STX   2 $I410+4     SAVE XR2                     PMN05050
00C9 0  0816      0511           XIO     $I492       SENSE DSW                    PMN05060
00CA 0  1002      0512           SLA     2           IS THIS INTERRUPT REQUEST    PMN05070
00CB 00 4C1000D0  0513           BSC   L $I403,-     BR IF NOT INTERRUPT REQUEST  PMN05080
00CD 00 4480002C  0514           BSI   I $IREQ       BR IF INTERRUPT REQUEST      PMN05090
00CF 0  FFFE      0515           DC      -2          ERROR CODE                   PMN05100
00D0 0  6109      0516     $I403 LDX   1 9           NO. DEVICES ON LEVEL TO XR1  PMN05110
00D1 0  0810      0517           XIO     $I494       SENSE ILSW                   PMN05120
00D2 0  1140      0518           SLCA  1             FIND CAUSE OF INTERRUPT      PMN05130
                  0519     * $I405+1 CONTAINS ADDR OF LEVEL 4 IBT MINUS 1        PMN05140
00D3 00 45800000  0520     $I405 BSI   I1 *-*        BR TO SERVICE THE INTERRUPT  PMN05150
00D5 0  2000      0521     $I410 LDS     0           RESTORE STATUS               PMN05160
00D6 00 65000000  0522           LDX   L1 *-*        RESTORE XR1                  PMN05170
00D8 00 66000000  0523           LDX   L2 *-*        RESTORE XR2                  PMN05180
00DA 0  C803      0524           LDD     $I490       RESTORE ACCUMULATOR, EXT.    PMN05190
00DB 00 4CC000C4  0525           BOSC I  $I400       RETURN                       PMN05200
                  0526     *                                                        PMN05210
                  0527     * CONSTANTS AND WORK AREAS                             PMN05220
                  0528     * EVEN-NUMBERED LABELS ARE ON EVEN BOUNDARIES          PMN05230
                  0529     *                                                        PMN05240
00DD 0  0000      0530     $DDSW DC      *-*         DSW FOR THE DISK             PMN05250
00DE    0002      0531     $I490 BSS   E 2       CONTENTS OF ACCUMULATOR, EXT.    PMN05260
00E0 0  0000      0532     $I492 DC      *-*                                      PMN05270
00E0 0            0533     $SYSC EQU     *-1   VERSION AND MOD NO.                PMN05280
00E1 0  0F00      0534           DC      /0F00 IOCC FOR SENSE IOCC FOR KB/CP      PMN05290
00E2    0001      0535     $I494 BSS     1     PATCH AREA                         PMN05300
00E3 0  0300      0536           DC      /0300 IOCC FOR SENSING ILSW04            PMN05310
                  0538     *                                                 2-2    PMN05330
                  0539     *     PATCH AREA                                  2-2    PMN05340
                  0540     *     FIX FOR APAR N5044                          2-2    PMN05350
                  0541     *                                                 2-2    PMN05360
00E4 0  0000      0542     $I496 DC      *-*         XR3 SETTING DURING XEQ 2-2   PMN05370
00E5 0  0F01      0543           DC      /0F01       SENSE KEY BOARD W RESET2-2   PMN05380
                  0544     *                                                 2-2    PMN05390
00E6 0  0000      0545     $I420 DC      *-*         ENTRY POINT FLUSH JOB  2-2   PMN05400
00E7 0  08FC      0546           XIO     $I496       SENSE KEY BOARD W RESET2-2   PMN05410
00E8 00 4C4000EA  0547           BOSC L  $I425       TURN OF INTERRUPT      2-2   PMN05420
00EA 00 4400003F  0548     $I425 BSI   L $DUMP       BRANCH TO WAIT OUT PEND2-2   PMN05430
00EC 0  FFFE      0549           DC      -2          *INTER AND GET AUX SUP 2-2   PMN05440
                  0550     *                                                 2-2    PMN05450
00ED    0001      0551           BSS     1           PATCH AREA             2-2   PMN05460
00EE 0  0000      0552     $DBSY DC      *-*   NON-ZERO WHEN DISK I/O BUSY          PMN05470

DISKZ

ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                  0554     ************************************************** PMN05490
                  0555     *                                                * PMN05500
                  0556     *STATUS-VERSION 2, MODIFICATION 1                * PMN05510
                  0557     *                                                * PMN05520
                  0558     *PROGRAM NAME-                                   * PMN05530
                  0559     *   *FULL NAME-FORTRAN/SYSTEM DISK I/O SUBROUTINE * PMN05540
                  0560     *   *CALLING SEQUENCE-                           * PMN05550
                  0561     *       LDD     PARAM                            * PMN05560
                  0562     *       BSI   L DZ000                            * PMN05570
                  0563     *    WHERE PARAM IS THE LABEL OF A DOUBLE-WORD    * PMN05580
                  0564     *    CELL CONTAINING THE FUNCTION CODE AND THE    * PMN05590
                  0565     *    ADDR OF THE I/O BUFFER,I.E., ADDR OF WD CNT. * PMN05600
                  0566     *    SEE 'CAPABILITIES' FOR DISCUSSION OF PARAM-  * PMN05610
                  0567     *    ETERS.                                       * PMN05620
                  0568     *                                                   PMN05630
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO
                  0569     *PURPOSE-                                               * PMN05640
                  0570     *   TO PROVIDE A SUBROUTINE TO PERFORM DISK OPERA-*     PMN05650
                  0571     *   TIONS. THIS SUBROUTINE IS INTENDED FOR USE BY *     PMN05660
                  0572     *   MONITOR PROGRAMS AND USER PROGRAMS WRITTEN IN *     PMN05670
                  0573     *   FORTRAN. THUS,IT IS INTENDED FOR USE IN AN    *     PMN05680
                  0574     *   ERROR-FREE ENVIRONMENT.                       *     PMN05690
                  0575     *                                                 *     PMN05700
                  0576     *METHOD-                                          *     PMN05710
                  0577     *   DISKZ REQUIRES A BUFFER,THE LENGTH OF WHICH IS*     PMN05720
                  0578     *   2 GREATER THAN THE NO. WORDS TO BE READ/WRIT- *     PMN05730
                  0579     *   TEN.                                          *     PMN05740
                  0580     *                                                 *     PMN05750
                  0581     *CAPABILITIES AND LIMITATIONS-                    *     PMN05760
                  0582     *   THE WD CNT,AS WELL AS DZ000,MUST BE ON AN EVEN*     PMN05770
                  0583     *   BOUNDARY,MUST BE IN THE RANGE 0-32767. THE    *     PMN05780
                  0584     *   DRIVE CODE MUST BE IN BITS 1-3 OF THE SECTOR  *     PMN05790
                  0585     *   ADDR,WHICH FOLLOWS THE WD CNT. THE FUNCTION   *     PMN05800
                  0586     *   INDICATOR MUST BE XX00 FOR A READ OR XX01 FOR *     PMN05810
                  0587     *   A WRITE,WHERE 'XX' MEANS ANY 2 HEXADECIMAL    *     PMN05820
                  0588     *   CHARACTERS. A WD CNT OF ZERO INDICATES A SEEK.*     PMN05830
                  0589     *   (READ OR WRITE MAY BE INDICATED.) AUTOMATIC   *     PMN05840
                  0590     *   SEEKING IS PROVIDED AS A PART OF READ/WRITE.  *     PMN05850
                  0591     *   A WRITE IS ALWAYS WITH A READ-BACK-CHECK.     *     PMN05860
                  0592     *   DISKZ MAKES NO PREOPERATIVE PARAMETER CHECKS. *     PMN05870
                  0593     *                                                 *     PMN05880
                  0594     *SPECIAL FEATURES-                                *     PMN05890
                  0595     *   DISKZ PROVIDES ONLY THOSE FUNCTIONS MENTIONED *     PMN05900
                  0596     *   ABOVE. DISK1 AND DISKN OFFER THIS BASIC SET OF*     PMN05910
                  0597     *   FUNCTIONS PLUS OTHERS.                        *     PMN05920
                  0598     *                                                 *     PMN05930
                  0599     ***************************************************     PMN05940
                  0601     * PROVIDE PARAMETERS FOR SYSTEM LOADER                  PMN05960
                  0602     *                                                       PMN05970
00F0      0000    0603           BSS  E  0                                         PMN05980
00F0 0    00EF    0604           DC      $ZEND-*     DISKZ WORD COUNT              PMN05990
00F1 0    FF6A    0605           DC      -'DZID      PHASE ID                      PMN06000
00F2 0    00E8    0606           DC      $ZEND-6-*+1 ADDR OF SLET EXTRACT          PMN06010
00F3 0    0001    0607           DC      1           NO. ENTRIES IN SLET EXTRACT   PMN06020
00F4              0608           ORG     *-2                                       PMN06030
00F2 0    0000    0610     DZ000 DC      *-*         ENTRY POINT                   PMN06050
00F3 00 740000EE  0611           MDX  L  $DBSY,0     LOOP UNTIL OPERATION IN       PMN06060
00F5 0    70FD    0612           MDX     *-3         *PROGRESS IS COMPLETE         PMN06070
00F6 0    7002    0613           MDX     DZ020       BR AROUND INT ENTRY POINT     PMN06080
                  0614     *                                                       PMN06090
                  0615     * INTERRUPT ENTRY POINT                                 PMN06100
                  0616     *                                                       PMN06110
00F7 0    0000    0617     DZ010 DC      *-*         INTERRUPT ADDRESS             PMN06120
00F8 0    7015    0618           MDX     DZ180       BR TO SERVICE INTERRUPT       PMN06130
00F9 0    690F    0619     DZ020 STX   1 DZ100+1     SAVE XR1                      PMN06140
00FA 0    6A10    0620           STX   2 DZ100+3     SAVE XR2                      PMN06150
00FB 0    1008    0621           SLA     8           SHIFT INDICATOR 8 BITS        PMN06160
00FC 0    D03C    0622           STO     DZ945       SAVE FUNCTION INDICATOR       PMN06170
00FD 0    18D0    0623           RTE     16                                        PMN06180
00FE 0    D056    0624           STO     DZ235+1     SAVE ADDR OF THE I/O AREA     PMN06190
00FF 0    6211    0625     DZ030 LDX   2 'TCNT       TURN BUSY INDICATOR ON AND    PMN06200
0100 0    6AED    0626           STX   2 $DBSY       *SET RETRY COUNT              PMN06210
0101 0    C0F0    0627           LD      DZ000                                     PMN06220
0102 0    D0F4    0628           STO     DZ010                                     PMN06230
0103 0    704E    0629           MDX     DZ230       BR TO CONTINUE                PMN06240
0104 00 4C000000  0630     DZ060 BSC  L  *-*         BR TO SERVICE THE INTERRUPT   PMN06250
                  0631     *                                                       PMN06260
                  0632     * START ALL DISK OPERATIONS                            PMN06270
                  0633     *                                                       PMN06280
0106 0    6908    0634     DZ070 STX   1 DZ180+1     SAVE ADDR OF THE I/O AREA     PMN06290
0107 0    081E    0635           XIO     DZ904       START AN OPERATION            PMN06300
                  0636     *                                                       PMN06310
                  0637     * RETURN TO USER                                       PMN06320
                  0638     *                                                       PMN06330
010B 00 65000000  0639     DZ100 LDX  L1 *-*         RESTORE XR1                   PMN06340
010A 00 66000000  0640           LDX  L2 *-*         RESTORE XR2                   PMN06350
010C 00 4C8000F7  0641           BSC  I  DZ010       RETURN                        PMN06360
                  0642     *                                                       PMN06370
                  0643     * SERVICE ALL INTERRUPTS                               PMN06380
                  0644     *                                                       PMN06390
010E 00 65000000  0645     DZ180 LDX  L1 *-*         ADDR OF I/O AREA TO XR1       PMN06400
0110 00 660000F2  0646           LDX  L2 DZ000       ADDR OF DZ000 TO XR2          PMN06410
0112 0    0819    0647           XIO     DZ910       SENSE THE DSW                 PMN06420
0113 0    D0C9    0648           STO     $DDSW       SAVE THE DSW                  PMN06430
0114 0    4850    0649           BOSC    -           SKIP IF ERROR BIT SET         PMN06440
0115 0    70FE    0650           MDX     DZ060       BRANCH IF ERROR BIT NOT SET   PMN06450
0116 0    C800    0651     DZ185 LDD     DZ902       RESTORE WORD COUNT            PMN06460
0117 0    D900    0652           STD   1 0           *AND SECTOR ADDRESS           PMN06470
```

```
ADDR REL OBJECT    ST.NO.   LABEL OPCD FT OPERANDS                                  ID/SEQNO
0118 00 74FF00EE   0653           MDX  L  $DBSY,-1  SKIP IF 16 RETRIES DONE       PMN06480
011A 0  7039       0654           MDX     DZ235     BRANCH IF LESS THAN 16        PMN06490
                   0655     *                                                      PMN06500
                   0656     * TRAP OUT TO POSTOPERATIVE TRAP                       PMN06510
                   0657     *                                                      PMN06520
011B 0  C812       0658           LDD     DZ912     1+SCTR ADDR TO EXTENSION      PMN06530
011C 0  C014       0659           LD      DZ915                                   PMN06540
011D 0  4293       0660     DZ190 BSI   2 $PST2-X2  BR TO POSTOPERATIVE ER TRAP   PMN06550
011E 0  1810       0661           SRA     16        CLEAR                         PMN06560
011F 00 048C0191   0662           STO  I  DZ350+1   *ARM POSITION                 PMN06570
0121 0  70DD       0663           MDX     DZ030     RETRY OPERATION               PMN06580
                   0664     *                                                      PMN06590
                   0665     * CONSTANTS AND WORK AREAS                            PMN06600
                   0666     *                                                      PMN06610
0122       0000    0667           BSS  E  0                                       PMN06620
                   0668     * EVEN-NUMBERED LABELS ARE ON EVEN BOUNDARIES         PMN06630
0122 0  0001       0669     DZ900 DC       1     CONSTANT,READ-AFTER-SEEK WD CNT PMN06640
0123 0  0000       0670     DZ901 DC       0     CURRENT ARM POSITION            PMN06650
0124 0  0000       0671     DZ902 DC       *-*   LAST TWO WORDS OF SECTOR        PMN06660
0125 0  0000       0672           DC       *-*   *PREVIOUSLY READ                PMN06670
0126 0  0000       0673     DZ904 DC       *-*   IOCC FOR OPERATION CURRENTLY    PMN06680
0127 0  0000       0674     DZ905 DC       *-*   *BEING PERFORMED                PMN06690
0128 0  0000       0675     DZ906 DC       *-*   SAVE AREA FOR IOCC FOR          PMN06700
0129 0  0000       0676     DZ907 DC       *-*   *USER-REQUESTED OPERATION       PMN06710
012A 0  0122       0677     DZ908 DC       DZ900 IOCC FOR READ                   PMN06720
012B 0  0000       0678     DZ909 DC       *-*   *AFTER SEEK                     PMN06730
012C 0  0000       0679     DZ910 DC       *-*   2ND WORD OF SEEK IOCC           PMN06740
012D 0  0000       0680     DZ911 DC       *-*   SENSE IOCC                      PMN06750
012E 0  0000       0681     DZ912 DC       *-*   INTERMEDIATE WORD COUNT         PMN06760
012F 0  0000       0682     DZ913 DC       *-*   ADDR OF NEXT SEQUENTIAL SECTOR  PMN06770
0130 0  5002       0683     DZ914 DC       /5002 WRITE SELECT/POWER UNSAFE INDR  PMN06780
0131 0  5004       0684     DZ915 DC       /5004 READ/WRITE/SEEK ERROR INDICATOR PMN06790
0132 0  FEC0       0685     DZ916 DC       -320 TO BE USED TO SIMULTANEOUSLY     PMN06800
0133 0  0001       0686           DC       1     *DECR WD CNT, INCR SCTR ADDR    PMN06810
0134 0  0080       0687     DZ920 DC       /0080 READ CHECK BIT FOR IOCC         PMN06820
0135 0  0600       0688     DZ925 DC       /0600 2ND WD OF READ IOCC W/O AREA CD PMN06830
0136 0  0008       0689     DZ930 DC       8     NO. SECTORS PER CYLINDER        PMN06840
0137 0  5000       0690     DZ935 DC       /5000 NOT READY DISPLAY CODE          PMN06850
0138 0  0FF8       0691     DZ940 DC       /0FF8 'AND' OUT DR CODE, SCTR ADDR    PMN06860
0139 0  0000       0692     DZ945 DC       *-*   FUNC INDICATOR (0#READ,1#WRITE) PMN06870
013A 0  0701       0693     DZ950 DC       /0701 SENSE IOCC W/O AREA CODE        PMN06880
013B 0  0007       0694     DZ955 DC       /0007 'AND' OUT ALL BUT SCTR NO.      PMN06890
013C 0  000A       0695     DZ960 DC       $DCYL-$CYLN BASE DEFECTIVE CYL ADDR   PMN06900
013D 0  009F       0696     DZ965 DC       $ACDE BASE AREA CODE ADDR             PMN06910
013E 0  FFF8       0697     DZ970 DC       $CYLN-$ACDE BASE ARM POSITION ADDR    PMN06920
013F 0  0000       0698     DZ975 DC       *-*   2ND WORD OF READ CHECK IOCC     PMN06930
0140 0  0400       0699     DZ980 DC       /0400 2ND WD OF SEEK IOCC W/O AREA CD PMN06940
0141 0  0141       0700     DZ985 DC       321   NO. WORDS PER SECTOR (W/ ADDR)  PMN06950
0142 0  0000       0701     DZ990 DC       *-*   CURRENT SECTOR NO.              PMN06960
0143 0  FFFF       0702     DZ995 DC       -1    MASK FOR COMPLEMENTING          PMN06970
                   0703     *                                                      PMN06980
                   0704     * RESERVED FOR SAVING CORE ON A DUMP ENTRY TO SKEL    PMN06990
                   0705     *                                                      PMN07000
0144       0002    0706           BSS     2     THIS AREA MUST BE AT $CIBA+319  PMN07010
00F2 0             0707     X2    EQU     DZ000                                   PMN07020
                   0708     *                                                      PMN07030
                   0709     *                                                      PMN07040
                   0710     *                                                      PMN07050
0146 0  1810       0711     DZ210 SRA     16                                      PMN07060
0147 0  D0A6       0712           STO     $DBSY     CLEAR BUSY INDICATOR          PMN07070
0148 00 74FF0032   0713           MDX  L  $IOCT,-1  DECREMENT IOCS COUNTER        PMN07080
014A 0  1000       0714           NOP                                             PMN07090
014B 0  70BC       0715           MDX     DZ100     TO EXIT                       PMN07100
                   0716     *                                                      PMN07110
                   0717     * PREPARE TO TRAP OUT ON 'POWER UNSAFE' CONDITION     PMN07120
                   0718     *                                                      PMN07130
014C 0  C0E3       0719     DZ215 LD      DZ914                                   PMN07140
014D 0  70CF       0720           MDX     DZ190     BR TO TPAP OUT                PMN07150
                   0721     *                                                      PMN07160
                   0722     * PREPARE TO TRAP OUT ON 'NOT READY' CONDITION        PMN07170
                   0723     *                                                      PMN07180
014E 0  C0E8       0724     DZ220 LD      DZ935     FETCH ERROR CODE              PMN07190
014F 00 44000028   0725           BSI  L  $PRET     BR TO PREOPERATIVE ERR TRAP   PMN07200
0151 0  7036       0726           MDX     DZ340     RETRY THE OPERATION           PMN07210
                   0727     *                                                      PMN07220
                   0728     *                       STATEMENTS MOVED        2-1   PMN07230
                   0729     *                                                      PMN07240
0152 00 74010032   0730     DZ230 MDX  L  $IOCT,1   INCREMENT IOCS COUNTER        PMN07250
0154 00 65000000   0731     DZ235 LDX  L1 *-*       ADDR I/O AREA TO XR1          PMN07260
0156 0  C900       0732           LDD   1 0                                       PMN07270
0157 0  D8CC       0733           STD     DZ902     SAVE WORD COUNT, SCTR ADDR    PMN07280
0158 0  D8D5       0734           STD     DZ912                                   PMN07290
```

```
ADDR REL OBJECT    ST.NO.  LABEL OPCD FT OPERANDS                                  ID/SEQNO
0159 0  1810       0735    DZ240 SRA     16                                        PMN07300
015A 0  1084       0736          SLT     4          DRIVE CODE IN BITS 12-15       PMN07310
015B 0  D00E       0737          STO     DZ280+1                                   PMN07320
015C 0  80E0       0738          A       DZ965      COMPUTE AND STORE THE          PMN07330
015D 0  D01C       0739          STO     DZ330+1    *ADDR OF THE AREA CODE         PMN07340
015E 0  800F       0740          A       DZ970      COMPUTE AND STORE THE          PMN07350
015F 0  D031       0741          STO     DZ350+1    *ADDR OF THE ARM POSITION      PMN07360
0160 0  8008       0742          A       DZ960      ADD IN BASE OF ADDR            PMN07370
0161 0  8008       0743          A       DZ280+1    ADD IN THE DRIVE               PMN07380
0162 0  8007       0744          A       DZ280+1    *CODE TWICE MORE               PMN07390
0163 0  D006       0745          STO     DZ280+1                                   PMN07400
0164 0  62FD       0746          LDX   2 -3         INITIALIZE COUNTER FOR LOOP    PMN07410
0165 0  69C2       0747          STX   1 DZ906                                     PMN07420
0166 0  C101       0748          LD    1 1          FETCH DESIRED SECTOR ADDR      PMN07430
0167 0  E0D0       0749          AND     DZ940      'AND' OUT SECTOR NO.           PMN07440
0168 0  D101       0750    DZ250 STO   1 1          *AND DRIVE CODE                PMN07450
0169 00 94000000   0751    DZ280 S     L *-*        SUB DEFECTIVE CYLINDER ADDR    PMN07460
016B 0  4828       0752          BSC     Z+         SKIP IF BAD CYLINDER           PMN07470
016C 0  7007       0753          MDX     DZ300      BR TO CONTINUE PROCESSING      PMN07480
016D 0  C101       0754          LD    1 1                                         PMN07490
016E 0  80C7       0755          A       DZ930      INCREMENT SCTR ADDR BY 8       PMN07500
016F 00 7401016A   0756          MDX   L DZ280+1,1  POINT TO NEXT DEFECTIVE CYL    PMN07510
0171 0  7201       0757          MDX   2 1          SKIP AFTER 3RD PASS            PMN07520
0172 0  70F5       0758          MDX     DZ250      COMPARE W/ NEXT DEF CYL ADR    PMN07530
0173 0  D101       0759          STO   1 1          SCTR ADDR WITH 3 DEF CYL2-4    PMN07535
                   0760    *                                                       PMN07540
                   0761    * CONSTRUCT THE 2ND WORD OF ALL IOCC'S                  PMN07550
                   0762    *                                                       PMN07560
0174 00 660000F2   0763    DZ300 LDX   L2 DZ000       ADDR OF DZ000 TO XR2        PMN07570
0176 0  C23D       0764          LD    2 DZ913-X2     FETCH SECTOR ADDRESS        PMN07580
0177 0  E249       0765          AND   2 DZ955-X2     'AND' OUT ALL BUT SECTOR NO PMN07590
0178 0  D250       0766          STO   2 DZ990-X2     SAVE SECTOR NO.             PMN07600
0179 00 C4000000   0767    DZ330 LD    L *-*          FETCH AREA CODE             PMN07610
017B 0  EA4F       0768          OR    2 DZ980-X2     'OR' IN SEEK FUNCTION CODE  PMN07620
017C 0  D23A       0769          STO   2 DZ910-X2     SEEK IOCC MINUS DIRECTION   PMN07630
017D 0  EA43       0770          OR    2 DZ925-X2     'OR' IN READ FUNCTION CODE  PMN07640
017E 0  D239       0771          STO   2 DZ909-X2     IOCC FOR READ-AFTER-SEEK    PMN07650
017F 0  EA50       0772          OR    2 DZ990-X2     'OR' IN SECTOR NO.          PMN07660
0180 0  9247       0773          S     2 DZ945-X2     COMPLETE READ/WRITE CODE    PMN07670
0181 0  D237       0774          STO   2 DZ907-X2     2ND WD OF READ/WRITE IOCC   PMN07680
0182 0  EA42       0775          OR    2 DZ920-X2     'OR' IN READ CHECK BIT      PMN07690
0183 0  8247       0776          A     2 DZ945-X2                                 PMN07700
0184 0  D24D       0777          STO   2 DZ975-X2     2ND WD OF READ CHECK IOCC   PMN07710
0185 0  EA48       0778          OR    2 DZ950-X2     'OR' IN SENSE IOCC BITS     PMN07720
0186 0  D23B       0779          STO   2 DZ911-X2     COMPLETED SENSE IOCC        PMN07730
0187 0  CA3C       0780          LDD   2 DZ912-X2     1+SCTR ADDR TO EXTENSION    PMN07740
0188 0  0A3A       0781    DZ340 XIO   2 DZ910-X2     SENSE FOR DISK READY        PMN07750
0189 0  D2EB       0782          STO   2 $DDSW-X2     SAVE THE DSW                PMN07760
018A 0  4828       0783          BSC     Z+         SKIP UNLESS POWER UNSAFE OR PMN07770
018B 0  70C0       0784          MDX     DZ215      *WRITE SELECT, BR OTHERWISE PMN07780
018C 0  1002       0785          SLA     2          BR TO PREOPERATIVE ERR TRAP PMN07790
018D 0  4828       0786          BSC     Z+         *IF DISK NOT READY, SKIP    PMN07800
018E 0  70BF       0787          MDX     DZ220      *OTHERWISE                  PMN07810
                   0788    *                        STATEMENTS REMOVED     2-1 PMN07820
018F 0  C101       0789          LD    1 1          FETCH DESIRED CYLINDER ADDR PMN07830
0190 00 94000000   0790    DZ350 S     L *-*        SUBTRACT ARM POSITION       PMN07840
0192 0  4818       0791          BSC     +-         SKIP IF SEEK NECESSARY      PMN07850
0193 0  7018       0792          MDX     DZ400      BRANCH TO PERFORM OPERATION PMN07860
                   0793    *                                                     PMN07870
                   0794    * SEEK                                                PMN07880
                   0795    *                                                     PMN07890
0194 0  1893       0796          SRT     19         PUT NO. CYLINDERS IN EXT    PMN07900
0195 0  180F       0797          SRA     15         + OR - SIGN TO BIT 15       PMN07910
0196 0  1002       0798          SLA     2          SHIFT SIGN TO BIT 13        PMN07920
0197 0  EA3A       0799          OR    2 DZ910-X2   "OR" IN REMAINDER OF IOCC   PMN07930
0198 0  18D0       0800          RTE     16                                     PMN07940
0199 0  4810       0801          BSC     -          SKIP IF SEEK TOWARD HOME    PMN07950
019A 0  7002       0802          MDX     DZ380      BRANCH IF SEEK TOWARD CENTR PMN07960
019B 0  F251       0803          EOR   2 DZ995-X2   COMPLEMENT NO. CYLS TO BE   PMN07970
019C 0  8230       0804          A     2 DZ900-X2   *SOUGHT TO GET POSITIVE NO. PMN07980
019D 0  DA34       0805    DZ380 STD   2 DZ904-X2                               PMN07990
019E 0  C2E8       0806          LD    2 $DDSW-X2   FETCH THE DSW           2-1 PMN08000
019F 0  100D       0807          SLA     13                                 2-1 PMN08010
01A0 0  4810       0808          BSC     -                                  2-1 PMN08020
01A1 0  7003       0809          MDX     DZ390                              2-1 PMN08030
01A2 0  C101       0810    DZ385 LD    1 1          FETCH SECTOR ADDR       2-1 PMN08040
01A3 0  1803       0811          SRA     3          CONVERT TO CYLINDER ADDR2-1 PMN08050
01A4 0  D234       0812          STO   2 DZ904-X2 *AND STORE IN IOCC          2-1 PMN08060
01A5 0  4213       0813    DZ390 BSI   2 DZ070-1-X2 START SEEK                2-1 PMN08070
```

```
ADDR REL OBJECT   ST.NO.  LABEL OPCD FT OPERANDS                                  ID/SEQNO
                  0814    *                                                       PMN08080
                  0815    * SEEK COMPLETE INTERRUPT PROCESSING                   PMN08090
                  0816    *                                                       PMN08100
01A6 0  CA38      0817          LDD   2 DZ908-X2  SET UP IOCC FOR               PMN08110
01A7 0  DA34      0818          STD   2 DZ904-X2  *READ AFTER SEEK               PMN08120
01A8 0  4213      0819          BSI   2 DZ070-1-X2 START READ-AFTER-SEEK         PMN08130
                  0820    *                                                       PMN08140
                  0821    * READ-AFTER-SEEK COMPLETE INTERRUPT PROCESSING        PMN08150
                  0822    *                                                       PMN08160
01A9 0  C231      0823          LD    2 DZ901-X2  FETCH ADR OF SCTR JUST READ   PMN08170
01AA 00 04800191  0824          STO   I DZ350+1   UPDATE ARM POSITION           PMN08180
01AC 0  9101      0825          S     1 1         SUB DESIRED SCTR ADDR         PMN08190
01AD 00 4C200116  0826          BSC   L DZ185,Z   BR IF SEEK UNSUCCESSFUL       PMN08200
                  0827    *                                                       PMN08210
                  0828    *                                                       PMN08220
                  0829    * READ/WRITE                                           PMN08230
                  0830    *                                                       PMN08240
01AF 0  CA3C      0831    DZ400 LDD   2 DZ912-X2  FETCH INTERMEDIATE WD CNT       PMN08250
01B0 0  4808      0832          BSC     +         SKIP, WD CNT NOT EXHAUSTED    PMN08260
01B1 0  7094      0833    DZ410 MDX     DZ210     BRANCH IF READ/WRITE DONE     PMN08270
01B2 0  8A40      0834          AD    2 DZ916-X2  DECREMENT WORD COUNT AND       PMN08280
01B3 0  DA3C      0835          STD   2 DZ912-X2  *INCREMENT SECTOR ADDRESS      PMN08290
01B4 0  4830      0836          BSC     Z-        SKIP IF THIS IS LAST SECTOR   PMN08300
01B5 0  1810      0837          SRA     16        CLEAR ACCUMULATOR             PMN08310
01B6 0  824F      0838          A     2 DZ985-X2  ADD BACK 321 TO WD CNT         PMN08320
01B7 0  D100      0839          STO   1 0         STORE RESULT IN I/O AREA      PMN08330
01B8 0  CA36      0840          LDD   2 DZ906-X2  RESTORE IOCC FOR ORIGINALLY    PMN08340
01B9 0  DA34      0841          STD   2 DZ904-X2  *REQUESTED OPERATION          PMN08350
01BA 0  C101      0842          LD    1 1         ADD SECTOR NO. TO SECTOR      PMN08360
01BB 0  EA50      0843          OR    2 DZ990-X2  *ADDRESS                      PMN08370
01BC 0  D101      0844          STO   1 1                                       PMN08380
01BD 0  4213      0845          BSI   2 DZ070-1-X2 START READ/WRITE OPERATION   PMN08390
                  0846    *                                                       PMN08400
                  0847    * READ/WRITE COMPLETE INTERRUPT PROCESSING             PMN08410
                  0848    *                                                       PMN08420
01BE 0  C24D      0849          LD    2 DZ975-X2  SET UP FOR READ CHECK         PMN08430
01BF 0  0235      0850          STO   2 DZ905-X2                                PMN08440
01C0 0  C247      0851          LD    2 DZ945-X2  FETCH FUNCTION INDICATOR      PMN08450
01C1 0  4820      0852          BSC     Z         SKIP IF READ REQUESTED        PMN08460
01C2 0  4213      0853          BSI   2 DZ070-1-X2 START READ CHECK OPERATION   PMN08470
01C3 0  CA32      0854          LDD   2 DZ902-X2  RESTORE LAST 2 WDS OF SEC-    PMN08480
01C4 0  0900      0855          STD   1 0         *TOR PREVIOUSLY READ          PMN08490
01C5 0  C23C      0856          LD    2 DZ912-X2  FETCH INTERMEDIATE WD CNT     PMN08500
01C6 0  4808      0857          BSC     +         SKIP IF MORE READING/WRTING   PMN08510
01C7 0  70E9      0858          MDX     DZ410     BRANCH IF FINISHED            PMN08520
01C8 00 75000140  0859          MDX   L1 320      POINT XR1 TO NEW I/O AREA     PMN08530
01CA 0  C900      0860          LDD   1 0         SAVE LAST 2 WDS OF SECTOR     PMN08540
01CB 0  DA32      0861          STD   2 DZ902-X2  *JUST READ/WRITTEN            PMN08550
01CC 0  CA3C      0862          LDD   2 DZ912-X2  WD CNT, SCTR ADDR NEXT OP     PMN08560
01CD 0  0900      0863          STD   1 0         STORE BOTH IN NEW I/O AREA    PMN08570
01CF 0  708A      0864          MDX     DZ240     BACK TO SET UP NEXT OPERATN   PMN08580
                  0865    *                                                       PMN08590
                  0866    *                                                       PMN08600
01CF      0008    0867          BSS     11        PATCH AREA              2-4   PMN08610
                  0868    *                                                       PMN08620
                  0869    *                                                       PMN08630
01DA 0  00A0      0870          DC      *CIL1     ID NO. OF CORE IMAGE LDR,P1   PMN08640
01DB 0  0000      0871    $CION DC      *-*       CORE ADDR/CID NO.             PMN08650
01DC 0  0000      0872          DC      *-*       WORD COUNT                    PMN08660
01DD 0  0000      0873          DC      *-*       SCTR ADDR                     PMN08670
01DE      0002    0874          BSS     2         WD CNT, SCTR ADDR CORE LDS    PMN08680
01E0 0            0875    $ZEND EQU     *         1 + END OF DISKZ              PMN08690
```

EQUIVALENCES

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                  ID/SEQNO

                  0877     *                                                      PMN08710
                  0878     * EQUIVALENCES FOR DCOM PARAMETERS                     PMN08720
                  0879     *                                                      PMN08730
0004 0            0880     #NAME EQU     4     NAME OF PROGRAM/CORE LOAD          PMN08740
0006 0            0881     #DBCT EQU     6     BLOCK CT OF PROGRAM/CORE LOAD      PMN08750
0007 0            0882     #FCNT EQU     7     FILES SWITCH                       PMN08760
0008 0            0883     #SYSC EQU     8     SYSTEM/NON-SYSTEM CARTRIDGE INDR   PMN08770
0009 0            0884     #JBSW EQU     9     JOBT SWITCH                        PMN08780
000A 0            0885     #CBSW EQU     10    CLB-RETURN SWITCH                  PMN08790
000B 0            0886     #LCNT EQU     11    NO. OF LOCALS                      PMN08800
000C 0            0887     #MPSW EQU     12    CORE MAP SWITCH                    PMN08810
000D 0            0888     #MDF1 EQU     13    NO. DUP CTRL RECORDS (MODIF)       PMN08820
000E 0            0889     #MDF2 EQU     14    ADDR OF MODIF BUFFER               PMN08830
000F 0            0890     #NCNT EQU     15    NO. OF NOCALS                      PMN08840
0010 0            0891     #ENTY EQU     16    RLTV ENTRY ADDR OF PROGRAM         PMN08850
0011 0            0892     #RP67 EQU     17    1442-5 SWITCH                      PMN08860
0012 0            0893     #TODR EQU     18    OBJECT WORK STORAGE DRIVE CODE     PMN08870
0014 0            0894     #FHOL EQU     20    ADDR LARGEST HOLE IN FIXED AREA    PMN08880
0015 0            0895     #FSZE EQU     21    BLK CNT LARGEST HOLE IN FXA        PMN08890
0016 0            0896     #UHOL EQU     22    ADDR LARGEST HOLE IN USER AREA     PMN08900
0017 0            0897     #USZE EQU     23    BLK CNT LARGEST HOLE IN UA.        PMN08910
0018 0            0898     #DCSW EQU     24    DUP CALL SWITCH                    PMN08920
0019 0            0899     #PIOD EQU     25    PRINCIPAL I/O DEVICE INDICATOR     PMN08930
001A 0            0900     #PPTR EQU     26    PRINCIPAL PRINT DEVICE INDICATOR   PMN08940
001B 0            0901     #CIAD EQU     27    RLTV ADDR IN 'STRT OF CIL ADDR     PMN08950
001C 0            0902     #ACIN EQU     28    AVAILABLE CARTRIDGE INDICATOR      PMN08960
001D 0            0903     #GRPH EQU     29    2250 INDICATOR              2G2    PMN08970
001E 0            0904     #GCNT EQU     30    NO. G2250 RECORDS           2G2    PMN08980
001F 0            0905     #LOSW EQU     31    LOCAL-CALLS-LOCAL SWITCH    2-2    PMN08990
0020 0            0906     #X3SW EQU     32    SPECIAL ILS SWITCH          2-2    PMN09000
0021 0            0907     #ECNT EQU     33    NO. OF *EQUAT RCDS          2-4    PMN09005
0023 0            0908     #ANDU EQU     35    1+BLK ADDR END OF UA (ADJUSTED)    PMN09010
0028 0            0909     #BNDU EQU     40    1+BLK ADDR END OF UA (BASE)        PMN09020
002D 0            0910     #FPAD EQU     45    FILE PROTECT ADDR                  PMN09030
0032 0            0911     #PCID EQU     50    CARTRIDGE ID, PHYSICAL DRIVE       PMN09040
0037 0            0912     #CIDN EQU     55    CARTRIDGE ID, LOGICAL DRIVE        PMN09050
003C 0            0913     #CIBA EQU     60    SCTR ADDR OF CIB                   PMN09060
0041 0            0914     #SCRA EQU     65    SCTR ADDR OF SCRA                  PMN09070
0046 0            0915     #FMAT EQU     70    FORMAT OF PROG IN WORKING STG      PMN09080
004B 0            0916     #FLET EQU     75    SCTR ADDR 1ST SCTR OF FLET         PMN09090
0050 0            0917     #ULET EQU     80    SCTR ADDR 1ST SCTR OF LET          PMN09100
0055 0            0918     #WSCT EQU     85    BLK CNT OF PROG IN WORKING STG     PMN09110
005A 0            0919     #CSHN EQU     90    NO. SCTRS IN CUSHION AREA          PMN09120
                  0920     *                                                      PMN09130
                  0921     * EQUIVALENCES FOR PHASE ID NUMBERS                    PMN09140
                  0922     *                                                      PMN09150
006E 0            0923     'MCRA EQU     110       PHASE ID FOR MCRA              PMN09160
0073 0            0924     'SUP6 EQU     115       PHASE ID FOR DUMP PROG  2-4    PMN09170
0074 0            0925     'SUP7 EQU     116       PHASE ID FOR AUX SUPV   2-4    PMN09180
0078 0            0926     'CLB0 EQU     120       PHASE ID FOR CLB, PHASE 0/1    PMN09190
008C 0            0927     '1403 EQU     140       PHASE ID FOR SYS 1403 SUBR     PMN09200
008D 0            0928     '1132 EQU     141       PHASE ID FOR SYS 1132 SUBR     PMN09210
008E 0            0929     'CPTR EQU     142       PHASE ID FOR SYS CP SUBR       PMN09220
008F 0            0930     '2501 EQU     143       PHASE ID FOR SYS 2501 SUBR     PMN09230
0090 0            0931     '1442 EQU     144       PHASE ID FOR SYS 1442 SUBR     PMN09240
0091 0            0932     '1134 EQU     145       PHASE ID FOR SYS 1134 SUBR     PMN09250
0092 0            0933     'KBCP EQU     146       PHASE ID FOR SYS KB/CP SUBR    PMN09260
0093 0            0934     'CDCV EQU     147       PHASE ID FOR SYS CD   CONV     PMN09270
0094 0            0935     'PTCV EQU     148       PHASE ID FOR SYS 1134 CONV     PMN09280
0095 0            0936     'KBCV EQU     149       PHASE ID FOR SYS KB   CONV     PMN09290
0096 0            0937     'DZID EQU     150       PHASE ID FOR DISKZ             PMN09300
0097 0            0938     'DIID EQU     151       PHASE ID FOR DISKI             PMN09310
0098 0            0939     'DNID EQU     152       PHASE ID FOR DISKN             PMN09320
00A0 0            0940     'CIL1 EQU     160       PHASE ID FOR CI LOADER,PH 1    PMN09330
00A1 0            0941     'CIL2 EQU     161       PHASE ID FOR CI LOADER,PH 2    PMN09340
                  0942     *                                                      PMN09350
                  0943     * EQUIVALENCES FOR RESIDENT MONITOR                    PMN09360
                  0944     *                                                      PMN09370
0014 0            0945     $LKNM EQU     $HASH     SAVE AREA FOR NAME OF LINK     PMN09380
0016 0            0946     $RMSW EQU     $HASH+2   EXIT-LINK-DUMP SW(-1,0,+1)     PMN09390
0017 0            0947     $CXR1 EQU     $HASH+3   SAVE AREA FOR XR1              PMN09400
0018 0            0948     $CLSW EQU     $HASH+4   SW FOR CORE IMAGE LDR,PH 2     PMN09410
0019 0            0949     $DMPF EQU     $HASH+5   DUMP FORMAT CODE               PMN09420
001A 0            0950     $ACEX EQU     $HASH+6   ACC AND EXT WHEN ENTER DUMP    PMN09430
005A 0            0951     $CILA EQU     $S150+1   ADDR OF END OF DK I/O - 3      PMN09440
00B9 0            0952     $IBT2 EQU     $I205+1   ADR OF SERVICE PART OF DKIO    PMN09450
00D4 0            0953     $IBT4 EQU     $I405+1   ADDR OF THE IBT                PMN09460
00FF 0            0954     $SNLT EQU     $DBSY+1   SENSE LIGHT INDICATOR          PMN09470
00F0 0            0955     $PAUS EQU     DZ000-2   PAUSE,INTERRUPT INDICATOR     PMN09480
00F1 0            0956     $RWCZ EQU     DZ000-1   READ/WRITE SWITCH (CARDZ)     PMN09490
00E4 0            0957     $XR3X EQU     $I496     XR3 SETTING DURING XEQ  2-2    PMN09500
                  0958     *                                                      PMN09510
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                  0959     * EQUIVALENCES FOR ABSOLUTE SECTOR ADDRESSES             PMN09520
                  0960     *                                                        PMN09530
0000 0            0961     'IDAD EQU     0    ADDR OF SCTR WITH ID,DEF CYL ADR      PMN09540
0001 0            0962     'DCOM EQU     1    ADDR OF SCTR CONTAINING DCOM          PMN09550
0002 0            0963     'RIAD EQU     2    ADDROOF SCTR CONTAINING RES IMGE      PMN09560
0003 0            0964     'SLET EQU     3    ADDR OF SCTR CONTAINING SLET          PMN09570
0006 0            0965     'RTBL EQU     6    ADDR OF SCTR CONTAINING RELD TBL      PMN09580
0007 0            0966     'HDNG EQU     7    ADDR OF SCTR CONTAINING PAGE HDR      PMN09590
0000 0            0967     'STRT EQU     0    ADDR OF SCTR W/ COLD START PROG       PMN09600
                  0968     *                                                        PMN09610
                  0969     * EQUIVALENCES FOR THE CORE IMAGE HEADER                 PMN09620
                  0970     *                                                        PMN09630
0000 0            0971     'XEQA EQU     0    RLTV ADDR OF CORE LOAD EXEC ADDR      PMN09640
0001 0            0972     'CMON EQU     1    RLTV ADDR OF WD CNT OF COMMON         PMN09650
0002 0            0973     'DREQ EQU     2    RLTV ADDR OF DISK I/O INDICATOR       PMN09660
0003 0            0974     'FILE EQU     3    RLTV ADDR OF NO. FILES DEFINED        PMN09670
0004 0            0975     'HWCT EQU     4    RLTV ADDR OF WD CNT OF CI HEADER      PMN09680
0005 0            0976     'LSCT EQU     5    SCTR CNT OF FILES IN WK STORAGE       PMN09690
0006 0            0977     'LOAD EQU     6    RLTV ADDR OF LOAD ADDR CORE LOAD      PMN09700
0007 0            0978     'XCTL EQU     7    RLTV ADDR DISK1/DISKN EXIT CTRL       PMN09710
0008 0            0979     'TVWC EQU     8    RLTV ADDR OF WD CNT OF TV             PMN09720
0009 0            0980     'WCNT EQU     9    RLTV ADDR OF WD CNT OF CORE LOAD      PMN09730
000A 0            0981     'XR3X EQU     10   RLTV ADDR OF EXEC SETTING OF XR3      PMN09740
000B 0            0982     'ITVX EQU     11   RLTV ADDR OF 1ST WD OF ITV            PMN09750
0011 0            0983     'ILS4 EQU     17   RLTV ADDR OF 1ST WD OF IBT4           PMN09760
001A 0            0984     'OVSW EQU     26   RLTV ADDR OF LOCAL/SOCAL SWITCH       PMN09770
001C 0            0985     'CORE EQU     28   CORE SIZE OF BUILDING SYSTEM          PMN09780
001D 0            0986     'HEND EQU     29   RLTV ADDR OF LAST WD OF CI HDR        PMN09790
                  0987     *                                                        PMN09800
                  0988     * EQUIVALENCES FOR LET/FLET                             PMN09810
                  0989     *                                                        PMN09820
0005 0            0990     'LFHD EQU     5    WORD COUNT OF LET/FLET HEADER         PMN09830
0003 0            0991     'LFEN EQU     3    NO OF WDS PER LET/FLET ENTRY          PMN09840
0000 0            0992     'SCTN EQU     0    RLTV ADDR OF LET/FLET SCTR NO.        PMN09850
0001 0            0993     'UAFX EQU     1    RLTV ADDR OF SCTR ADDR OF UA/FXA      PMN09860
0003 0            0994     'WDSA EQU     3    RLTV ADDR OF WDS AVAIL IN SCTR        PMN09870
0004 0            0995     'NEXT EQU     4    RLTV ADDR OF ADDR NEXT SCTR           PMN09880
0000 0            0996     'LFNM EQU     0    RLTV ADDR OF LET/FLET ENTRY NAME      PMN09890
0002 0            0997     'BLCT EQU     2    RLTV ADDR OF LET/FLET ENTRY DBCT      PMN09900
                  0998     *                                                        PMN09910
                  0999     * MISCELLANEOUS EQUIVALENCES                            PMN09920
                  1000     *                                                        PMN09930
0033 0            1001     'ISTV EQU     51   ISS NO. ADJUSTMENT FACTOR    2-1      PMN09940
0005 0            1002     'MXDR EQU     5    MAX NO. DRIVES SUPPORTED              PMN09950
0380 0            1003     'COMZ EQU     896  LOW COMMON LIMIT FOR DISKZ            PMN09960
04C0 0            1004     'COM1 EQU     1216 LOW COMMON LIMIT FOR DISK1            PMN09970
0600 0            1005     'COM2 EQU     1536 LOW COMMON LIMIT OF DISKN             PMN09980
0011 0            1006     'TCNT EQU     17   NO. TRIES BEFORE DISK ERROR           PMN09990
00F9 0            1007     'DKEP EQU     DZ000+7 LIBF ENTRY TO DISK1/N              PMN10000
00F7 0            1008     'DKIP EQU     DZ000+5 DISK I/O INTERRUPT ENTRY PT        PMN10010
0010 0            1009     'SCIB EQU     16   CIB SECTOR COUNT             2-2      PMN10020
0003 0            1010     'HCIB EQU     3    HIGH COMMON SECTOR COUNT     2-2      PMN10030
1000 0            1011     'MCOR EQU     4096       SIZE OF MINIMUM CORE   2-2      PMN10040
007F 0            1012     Y     EQU     127                                       PMN10050
                  1013     *                                                        PMN10060
0004 0            1014     'CIDN EQU     4    RLTV ADDR CARTRIDGE ID       2-2      PMN10070
0005 0            1015     'COPY EQU     5    RLTV ADDR COPY INDICATOR     2-2      PMN10080
0001 0            1016     'DCTB EQU     1    RLTV ADDR DEFECTIV CYL TBL   2-2      PMN10090
0008 0            1017     'DTYP EQU     8    RLTV ADDR DISK TYPE INDR     2-2      PMN10100
```

COLD START PROGRAM

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                    ID/SEQNO

                   1019    ************************************************** PMN10120
                   1020    *                                                * PMN10130
                   1021    *STATUS - VERSION 2, MODIFICATION LEVEL 5.       * PMN10140
                   1022    *                                                * PMN10150
                   1023    *FUNCTION/OPERATION -                            * PMN10160
                   1024    *   THIS PROGRAM IS READ INTO CORE FROM SECTOR 0  * PMN10170
                   1025    *   OF THE SYSTEM CARTRIDGE AND TRANSFERRED TO BY * PMN10180
                   1026    *   THE COLD START CARD.  DEFECTIVE CYLINDER     * PMN10190
                   1027    *   ADDRESSES, CARTRIDGE ID AND DISKZ ARE ALSO ON * PMN10200
                   1028    *   SECTOR 0 AND ARE READ IN AT THE SAME TIME.   * PMN10210
                   1029    *   ALL THAT REMAINS FOR THE COLD START PROGRAM IS* PMN10220
                   1030    *   TO READ IN THE RESIDENT IMAGE, SAVE THE      * PMN10230
                   1031    *   CARTRIDGE ID AND TRANSFER TO THE AUXILIARY   * PMN10240
                   1032    *   SUPERVISOR THROUGH $DUMP IN THE RESIDENT     * PMN10250
                   1033    *   MONITOR.                                     * PMN10260
                   1034    *                                                * PMN10270
                   1035    *ENTRY - CR010-2                                 * PMN10280
                   1036    *   ENTER PROGRAM BY TRANSFER FROM COLD START CARD* PMN10290
                   1037    *                                                * PMN10300
                   1038    *INPUT -                                         * PMN10310
                   1039    *   THE CARTRIDGE ID OF LOGICAL DRIVE ZERO (THE  * PMN10320
                   1040    *   SYSTEM CARTRIDGE) IS READ IN FROM SECTOR 0   * PMN10330
                   1041    *   WITH THE COLD START PROGRAM.                 * PMN10340
                   1042    *                                                * PMN10350
                   1043    *OUTPUT -                                        * PMN10360
                   1044    *   * THE RESIDENT IMAGE IS READ INTO CORE FROM  * PMN10370
                   1045    *     THE DISK.                                  * PMN10380
                   1046    *   * IN COMMA-                                  * PMN10390
                   1047    *     $ACDE                                      * PMN10400
                   1048    *     $CIBA-1                                    * PMN10410
                   1049    *     $CIDN                                      * PMN10420
                   1050    *     $CYLN                                      * PMN10430
                   1051    *     $DBSY                                      * PMN10440
                   1052    *     $IOCT                                      * PMN10450
                   1053    *                                                * PMN10460
                   1054    *EXTERNAL REFERENCES -                           * PMN10470
                   1055    *   DZ000  SUBROUTINE TO PERFORM DISK I/O.       * PMN10480
                   1056    *                                                * PMN10490
                   1057    *EXITS -                                         * PMN10500
                   1058    *   THE ONLY EXIT IS TO THE AUXILIARY SUPERVISOR * PMN10510
                   1059    *   AS FOLLOWS-                                  * PMN10520
                   1060    *     BSI     $DUMP                              * PMN10530
                   1061    *     DC      -1                                 * PMN10540
                   1062    *                                                * PMN10550
                   1063    *TABLES/WORK AREAS - N/A                         * PMN10560
                   1064    *                                                * PMN10570
                   1065    *ATTRIBUTES -                                    * PMN10580
                   1066    *   THIS PROGRAM IS NOT NATURALLY RELOCATABLE.   * PMN10590
                   1067    *                                                * PMN10600
                   1068    *NOTES -                                         * PMN10610
                   1069    *   DISK ERRORS RESULT IN A WAIT AT $PST2.       * PMN10620
                   1070    ************************************************** PMN10630
                   1072    *                                                  PMN10650
                   1073    * READ THE RESIDENT IMAGE INTO CORE                PMN10660
                   1074    *                                                  PMN10670
01E0 0  617F       1075          LDX   1 Y                                    PMN10680
01E1 0  C82E       1076          LDD     CR920      SET UP WORD COUNT AND SCTR  PMN10690
01E2 00 DC000004   1077    CR010 STD  L  $CIBA-1    *ADDR OF RESIDENT IMAGE     PMN10700
01E4 0  D125       1078          STO   1 $DCYL-Y    *INITIALIZE DEF CYL NO. 1   PMN10710
01E5 0  C184       1079          LD    1 3-Y        FETCH LOG DRIVE 0 AREA CODE PMN10720
01E6 0  D120       1080          STO   1 $ACDE-Y    *AND STORE IT IN COMMA      PMN10730
01E7 0  D029       1081          STO     CR920+1    SAVE THE AREA CODE          PMN10740
01E8 0  C156       1082          LD    1 DZ000-2-27-Y FETCH AND SAVE THE        PMN10750
01E9 0  D0F1       1083          STO     $CIDN      *CARTRIDGE ID               PMN10760
01EA 0  C0F8       1084          LD      CR010+1    FETCH CORE ADDR OF RESIDENT PMN10770
01EB 0  1890       1085          SRT     16         *IMAGE AND PUT IN EXTENSION PMN10780
01EC 0  D16F       1086          STO   1 $DBSY-Y    CLEAR DISK BUSY INDICATOR   PMN10790
01ED 0  D118       1087          STO   1 $CYLN-Y    INITIALIZE ARM POSITION     PMN10800
01EE 0  4173       1088          BSI   1 DZ000-Y    FETCH RESIDENT IMAGE        PMN10810
01EF 0  3000       1089          WAIT               WAIT OUT THE INTERRUPT      PMN10820
                   1090    *                                                  PMN10830
                   1091    * INITIALIZE ITEMS IN COMMA                       PMN10840
                   1092    *                                                  PMN10850
01F0 0  1810       1093          SRA     16                                   PMN10860
01F1 0  D183       1094          STO   1 $IOCT-Y    CLEAR IOCS COUNTER          PMN10870
01F2 0  C818       1095          LDD     CR910                                PMN10880
01F3 0  D985       1096          STD   1 $CIBA-1-Y *FOR SAVING CORE ON THE CIB PMN10890
01F4 0  C01C       1097          LD      CR920+1    FETCH AREA CODE             PMN10900
01F5 0  D120       1098          STO   1 $ACDE-Y    RESET AREA CODE             PMN10910
01F6 0  C016       1099          LD      CR905      INITIALIZE WD ZERO TO BR TO PMN10920
01F7 0  D181       1100          STO   1 0-Y        *DUMP ENTRY POINT PLUS 1    PMN10930
                   1101    *                                                  PMN10940
                   1102    * TRANSFER TO THE AUXILIARY SUPERVISOR            PMN10950
```

COLD START PROGRAM

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD ET OPERANDS                          ID/SEQND

020E 0  0001       1113    CR910 DC      1          WD CNT,SCTR ADDR OF CAUSE  PMN11060
020F 0  0007       1114          DC      'HONG      *HARMLESS WRITE TO DISK    PMN11070
0210 0  00E8       1115    CR920 DC      $DBSY-$CH12 WD CNT AND SCTR           PMN11080
0211 0  0002       1116          DC      'RIAD      *ADDR OF RESIDENT IMAGE    PMN11090
0212    0212       1117          END     *                                     PMN11100
```

```
ADDR REL OBJECT   ST.NO.   LABEL OPCD FT OPERANDS                                  ID/SEQNO

                  1103    * TO COMPLETE INITIALIZATION                           PMN10960
                  1104    *                                                      PMN10970
01F8 0  41C0      1105          BSI    1 $DUMP-Y    BR TO AUXILLIARY SUPERVISOR PMN10980
01F9 0  FFFF      1106          DC      -1          *FOR JOB PROCESSING         PMN10990
                  1107    *                                                      PMN11000
01FA    0013      1108          BSS     19          PATCH AREA                  PMN11010
                  1109    *                                                      PMN11020
                  1110    * CONSTANTS AND WORK AREAS                            PMN11030
                  1111    *                                                      PMN11040
020D 0  703F      1112    CR905 MDX  X  $DUMP+1-1 TO BE STORED IN LOCN ZERO      PMN11050
020E 0  0000      1113    CR910 DC      0           WC CNT,SCTR ADDR OF     2-5 PMN11060
020F 0  0007      1114          DC      'HDNG       *HARMLESS WRITE TO DISK     PMN11070
0210 0  00E8      1115    CR920 DC      $DBSY-$CH12 WD CNT AND SCTR             PMN11080
0211 0  0002      1116          DC      'RIAD       *ADDR OF RESIDENT IMAGE     PMN11090
0212    0212      1117          END     *                                       PMN11100
```

CROSS-REFERENCE

SYMBOL	VALUE	REL	DEFN	REFERENCES
CR010	01E2	0	1077	1084
CR905	020D	0	1112	1099
CR910	020E	0	1113	1095
CR920	0210	0	1115	1076 1081 1097
DZ000	00F2	0	0610	0323 0350 0627 0646 0707 0763 0955 0956 1007 1008 1082 1088
DZ010	00F7	0	0617	0628 0641
DZ020	00F9	0	0619	0613
DZ030	00FF	0	0625	0663
DZ060	0104	0	0630	0650
DZ070	0106	0	0634	0813 0819 0845 0853
DZ100	0108	0	0639	0619 0620 0715
DZ180	010E	0	0645	0618 0634
DZ185	0116	0	0651	0826
DZ190	0110	0	0660	0720
DZ210	0146	0	0711	0833
DZ215	014C	0	0719	0784
DZ220	014E	0	0724	0787
DZ230	0152	0	0730	0629
DZ235	0154	0	0731	0624 0654
DZ240	0159	0	0735	0864
DZ250	0168	0	0750	0758
DZ280	0169	0	0751	0737 0743 0744 0745 0756
DZ300	0174	0	0763	0753
DZ330	0179	0	0767	0739
DZ340	0188	0	0781	0726
DZ350	0190	0	0790	0662 0741 0824
DZ380	019D	0	0805	0802
DZ385	01A2	0	0810	
DZ390	01A5	0	0813	0809
DZ400	01AF	0	0831	0792
D7410	0181	0	0833	0858
DZ900	0122	0	0669	0677 0804
DZ901	0123	0	0670	0823
DZ902	0124	0	0671	0651 0733 0854 0861
DZ904	0126	0	0673	0635 0805 0812 0818 0841
DZ905	0127	0	0674	0850
DZ906	0128	0	0675	0747 0840
DZ907	0129	0	0676	0774
DZ908	012A	0	0677	0817
DZ909	012B	0	0678	0771
D7910	012C	0	0679	0647 0769 0781 0799
DZ911	012D	0	0680	0779
DZ912	012E	0	0681	0658 0734 0780 0831 0835 0856 0862
DZ913	012F	0	0682	0764
DZ914	0130	0	0683	0719
DZ915	0131	0	0684	0659
DZ916	0132	0	0685	0834
DZ920	0134	0	0687	0775
DZ925	0135	0	0688	0770
DZ930	0136	0	0689	0755
DZ935	0137	0	0690	0724
DZ940	0138	0	0691	0749
DZ945	0139	0	0692	0622 0773 0776 0851
DZ950	013A	0	0693	0778
DZ955	013B	0	0694	0765
DZ960	013C	0	0695	0742
DZ965	013D	0	0696	0738

SYMBOL	VALUE	REL	DEFN	REFERENCES						
DZ970	013E	0	0697	0740						
DZ975	013F	0	0698	0777	0849					
DZ980	0140	0	0699	0768						
DZ985	0141	0	0700	0838						
DZ990	0142	0	0701	0766	0772	0843				
DZ995	0143	0	0702	0803						
$ACDE	009F	0	0437	0696	0697	1080	1098			
$ACEX	001A	0	0950	0317						
$CCAD	0074	0	0364							
$CH12	0006	0	0254	1115						
$CIBA	0005	0	0253	0311	1077	1096				
$CIDN	01DB	0	0871	1083						
$CILA	005A	0	0951	0348						
$CLSW	0018	0	0948							
$COMN	0007	0	0255							
$CORE	000E	0	0267							
$CPTR	007E	0	0374							
$CTSW	000F	0	0268							
$CWCT	0072	0	0362							
$CXR1	0017	0	0947	0319						
$CYLN	009A	0	0427	0695	0697	1087				
$DADR	0010	0	0269							
$DBSY	00EE	0	0552	0611	0626	0653	0712	0954	1086	1115
$DCDE	0077	0	0367							
$DCYL	00A4	0	0448	0695	1078					
$DDSW	00DD	0	0530	0648	0782	0806				
$DMPF	0019	0	0949	0321						
$DREQ	0012	0	0271							
$DUMP	003F	0	0316	0320	0548	1105	1112			
$DZIN	0076	0	0366							
$EXIT	0038	0	0302							
$FLSH	0071	0	0360							
$FPAD	0095	0	0418							
$GCOM	0063	0	0354							
$GRIN	0064	0	0355							
$HASH	0014	0	0273	0945	0946	0947	0948	0949	0950	
$IBSY	0013	0	0272							
$IBT2	00B9	0	0952							
$IBT4	00D4	0	0953							
$IOCT	0032	0	0293	0330	0713	0730	1094			
$IREQ	002C	0	0287	0514						
$I200	00B3	0	0474	0261	0485					
$I205	00B8	0	0480	0952						
$I210	00BA	0	0481	0475	0476	0477				
$I290	00C2	0	0486	0478	0484					
$I400	00C4	0	0506	0263	0525					
$I403	00D0	0	0516	0513						
$I405	00D3	0	0520	0953						
$I410	00D5	0	0521	0508	0509	0510				
$I420	00E6	0	0545							
$I425	00EA	0	0548	0547						
$I490	00DE	0	0531	0507	0524					
$I492	00E0	0	0532	0511						
$I494	00E2	0	0535	0517						
$I496	00E4	0	0542	0546	0957					
$KCSW	007C	0	0372							
$LAST	0033	0	0294							
$LEV0	0008	0	0259							
$LEV1	0009	0	0260							
$LEV2	000A	0	0261							
$LEV3	000B	0	0262							
$LEV4	000C	0	0263							
$LEV5	000D	0	0264							
$LINK	0039	0	0306	0340						
$LKNM	0014	0	0945	0346						
$LSAD	0075	0	0365							
$NDUP	0034	0	0295							
$NXEQ	0035	0	0296							
$PAUS	00F0	0	0955							
$PBSY	0036	0	0297							
$PGCT	0037	0	0298							
$PHSE	0078	0	0368							
$PRET	0028	0	0282	0284	0725					
$PST1	0081	0	0380	0382						
$PST2	0085	0	0386	0388	0660					
$PST3	0089	0	0392	0394						
$PST4	008D	0	0398	0400						
$RMSW	0016	0	0946	0339						
$RWCZ	00F1	0	0956							

SYMBOL	VALUE	REL	DEFN	REFERENCES
$SCAN	0020	0	0276	
$SCAT	0011	0	0270	0331
$SNLT	00EF	0	0954	
$STOP	0091	0	0405	0264 0407
$SYSC	00E0	0	0533	
$S000	0052	0	0338	0302
$S100	0053	0	0339	0308 0325
$S150	0059	0	0345	0951
$S200	005E	0	0349	
$S250	004B	0	0329	0318 0333 0347 0351
$S300	004C	0	0330	0332
$S900	003C	0	0310	0322 0324
$S910	003E	0	0312	0338
$UFDR	007D	0	0373	
$UFIO	0079	0	0369	
$ULET	002D	0	0288	
$WRD1	007B	0	0371	
$WSDR	007A	0	0370	
$XR3X	00F4	0	0957	
$ZEND	01E0	0	0875	0604 0606
$1132	007F	0	0375	
$1403	0080	0	0376	
X2	00F2	0	0707	0660 0764 0765 0766 0768 0769 0770 0771 0772 0773 0774 0775 0776 0777 0778 0779 0780 0781 0782 0799 0803 0804 0805 0806 0812 0813 0817 0818 0819 0823 0831 0834 0835 0838 0840 0841 0843 0845 0849 0850 0851 0853 0854 0856 0861 0862
Y	007F	0	1012	1075 1078 1079 1080 1082 1086 1087 1088 1094 1096 1098 1100 1105
#ACIN	001C	0	0902	
#ANDU	0023	0	0908	
#BNDU	0028	0	0909	
#CBSW	000A	0	0885	
#CIAD	001B	0	0901	
#CIBA	003C	0	0913	
#CIDN	0037	0	0912	
#CSHN	005A	0	0919	
#DBCT	0006	0	0881	
#DCSW	0018	0	0898	
#ECNT	0021	0	0907	
#ENTY	0010	0	0891	
#FCNT	0007	0	0882	
#FHOL	0014	0	0894	
#FLET	004B	0	0916	
#FMAT	0046	0	0915	
#FPAD	002D	0	0910	
#FSZE	0015	0	0895	
#GCNT	001E	0	0904	
#GRPH	0010	0	0903	
#JBSW	0009	0	0884	
#LCNT	0008	0	0886	
#LOSW	001F	0	0905	
#MDF1	000D	0	0888	
#MDF2	000E	0	0889	
#MPSW	000C	0	0887	
#NAME	0004	0	0880	
#NCNT	000F	0	0890	
#PCID	0032	0	0911	
#PIOD	0019	0	0899	
#PPTR	001A	0	0900	
#RP67	0011	0	0892	
#SCRA	0041	0	0914	
#SYSC	0008	0	0883	
#TODR	0012	0	0893	
#UHOL	0016	0	0896	
#ULET	0050	0	0917	
#USZE	0017	0	0897	
#WSCT	0055	0	0918	
#X3SW	0020	0	0906	
*BLCT	0002	0	0997	
*CDCV	0093	0	0934	
*CIDN	0004	0	1014	
*CIL1	00A0	0	0940	0870
*CIL2	00A1	0	0941	
*CLR0	0078	0	0926	
*CMON	0001	0	0972	
*COM2	0380	0	1003	
*COM1	0400	0	1004	
*COM2	0600	0	1005	
*COPY	0005	0	1015	

SYMBOL	VALUE	REL	DEFN	REFERENCES
*CORE	001C	0	0985	
*CPTR	008E	0	0929	
*DCOM	0001	0	0962	
*DCTR	0001	0	1016	
*DKEP	00F9	0	1007	
*DKIP	00F7	0	1008	
*DNID	0098	0	0939	
*DREQ	0002	0	0973	
*DTYP	0008	0	1017	
*DZID	0096	0	0937	0605
*DIID	0097	0	0938	
*FILE	0003	0	0974	
*HCIB	0003	0	1010	
*HDNG	0007	0	0966	1114
*HEND	001D	0	0986	
*HWCT	0004	0	0975	
*IDAD	0000	0	0961	
*ILS4	0011	0	0983	
*ISTV	0033	0	1001	
*ITVX	000B	0	0982	
*KBCP	0092	0	0933	
*KBCV	0095	0	0936	
*LDAD	0006	0	0977	
*LFEN	0003	0	0991	
*LFHD	0005	C	0990	
*LFNM	0000	0	0996	
*LSCT	0005	0	0976	
*MCOR	1000	0	1011	
*MCRA	006E	0	0923	
*MXDR	0005	0	1002	
*NEXT	0004	0	0995	
*OVSW	001A	0	0984	
*PTCV	0094	0	0935	
*RIAD	0002	0	0963	1116
*RTBL	0006	0	0965	
*SCIB	0010	0	1009	
*SCTN	0000	0	0992	
*SLET	0003	0	0964	
*STRT	0000	0	0967	
*SUP6	0073	0	0924	
*SUP7	0074	0	0925	
*TCNT	0011	0	1006	0625
*TVWC	0008	0	0979	
*UAFX	0001	0	0993	
*WCNT	0009	0	0980	
*WDSA	0003	0	0994	
*XCTL	0007	0	0978	
*XEQA	0000	0	0971	
*XR3X	000A	0	0981	
*1132	008D	0	0928	
*1134	0091	0	0932	
*1403	008C	0	0927	
*1442	0090	0	0931	
*2501	008F	0	0930	

The addresses listed on the SLET printout are subject to change. Only the symbols and phase IDs will remain constant.

SYSTEM LOCATION EQUIVALENCE TABLE (SLET)

SYMBOL	PH ID	CORE ADDR	WORD COUNT	SCTR ADDR	SYMBOL	PH ID	CORE ADDR	WORD COUNT	SCTR ADDR	SYMBOL	PH ID	CORE ADDR	WORD COUNT	SCTR ADDR	SYMBOL	PH ID	CORE ADDR	WORD COUNT	SCTR ADDR
*****	**	****	****	****	*****	**	****	****	****	*****	**	****	****	****	*****	**	****	****	****
*DDUP	01	7C50	0327	0008	*DCTL	02	11DE	05A2	000B	*STOR	03	21DE	056B	0010	*FILQ	04	01DE	03C0	0015
*DUMP	05	41DE	0543	0018	*DL/F	06	01DE	03C0	001D	*DLTE	07	01DE	05A2	0020	*DFNE	08	01DE	05A2	0025
*EXIT	09	01DE	0500	002A	*CFCE	0A	7A06	00DB	002E	*DU11	0B	7A06	0035	002F	*DU12	0C	7A06	00D1	0030
*DU13	0D	7782	087C	0031	*DU14	0E	7A06	0248	0038	*DU15	0F	7A06	0248	003A	*DU16	10	7A06	0248	003C
*PRCI	11	01DE	0280	003E	*DU18	12	0E6E	0140	0040	*FR01	1F	761C	09E1	0041	*FR02	20	7A34	0500	0049
*FR03	21	7A34	0280	004D	*FR04	22	7A34	03C0	004F	*FR05	23	7A34	0500	0052	*FR06	24	7A34	03C0	0056
*FR07	25	7A34	0280	0059	*FR08	26	7A34	0500	005B	*FR09	27	7A34	03F0	005F	*FR10	28	7A34	03C0	0063
*FR11	29	7A34	03C0	0066	*FR12	2A	7A34	03C0	0069	*FR13	2B	7A34	03C0	006C	*FR14	2C	7A34	0500	006F
*FR15	2D	7A34	0500	0073	*FR16	2E	7A34	0500	0077	*FR17	2F	7A34	0500	007B	*FR18	30	7A34	0500	007F
*FR19	31	7A34	0404	0083	*FR20	32	7A34	03C0	0087	*FR21	33	7A34	03C0	008A	*FR22	34	7A34	0280	008D
*FR23	35	7A34	03C0	008F	*FR24	36	7A34	03C0	0092	*FR25	37	7A34	0500	0095	*FR26	38	7B8E	03C0	0099
*FR27	39	766E	0140	009C	*AS00	51	01E0	024F	009D	*ACNV	52	01E8	00BB	009F	*AS10	53	01E8	0067	00A0
*AS11	54	01E8	0050	00A1	*AS12	55	026A	0185	00A2	*AERM	56	0AC4	00C1	00A4	*AS01	57	026A	019A	00A5
*AS1A	58	026A	00B5	00A7	*ASYM	59	0000	0130	00A8	*AS03	5A	077A	0237	00A9	*AS04	5B	026A	01AF	00AB
*AS02	5C	026A	015B	00AD	*AS2A	5D	026C	00A6	00AF	*AS09	5E	0408	05ED	00B0	*AS05	5F	026A	019A	00B5
*AS06	60	026A	0196	00B7	*AS07	61	026A	0166	00B9	*AS7A	62	026C	0127	00BB	*AS08	63	026A	0191	00BC
*AS8A	64	026A	0199	00BE	*APCV	65	026A	0097	00C0	*AINT	66	095A	005B	00C1	*ASAA	67	095A	005E	00C2
*ASGR	68	0EB8	03BD	00C3	*SUP1	6E	04FE	02FE	00C6	*SUP2	6F	07FE	052B	00C9	*SUP3	70	07FE	0280	00CE
*SUP4	71	07FE	0280	00D0	*SUP5	72	07FE	03EA	00D2	*SUP6	73	0506	04F8	00D6	*SUP7	74	0400	0189	00DA
*CLB1	78	01E0	0642	00DC	*CLB2	79	0580	04E2	00E2	*CLB3	7A	087A	01E8	00E6	*CLB4	7B	087A	01E8	00E8
*CLB5	7C	087A	01E8	00EA	*CLB6	7D	087A	01E8	00EC	*CLB7	7E	0A64	0140	00EE	*CLB8	7F	0A64	0140	00EF
*CLB9	80	0A64	0140	00F0	*CLBA	81	0A64	0140	00F1	*CLBB	82	08A6	0140	00F2	*CLBC	83	087A	01E8	00F3
*CLBD	84	0A64	0140	00F5	*1403	8C	0000	0131	00F6	*1132	8D	0000	0126	00F7	*CPTR	8E	0000	0118	00F8
*2501	8F	0000	009C	00F9	*1442	90	0000	00AB	00FA	*1134	91	0000	016C	00FB	*KBCP	92	0000	0174	00FD
*CDCV	93	0000	00B8	00FF	*PTCV	94	0000	0003	0100	*KBCV	95	0000	0003	0101	*DZID	96	00F0	00EC	0102
*DIID	97	00F0	01A2	0103	*DNID	98	00F0	02B0	0105	*PPRT	99	0000	0131	00F6	*PIWK	9A	0000	009C	00F9
*PIXK	9B	0000	009C	00F9	*PCWK	9C	0000	00BB	00FF	*PCXK	9D	0000	00BB	00FF	*CIL1	A0	0000	0170	0108
*CIL2	A1	0000	01C0	010A	*RG00	B0	0212	0924	010C	*RG02	B1	0906	0854	0114	*RG04	B2	0906	0794	011B
*RG06	B3	0906	06E9	0122	*RG08	B4	0906	088D	0128	*RG10	B5	04A6	07E6	012F	*RG12	B6	073A	082D	0136
*RG14	B7	073A	06B3	013D	*RG16	B8	0762	0467	0143	*RG17	B9	0762	068A	0147	*RG19	BA	073A	0909	014D
*RG20	BB	073A	059E	0155	*RG21	BC	073A	06A0	015A	*RG22	BD	0782	0246	0160	*RG24	BE	0782	0657	0162
*RG26	BF	0782	0205	0168	*RG28	C0	0782	0457	016A	*RG32	C1	0782	06A4	016E	*RG34	C2	0782	0443	0174
*RG36	C3	0782	0236	0178	*RG38	C4	0782	04E3	017A	*RG40	C5	0782	054D	017E	*RG42	C6	0782	037C	0183
*RG44	C7	0782	05B2	0186	*RG46	C8	0782	04A8	018B	*RG52	C9	073A	0374	018F	*RG54	CA	073A	0615	0192
*RG58	CB	073A	05D5	0197	*RG60	CC	073A	00F4	019C										

Sample programs 1, 2 and 3 are provided with the Monitor system. The first is a FORTRAN compilation, the second is an assembly and the third is an RPG compilation (RPG is supplied with the card system only). All three programs are loaded and processed as monitor jobs and are listed on the principal printer.

The output of the FORTRAN problem is printed on the printer specified on the IOCS control card. The output of the Assembler problem is printed on the Console Printer. The output of the RPG problem is printed on the printer specified as the output device on a file description sheet.

Sample programs 3, 4, 5 and 6 are not provided with the Monitor system. They illustrate techniques described in the Programming Tips and Techniques section of this manual.

1. FORTRAN SAMPLE PROGRAM

The FORTRAN sample program is listed below as it runs on a 4K and 8K system (the LIST ALL card is removed on the 8K run). This program reads data cards supplied with the program and builds three files on disk, one in the User Area and two in Working Storage. The core and file maps for the program are described in the Programming Tips and Techniques section of this manual.

Card CHK13030

If printed output is on a 1403 Printer, change the entry from 1132 PRINTER to 1403 PRINTER.

If printed output is on the Console Printer, change the IOCS entry from 1132 PRINTER to TYPEWRITER.

Card CHK13040

If card input is from a 2501 Reader, change the IOCS entry from CARD to 2501 READER.

Card CHK13180

If card input is from a 2501 Reader, change M=2 to M=8.

Card CHK13190

If printer output is on a 1403 Printer, change L=3 to L=1.

If printer output is on a Console Printer, change L=3 to L=1.

The FORTRAN card sample program as supplied uses a 1442-6, or -7, and 1132 Printer and disk. The paper tape sample program uses an 1134 Paper Tape Reader, a Console Printer, and disk. If your system does not have the required configuration, it will be necessary to make changes to the program. These changes are listed below.

FORTRAN Sample Program Run on 4K

```
// JOB                                                    09/27/67          CHK12970

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        000F        000F        0000

// DUP                                                                      CHK12980

*STOREDATA  WS  UA  FILEA    2                                             CHK12990
CART ID 000F   DB ADDR  1AE0   DB CNT   0020

// * IBM 1130 FORTRAN SAMPLE PROGRAM                                       CHK13000

// FOR                                                                      CHK13010
*ONE WORD INTEGERS                                                        CHK13020
*IOCS(DISK,1132 PRINTER)                                                  CHK13030
*IOCS(CARD)                                                               CHK13040
*LIST ALL                                                                 CHK13060
C      IBM 1130 FORTRAN SAMPLE PROGRAM                                    CHK13070
C      SIMULTANEOUS EQUATION PROGRAM                                      CHK13080
C                                                                         CHK13084
       INTEGER V1,V2,V3                                                   CHK13086
       DIMENSION A(10,10),X(10),B(140)                                    CHK13090
       DEFINE FILE 101(1,100,U,V1),102(1,10,U,V2),103(1,100,U,V3)         CHK13095
  301  FORMAT (1H1,20X15HINCOMPATIBILITY)                                 CHK13100
  302  FORMAT (1H 20X41HMORE EQUATIONS THAN UNKNOWNS-NO SOLUTIONS)        CHK13110
  303  FORMAT (1H 20X46HMORE UNKNOWNS THAN EQUATIONS-SEVERAL SOLUTIONS)   CHK13120
  304  FORMAT (1H 20X15HSOLUTION MATRIX)                                  CHK13130
  305  FORMAT (1H 20X8HMATRIX A)                                          CHK13140
  306  FORMAT (1H 20X8HMATRIX B)                                          CHK13150
  307  FORMAT (1H 20X10H A-INVERSE)                                       CHK13160
  308  FORMAT (1H 20X24HDIAGONAL ELEMENT IS ZERO)                         CHK13170
       M=2                                                                CHK13180
       L=3                                                                CHK13190
       READ (M,10)                                                        CHK13200
   10  FORMAT(80H           SPACE FOR TITLE                               CHK13210
     1                        )                                          CHK13220
       WRITE (L,10)                                                       CHK13230
   12  FORMAT (6I10,20X)                                                  CHK13240
       READ (M,12) M1,M2,L1,L2,N1,N2                                      CHK13250
C                                                                         CHK13255
C      M1 = NO. OF ROWS OF A                                              CHK13260
C      M2 = NO. OF COLS OF A                                              CHK13270
C      L1 = NO. OF ROWS OF X                                              CHK13280
C      L2 = NO. OF COLS OF X                                              CHK13290
C      N1 = NO. OF ROWS OF B                                              CHK13300
C      N2 = NO. OF COLS OF B                                              CHK13310
C                                                                         CHK13315
   13  FORMAT (7F10.4,10X)                                                CHK13320
   17  FORMAT (10F10.4)                                                   CHK13330
       IF (N2-1)63,64,63                                                  CHK13340
   64  IF (L2-1)63,65,63                                                  CHK13350
   65  IF (L1-M2)63,66,63                                                 CHK13360
   66  IF (M1-N1)63,11,63                                                 CHK13370
   63  WRITE (L,301)                                                      CHK13380
       GO TO 2                                                            CHK13390
   11  N=M1                                                               CHK13400
       N=M2                                                               CHK13410
       IF (M1-M2) 91,14,93                                                CHK13420
   91  WRITE (L,302)                                                      CHK13430
       GO TO 2                                                            CHK13440
   93  WRITE (L,303)                                                      CHK13450
       GO TO 2                                                            CHK13460
   14  WRITE (L,305)                                                      CHK13470
       DO 70 I=1,N                                                        CHK13480
       READ  (M,13) (A(I,J), J=1,N)                                       CHK13490
       WRITE (L,17) (A(I,J), J=1,N)                                       CHK13500
       WRITE (101'1)(A(I,J), J=1,N)                                       CHK13505
   70  CONTINUE                                                           CHK13510
   89  FORMAT (F10.4,70X)                                                 CHK13520
       WRITE (L,306)                                                      CHK13530
       READ  (M,89) (B(I), I=1,N)                                         CHK13540
       WRITE (L,89) (B(I), I=1,N)                                         CHK13550
       WRITE (102'1)(B(I), I=1,N)                                         CHK13554
C                                                                         CHK13556
C      INVERSION OF A                                                     CHK13560
C                                                                         CHK13565
       DO 120 K=1,N                                                       CHK13570
       D=A(K,K)                                                           CHK13580
       IF(D)40,200,40                                                     CHK13590
   40  A(K,K)=1.0                                                         CHK13600
       DO 60 J=1,N                                                        CHK13610
   60  A(K,J)=A(K,J)/D                                                    CHK13620
       IF(K-N)80,130,130                                                  CHK13630
   80  IK=K+1                                                             CHK13640
       DO 120 I=IK,N                                                      CHK13650
       D=A(I,K)                                                           CHK13660
       A(I,K)=0.0                                                         CHK13670
```

```
      DO 120 J=1,N                                                    CHK13680
  120 A(I,J)=A(I,J)-(D*A(K,J))                                        CHK13690
C                                                                     CHK13695
C     BACK SOLUTION                                                   CHK13700
C                                                                     CHK13705
  130 IK=N-1                                                          CHK13710
      DO 180 K=1,IK                                                   CHK13720
      I1=K+1                                                          CHK13730
      DO 180 I=I1,N                                                   CHK13740
      D=A(K,I)                                                        CHK13750
      A(K,I)=0.0                                                      CHK13760
      DO 180 J=1,N                                                    CHK13770
  180 A(K,J)=A(K,J)-(D*A(I,J))                                        CHK13780
      GO TO 202                                                       CHK13790
  200 WRITE (L,308)                                                   CHK13800
      GO TO 2                                                         CHK13810
  202 WRITE (L,307)                                                   CHK13820
      DO 201 I=1,N                                                    CHK13830
      WRITE (L,17)  (A(I,J), J=1,N)                                   CHK13840
      WRITE (103'1) (A(I,J), J=1,N)                                   CHK13845
  201 CONTINUE                                                        CHK13850
      DO 21 I=1,N                                                     CHK13860
      X(I)=0.0                                                        CHK13870
      DO 21 K=1,N                                                     CHK13880
   21 X(I)=X(I)+A(I,K)*B(K)                                           CHK13890
      WRITE (L,304)                                                   CHK13900
      WRITE (L,89) (X(I), I=1,N)                                      CHK13910
    2 CALL EXIT                                                       CHK13940
      END                                                             CHK13950
VARIABLE ALLOCATIONS
     A(R )=00DC-0016      X(R )=00F0-00DE      B(R )=0208-00F2      D(R )=020A          V1(I )=020C          V2(I )=020D
    V3(I )=020E           M(I )=020F           L(I )=0210          M1(I )=0211          M2(I )=0212          L1(I )=0213
    L2(I )=0214          N1(I )=0215          N2(I )=0216           N(I )=0217           I(I )=0218           J(I )=0219
     K(I )=021A          IK(I )=021B          I1(I )=021C

STATEMENT ALLOCATIONS
 301  =022A  302  =0237  303  =0251  304  =026D  305  =027A  306  =0283  307  =028C  308  =0296  10   =02A7  12   =09D1
 13   =02D5  17   =02D9  89   =02DC  64   =031C  65   =0322  66   =0328  63   =032E  11   =0334  91   =0344  93   =004A
 14   =0350  70   =03A2  40   =0407  60   =0416  80   =0432  120  =0451  130  =0484  180  =04AD  200  =04E2  202  =0=E8
 201  =0522  21   =053C  2    =0588

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CALLED SUBPROGRAMS
 FADDX   FMPYX   FDIV    FLD     FLDX    FSTO    FSTOX   FSBRX   CARDZ   PRNTZ   SRED    SWRT    SCOMP   SFIO    SIO1X
 SIOI    SUBSC   SDFIO   SDWRT   SDCOM   SDFX

REAL CONSTANTS
  .100000E 01=0220    .000000E 00=0222

INTEGER CONSTANTS
     2=0224      3=0225      1=0226    101=0227    102=0228    103=0229

CORE REQUIREMENTS FOR
 COMMON       0  VARIABLES    544  PROGRAM    874

 END OF COMPILATION

// XEQ       L  2                                                     CHK13960
*LOCAL,FLOAT,FARC,IFIX                                                CHK13963
*FILES(103,FILEA)                                                     CHK13965
FILES ALLOCATION
  103 01AE  0001  000F FILEA
  101 0000  0001  000F  01B0
  102 0001  0001  000F  01B0
STORAGE ALLOCATION
R 40  03AB (HEX) ADDITIONAL CORE REQUIRD
R 43  0124 (HEX) ARITH/FUNC SOCAL WD CNT
R 44  06AC (HEX) FI/O, I/O SOCAL WD CNT
R 45  02A2 (HEX) DISK FI/O SOCAL WD CNT
R 41  0004 (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 EBCTB  0F53 SOCAL 2
 HOLTB  0F17 SOCAL 2
 GETAD  0ED4 SOCAL 2
 XMDS   09B2 SOCAL 1
 HOLEZ  0E9E SOCAL 2
 NORM   07DC
 FADDX  095D SOCAL 1
 FSBRX  0934 SOCAL 1
 FMPYX  0900 SOCAL 1
 FDIV   08AE SOCAL 1
 FSTOX  0788
 FLDX   07A4
 SDCOM  0920 SOCAL 3
 SDFX   08E6 SOCAL 3
```

```
  SDWRT   0954 SOCAL 3
  SIOFX   099A SOCAL 2
  SUBSC   07BE
  SIOI    099E SOCAL 2
  SCOMP   0982 SOCAL 2
  SWRT    08AB SOCAL 2
  SRED    08B0 SOCAL 2
  FSTO    078C
  FLD     07A8
  PRNTZ   0DE0 SOCAL 2
  CARDZ   0D36 SOCAL 2
  SFIO    09AD SOCAL 2
  SDFIO   0959 SOCAL 3
  IFIX    087C LOCAL
  FARC    087C LOCAL
  FLOAT   087C LOCAL
 SYSTEM SUBROUTINES
  ILS04   00C4
  ILS02   00B3
  ILS01   0F5A
  ILS00   0F75
  FLIPR   0816
      04DD (HEX) IS THE EXECUTION ADDR

     IBM 1130 FORTRAN          SAMPLE PROGRAM                    CHK1397
                      MATRIX A
    4.2150   -1.2120    1.1050
   -2.1200    3.5050   -1.6320
    1.1220   -1.3130    3.9860
                      MATRIX B
    3.2160
    1.2470
    2.3456
                       A-INVERSE
    0.2915    0.0833   -0.0467
    0.1631    0.3836    0.1118
   -0.0283    0.1029    0.3008
                      SOLUTION MATRIX
    0.9321
    1.2654
    0.7429
```

FORTRAN Sample Program Run on 8K

```
// JOB                                              09/27/67        CHK12970

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        000F        000F        0000

// * IBM 1130 FORTRAN SAMPLE PROGRAM                                CHK13000

// FOR                                                             CHK13010
*ONE WORD INTEGERS                                                 CHK13020
*IOCS(DISK,1132 PRINTER)                                          CHK13030
*IOCS(CARD)                                                       CHK13040

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR
 COMMON       0  VARIABLES    544  PROGRAM    874

 END OF COMPILATION

// XEQ       L  1                                                 CHK13960
*FILES(103,FILEA)                                                 CHK13965
FILES ALLOCATION
  103 01AE  0001  000F FILEA
  101 0000  0001  000F  01B0
  102 0001  0001  000F  01B0
STORAGE ALLOCATION
R 41  0C9C (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 EBCTB  12CD
 HOLTB  1291
 GETAD  124E
 NORM   1224
 XMDS   1208
 FARC   11E6
 HOLEZ  11B0
 FLOAT  11A6
 IFIX   117A
 FADDX  1125
 FSBRX  10FC
```

```
 FMPYX   10C8
 FDIV    1076
 FSTOX   101E
 FLDX    103A
 SDCOM   07FE
 SDFX    07C4
 SDWRT   0832
 SIOFX   081A
 SUBSC   1054
 SIOI    081E
 SCOMP   0802
 SWRT    0A28
 SRED    0A30
 FSTO    1022
 FLD     103E
 PRNTZ   0F60
 CARDZ   0E86
 SFIO    082D
 SDFIO   0837
SYSTEM SUBROUTINES
 ILS04   00C4
 ILS02   0083
 ILS01   12D2
 ILS00   12ED
      04DD (HEX) IS THE EXECUTION ADDR

     IBM 1130 FORTRAN          SAMPLE PROGRAM                    CHK1397
                      MATRIX A
   4.2150    -1.2120     1.1050
  -2.1200     3.5050    -1.6320
   1.1220    -1.3130     3.9860
                      MATRIX B
   3.2160
   1.2470
   2.3456
                       A-INVERSE
   0.2915     0.0833    -0.0467
   0.1631     0.3836     0.1118
  -0.0283     0.1029     0.3008
                     SOLUTION MATRIX
   0.9321
   1.2654
   0.7429
```

2. ASSEMBLY SAMPLE PROGRAM

The core map for the Assembler sample program is described in the Programming Tips and Techniques section of this manual.

Output on Principal Printer

```
// JOB                                                              SMASM001

LOG DRIVE   CART SPEC   CART AVAIL   PHY DRIVE
  0000        2027        2027         0002

// ASM                                                              SMASM002
*LIST                                                              SMASM003
*PRINT SYMBOL TABLE                                                SMASM004


                         COMPUTE THE SQUARE ROOT OF 64

                      ************************************************** SMASM006
                      *                                                * SMASM007
                      *   THIS PROGRAM COMPUTES THE SQUARE ROOT OF 64  * SMASM008
                      *   *AND PRINTS THE RESULT ON THE CONSOLE PRINTER.* SMASM009
                      *                                                * SMASM010
                      ************************************************** SMASM011
0000 0  C030          BEGIN LD      D64       INPUT TO THE SQUARE ROOT   SMASM012
0001 20 064D6063            LIBF    FLOAT     INTEGER TO FLOATING PT.    SMASM013
0002 30 06898640            CALL    FSQR      FLOATING PT. SQRT.         SMASM014
0004 20 091899C0            LIBF    IFIX      FLOATING PT. TO INTEGER    SMASM015
0005 0  1008                SLA     8                                    SMASM016
                      *             MASK TO BUILD EBCDIC INTEGER         SMASM017
                      *             RESULT AND EBCDIC BLANK IN WORD1.    SMASM018
0006 0  E829                OR      MASK                                 SMASM019
0007 0  D01B                STO     WORD1     CONVERSION INPUT AREA      SMASM020
                      *             CONVERT MESSAGE FROM EBCDIC          SMASM021
                      *             TO ROTATE/TILT CODE.                 SMASM022
```

```
0008 20 05097663          LIBF    EBPRT     CALL CONVERSION SUBROUTINE  SMASM023
0009 0  0000              DC      0         CONTROL PARAMETER           SMASM024
000A 1  0023              DC      WORD1     INPUT AREA                  SMASM025
000B 1  0015              DC      TYPE+1    OUTPUT AREA                 SMASM026
000C 0  001A              DC      26        CHARACTER COUNT             SMASM027
000D 20 23A17170          LIBF    TYPEO     TYPE MESSAGE                SMASM028
000E 0  2000              DC      /2000     CONTROL PARAMETER           SMASM029
000F 1  0014              DC      TYPE      I/O AREA                    SMASM030
0010 20 23A17170    BUSY  LIBF    TYPEO     WAIT FOR TYPING COMPLETE    SMASM031
0011 0  0000              DC                                            SMASM032
0012 0  70FD              MDX     BUSY      BR TO WAIT FOR COMPLETION   SMASM033
0013 0  6038              EXIT              RETURN TO MONITOR CONTROL   SMASM034
0014 0  000E        TYPE  DC      14        I/O AREA WORD COUNT         SMASM035
0015    000D              BSS     13        RESERVE AS PRINT BUFFER     SMASM036
0022 0  8181              DC      /8181     TWO CARRIAGE RETURNS        SMASM037
0023 0  0000        WORD1 DC      *-*       CONVERSION INPUT AREA       SMASM038
0024    0018              EBC     .IS THE SQUARE ROOT OF 64.            SMASM039
0030 0  F040        MASK  DC      /F040     EBCDIC INTEGER MASK         SMASM040
0031 0  0040        D64   DC      64        CONSTANT FOR SQUARE ROOT    SMASM041
0032    0000              END     BEGIN                                 SMASM042


                          *SYMBOL TABLE*

   BEGIN  0000     BUSY   0010     D64    0031     MASK   0030     TYPE   0014
   WORD1  0023

   000 OVERFLOW SECTORS SPECIFIED
   000 OVERFLOW SECTORS REQUIRED
   006 SYMBOLS DEFINED
    NO ERROR(S)  FLAGGED IN ABOVE ASSEMBLY

// XEQ       L                                                          SMASM043
R 41  7904 (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR
 FSQR   0248
LIBF TRANSFER VECTOR
 FARC   069E
 XMDS   0682
 HOLL   0632
 PRTY   05E2
 EBPA   0592
 FADD   04E1
 FDIV   0540
 FLD    048C
 FADDX  04E7
 FMPYX  04A2
 FSTO   0470
 FGETP  0456
 NORM   042C
 TYPEO  0312
 EBPRT  02AC
 IFIX   0280
 FLOAT  0230
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  0083
      01FE (HEX) IS THE EXECUTION ADDR
```

Output on Console Printer

```
8 IS THE SQUARE ROOT OF 64
```

3. RPG SAMPLE PROGRAM

The RPG sample program is defined for 2501 input and 1132 output. For systems with other configurations the input and output file specifications can be changed. Use READ01 for a 2501 Card Read Punch, PRINT03 for a 1403 Printer and CONSOLE for a Console Printer.

Output on Principal Printer.

```
// JOB

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        1219        1219        0002

// RPG
```

```
                                V1-0  1130  RPG   RGSPL

SEQ NO  PG LIN    SPECIFICATIONS COL 6 - 74                                    ERRORS

                  H                                                            RGSPL
                  F*  1130 RPG SAMPLE PROGRAM.                                 RGS001
                  F*  THIS PROGRAM PRINTS AN ACCOUNTS RECEIVABLE REGISTER WITH RGS002
                  F*  INVOICE TOTALS . CUSTOMER TOTALS PRINT AS A RESULT OF A CONTROL RGS003
                  F*  BREAK IN COLUMNS 39-43 OF THE INPUT CARD. CORRECT OUTPUT  RGS004
                  F*  APPEARS IN ACCOMPANYING DOCUMENTATION. CARDS ARE SORTED ON RGS005
                  F*  COLUMNS 39-43 AND ARE IDENTIFIED BY AN ELEVEN PUNCH IN CARD RGS006
                  F*  COLUMN 1.                                                 RGS007
                  F*                                                           RGS008
 0001   01 010    FINPUT   IPE F      80            READ 42                    RGS009
 0002   01 020    FOUTPUT  O   F     120            PRINTER                    RGS010
 0003   02 010    IINPUT   AA  01   1 Z-                                       RGS011
 0004   02 020    I                                    8  29 NAME             RGS012
 0005   02 030    I                                   30  310MONTH            RGS013
 0006   02 040    I                                   32  330DAY              RGS014
 0007   02 050    I                                   34  360INVNO            RGS015
 0008   02 060    I                                   39  430CUSTNCL1         RGS016
 0009   02 070    I                                   44  450STATE            RGS017
 0010   02 080    I                                   46  480CITY             RGS018
 0011   02 090    I                                   74  802INVAMT           RGS019
 0012   03 010    C   01      INVAMT    ADD  TOTAL     TOTAL   82             RGS020
 0013   03 020    C   01      INVAMT    ADD  GRPTOT    GRPTOT  82             RGS021
 0014   04 010    OOUTPUT  H  201   1P                                        RGS022
 0015   04 020    O        OR        OF                                        RGS023
 0016   04 030    O                                   53 '      A C C O U N T S  R'  RGS024
 0017   04 040    O                                   77 ' E C E I V A B L E  R E '  RGS025
 0018   04 050    O                                   88 'R E G I S T E R'     RGS026
 0019   04 060    O         H  1     1P                                        RGS027
 0020   04 070    O        OR        OF                                        RGS028
 0021   04 080    O                                   25 'CUSTOMER'           RGS029
 0022   04 090    O                                   80 'LOCATION        INVOICE'  RGS030
 0023   04 100    O                                  109 'INVOICE DATE     INVOICE'  RGS031
 0024   04 110    O         H  2     1P                                        RGS032
 0025   04 120    O        OR        OF                                        RGS033
 0026   04 130    O                                   42 'NUMBER          CUSTOMER '  RGS034
 0027   04 140    O                                   46 'NAME'               RGS035
 0028   04 150    O                                   70 'STATE   CITY       NUMBER'  RGS036
 0029   04 160    O                                  108 'NO      DAY      AMOUNT'  RGS037
 0030   05 010    O         D  2     01                                        RGS038
 0031   05 020    O                        CUSTNOZ    22                      RGS039
 0032   05 030    O                        NAME       53                      RGS040
 0033   05 040    O                        STATE Z    59                      RGS041
 0034   05 050    O                        CITY  Z    67                      RGS042
 0035   05 060    O                        INVNO Z    79                      RGS043
 0036   05 070    O                        MONTH Z    89                      RGS044
 0037   05 080    O                        DAY   Z    97                      RGS045
 0038   05 090    O                        INVAMT     109 '$    .  0.  '      RGS046
 0039   05 100    O         T  2     L1                                        RGS047
 0040   05 110    O                        GRPTOT B   109 '$    .  0.  '      RGS048
 0041   05 120    O                                   110 '*'                 RGS049
 0042   05 130    O         T  2     LR                                        RGS050
 0043   05 140    O                        TOTAL      109 '$    .  0.  '      RGS051
 0044   05 150    O                                   111 '**'                RGS052
```

```
                                    INDICATORS

INC   DISP      IND   DISP      IND   DISP      IND   DISP      IND   DISP      IND   DISP
MR    0150      00    0151      OF    0152      OV    0153      1P    0154      L0    0156
L1    0156      L2    0157      L3    0158      L4    0159      L5    015A      L6    015B
L7    015C      L8    015D      L9    015E      LR    015F      H1    0160      H2    0161
H3    0162      H4    0163      H5    0164      H6    0165      H7    0166      H8    0167
H9    0168      01    0169

                                  FIELD NAMES

FIELD   DISP  L  T D    FIELD   DISP  L  T D    FIELD   DISP  L  T D    FIELD   DISP  L  T D
NAME    016A 022 A      MONTH   0181 002 N 0    DAY     0184 002 N 0    INVNO   0187 005 N 0
CUSTNO  018D 005 N 0    STATE   0193 002 N 0    CITY    0196 003 N 0    INVAMT  019A 007 N 2
TOTAL   01A2 008 N 2    GRPTOT  01AB 008 N 2

                                    LITERALS

       LITERAL           LENGTH   TYPE   DISP          LITERAL          LENGTH   TYPE   DISP
     A C C O U N T S  R    24       A     01B4    E C E I V A B L E  R E   24       A     01CD
R E G I S T E R           15       A     01E6    CUSTOMER                  8       A     01F6
LOCATION      INVOICE     22       A     01FF    INVOICE DATE    INVOICE  23       A     0216
NUMBER        CUSTOMER    24       A     022E    NAME                      4       A     0247
STATE   CITY      NUMBER    24       A     024C    MO     DAY     AMOUNT    21       A     0265
    *   *                 11       E     0278    *                         1       A     0287
**                         2       A     0286

                          KEY ADDRESSES OF OBJECT PROGRAM

  NAME OF ROUTINE          HEX DISP          NAME OF ROUTINE          HEX DISP

    H + D LINES              04DE              TOTAL LINES              04EC
    DETAIL CALCS             046E              TOTAL CALCS              047D
    CHAIN ROUT 1             03D2              CONTROL FLD              03F5
    LOW FIELD                042E              EXCPT LINES              04FA
    CLOSE FILES              0684              FILE SEQ 1               02EC
    FILE SEQ 2               0377

END OF COMPILATION

// XEQ       L             R
R 41  0D4C (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR
 RGERR  1180
 EBPT3  0B86
 HLEBC  0984
LIBF TRANSFER VECTOR
 RGSI5  11C6
 RGBLK  1146
 RGEDT  0FF6
 RGMV2  0F42
 RGADD  0D79
 RGSI1  0D1C
 RGMV5  0C0E
 RGMV3  0CEC
 RGCMP  0C9A
 RGMV1  0C06
 PRNT1  0A04
 ZIPCO  08E4
 READ0  0884
SYSTEM SUBROUTINES
 ILSX4  122B
 ILSX2  124D
 ILSX1  1266
      020F (HEX) IS THE EXECUTION ADDR
```

Output on 1132.

ACCOUNTS RECEIVABLE REGISTER

CUSTOMER NUMBER	CUSTOMER NAME	LOCATION STATE	LOCATION CITY	INVOICE NUMBER	INVOICE DATE MO	INVOICE DATE DAY		INVOICE AMOUNT
10712	AMALGAMATED CORP	33	61	11603	11	10	$	389.25
							$	389.25*
11315	BROWN WHOLESALE	30	231	12324	12	28	$	802.08
11315	BROWN WHOLESALE	30	231	99588	12	14	$	261.17
							$	1,063.25*
11897	FARM IMPLEMENTS	47	77	10901	10	18	$	27.63
							$	27.63*
18530	BLACK OIL	16	67	11509	11	6	$	592.95
18530	BLACK OIL	16	67	12292	12	23	$	950.97
							$	1,543.92*
20716	LEATHER BELT CO	36	471	11511	11	8	$	335.63
20716	LEATHER BELT CO	36	471	12263	12	17	$	121.75
							$	457.38*
29017	GENERAL MFG CO	6	63	11615	11	14	$	440.12
29017	GENERAL MFG CO	6	63	11676	11	23	$	722.22
							$	1,162.34*
29054	A-B-C DIST CO	25	39	9689	9	11	$	645.40
29054	A-B-C DIST CO	25	39	11605	11	11	$	271.69
29054	A-B-C DIST CO	25	39	12234	12	14	$	559.33
							$	1,476.42*
							$	6,120.19**

4. USING FORTRAN UNFORMATTED I/O

This program is referred to in the section "Initializing $$$$$ Data Files for Use with FORTRAN Unformatted I/O" in Programming Tips and Techniques.

```
// JOB    1111

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        1111        1111        0001

// DUP

*STOREDATA  WS  FX  $$$$$ 0010
CART ID 1111   DB ADDR  1FA0   DB CNT   00A0

// FOR
*IOCS(UDISK)
*LIST ALL
*NAME UNFOX
      DIMENSION A(200),B(24),C(300),E(12),F(300)
      DATA A/200*4.0/,B/24*5.0/,C/300*6.0/
      WRITE (10)A
      WRITE(10)B
      WRITE(10)C
      END FILE 10
      BACKSPACE 10
      BACKSPACE 10
      READ(10)F
      REWIND 10
      READ(10)
      READ(10)E
      PAUSE 9999
      CALL EXIT
      END
VARIABLE ALLOCATIONS
     A(R )=018E-0000     B(R )=01BE-0190     C(R )=0416-01C0     E(R )=042E-0418     F(R )=0686-0430

FEATURES SUPPORTED
 IOCS

CALLED SUBPROGRAMS
 URED    UWRT    UCOMP   BCKSP   EOF     REWND   PAUSE   UFIO    UIOAF

INTEGER CONSTANTS
    10=0688   9999=0689 -26215=068A

CORE REQUIREMENTS FOR UNFOX
 COMMON      0  VARIABLES   1672  PROGRAM     52

 END OF COMPILATION

// DUP

*STORE      WS  UA  UNFOX
CART ID 1111   DB ADDR  2B50   DB CNT   0041

// XEQ UNFOX
```

5. PROCESSING ON ONE DISK DRIVE A FILE THAT EXTENDS OVER TWO CARTRIDGES

This program is referred to in the section "Reeling" in Programming Tips and Techniques.

```
// JOB    1111

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        1111        1111        0000

// FOR
*NAME LINK2
*IOCS(1132 PRINTER)
*IOCS (DISK)
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      DIMENSION J(320)
      DEFINE FILE 2(200,320,U,K)
      K = 1
      L = 0
      DO 5 I = 1,199
      L = L+1
      DO 4 N = 1,320
4     J(N) = L
5     WRITE (2'K)J
      L = 999
      DO 6 N = 1,320
6     J(N) = L
      WRITE (2'K)J
      WRITE(3,10)
10    FORMAT(/' LINK NO. 2 EXECUTED.'/)
      CALL EXIT
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR LINK2
 COMMON      0  VARIABLES    334  PROGRAM    142

 END OF COMPILATION

// DUP

*DUMP       WS  CD  LINK2
CART ID 1111  DB ADDR 2A60   DB CNT 000B

// FOR
*NAME LINK1
*IOCS(DISK,1132 PRINTER)
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      DIMENSION J(320)
      DIMENSION L(2)
      DEFINE FILE 1(210,320,U,K)
      K=1
      L(2) = 8738
      L(1) = 0
      M=0
      DO 5 I = 1,209
      M = M+1
      DO 4 N = 1,320
4     J(N) = M
5     WRITE (1'K)J
      M= 999
      DO 6 N = 1,320
6     J(N) = M
      WRITE (1'K)J
      WRITE (3,40)
40    FORMAT (40HOLINK NO. 1 EXECUTED. CHANGE CARTRIDGES.///)
      PAUSE 1111
      CALL SYSUP (L(2))
      CALL LINK (LINK2)
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR LINK1
 COMMON      0  VARIABLES    336  PROGRAM    178

END OF COMPILATION
```

```
// DUP

*STORECI    WS  UA  LINK1 0001
*FILES(1,DATA1,1111)
FILES ALLOCATION
    1 0128  00D2  1111 DATA1
STORAGE ALLOCATION
R 41  0C02 (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR
 FSYSU  1359
 FSLEN  11A1
 SYSUP  0C5A
LIBF TRANSFER VECTOR
 NORM   1380
 FLOAT  1196
 IFIX   116A
 PAUSE  0C44
 SCOMP  077E
 SWRT   06A2
 SDCOM  0476
 SDAI   0433
 SDWRT  04AA
 SUBSC  0C26
 FSTO   0BF4
 FLD    0C10
 PRNTZ  0B32
 SFIO   07B7
 SDFIO  04AF
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  00B3
 ILS01  13AC
      036D (HEX) IS THE EXECUTION ADDR
CART ID 1111   DB ADDR  2A60   DB CNT   00F0

// PAUS             CHANGE TO CARTRIDGE 2222
// JOB    2222

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        2222        2222        0000

// DUP

*STORECI    CD  FX  LINK2 0001
*FILES(2,DATA2,2222)
FILES ALLOCATION
    2 00A8  00C8  2222 DATA2
STORAGE ALLOCATION
R 41  1334 (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 NORM   0C54
 FLOAT  0C4A
 IFIX   0C1E
 SCOMP  0758
 SWRT   067C
 SDCOM  0450
 SDAI   040D
 SDWRT  0484
 SUBSC  0C00
 FSTO   0BCE
 FLD    0BEA
 PRNTZ  0B0C
 SFIO   0791
 SDFIO  0489
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  00B3
 ILS01  0C80
      0362 (HEX) IS THE EXECUTION ADDR
CART ID 2222   DB ADDR  1700   DB CNT   0090

// PAUS             CHANGE TO CARTRIDGE 1111

// JOB    1111

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        1111        1111        0000

// XEQ LINK1


LINK NO. 1 EXECUTED. CHANGE CARTRIDGES.


LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        2222        2222        0000


LINK NO. 2 EXECUTED.
```

6. PROCESSING ON TWO DISK DRIVES A FILE THAT EXTENDS OVER TWO CARTRIDGES

This program is referred to in the section "Reeling" in Programming Tips and Techniques.

```
// JOB    1111 2222

LOG DRIVE   CART SPEC   CART AVAIL  PHY DRIVE
  0000        1111        1111        0001
  0001        2222        2222        0002

// FOR
*NAME MDEX1
*IOCS (DISK)
*ONE WORD INTEGERS
*LIST SOURCE PROGRAM
      DIMENSION J(320)
      DEFINE FILE 1(210,320,U,K)
      DEFINE FILE 2(200,320,U,KK)
      M = 111
      K = 1
      KK = 1
      DO 2 N = 1,320
2     J(N) = M
      DO 3 I = 1,209
3     WRITE (1'K)J
      M = 999
      DO 5 N = 1,320
5     J(N) = M
      WRITE (1'K)J
      M = 222
      DO 7 N = 1,320
7     J(N) = M
      DO 8 I = 1,199
8     WRITE (2'KK)J
      M = 999
      DO 9 N = 1,320
9     J(N) = M
      WRITE (2'KK)J
      CALL EXIT
      END

FEATURES SUPPORTED
 ONE WORD INTEGERS
 IOCS

CORE REQUIREMENTS FOR MDEX1
 COMMON       0  VARIABLES    340  PROGRAM    178

 END OF COMPILATION

// DUP

*STORE      WS  UA  MDEX1
CART ID 1111   DB ADDR  2B91   DB CNT   000D

// XEQ MDEX1 L  2
*FILES(1,DATA1,1111)
*FILES(2,DATA2,2222)
FILES ALLOCATION
    1 0128  00D2  1111 DATA1
    2 00A8  00C8  2222 DATA2
STORAGE ALLOCATION
R 41  1926 (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
 SDCOM  047A
 SDAI   0437
 SDWRT  04AE
 SUBSC  06A6
 SDFIO  04B3
SYSTEM SUBROUTINES
 ILS04  00C4
 ILS02  00B3
      035A (HEX) IS THE EXECUTION ADDR
```

7. CALCULATING ISAM FILE PARAMETERS

This program is referred to in the section "Calculating ISAM File Size" in Programming Tips and Techniques.

The Console printer output and source listing for a program that calculates ISAM file size is shown below. The user is requested to enter the first four values. The input fields are five positions. Enter right-justified decimal numbers. Leading zeros are required. Press EOF on the keyboard after each entry. This program does no error checking.

User-entries are required for the following printouts.

```
            ISAM FILE LOAD CALCULATIONS

INDEX ENTRY LENGTH IN WORDS =
RECORD LENGTH IN WORDS =
NUMBER OF RECORDS TO BE LOADED =
NUMBER OF OVERFLOW SECTORS =
```

After the number of overflow sectors is entered the program calculates the file size.

Sample Output

```
            ISAM FILE LOAD CALCULATIONS

INDEX ENTRY LENGTH IN WORDS = 00050
RECORD LENGTH IN WORDS = 00100
NUMBER OF RECORDS TO BE LOADED = 01000
NUMBER OF OVERFLOW SECTORS = 000080
NUMBER OF INDEXES PER SECTOR = 00006
NUMBER OF RECORDS PER SECTOR = 00003
NUMBER OF PRIME DATA CYLINDERS = 00042
NUMBER OF PRIME DATA SECTORS = 00334
NUMBER OF INDEX SECTORS = 00007
TOTAL NUMBER OF SECTORS = 00422
```

Program Listing

```
0000 20  23A17170    START LIBF       TYPE0
0001 0   2000              DC         /2000     TYPE HEADING LINE
0002 1   001B              DC         HEAD
0003 20  23A17170    WAIT4 LIBF       TYPE0
0004 0   0000              DC         /0000     WAIT FOR CONSOLE
0005 0   70FD              B          WAIT4
0006 00  65000004          LDX   L1   4         SET COUNT
0008 01  C5000016    IN    LD    L1   BTAB1     LOAD ADDR OF MESSAGE
000A 0   D002              STO        MESS      STORE FOR SUBROUTINE
000B 20  23A17170          LIBF       TYPE0
000C 0   2000              DC         /2000     TYPE MESSAGE
000D 0   0000        MESS  DC         *-*
000E 20  23A17170    WAIT1 LIBF       TYPE0
000F 0   0000              DC         /0000     WAIT FOR CONSOLE
0010 0   70FD              B          WAIT1
0011 00  66000005          LDX   L2   5         SET CHARACTER
0013 01  6E000080          STX   L2   IO        COUNT FOR TYPE0
0015 0   705B              B          CONV
0016 0   1000        BTAB1 NOP                   TABLE OF ADDRESSES
0017 1   0061              DC         MESS4     FOR MESSAGES
0018 1   004F              DC         MESS3     FOR INPUT
0019 1   0041              DC         MESS2
001A 1   0030              DC         MESS1
001B 0   0014        HEAD  DC         20
001C     0028              DMES       'R'10SISAM FILE LOAD CALCULATIONS'R'E
0030 0   0010        MESS1 DC         16
0031     0020              DMES       '2RINDEX ENTRY LENGTH IN WORDS = 'E
0041 0   000D        MESS2 DC         13
0042     001A              DMES       'RRECORD LENGTH IN WORDS = 'E
004F 0   0011        MESS3 DC         17
0050     0022              DMES       'RNUMBER OF RECORDS TO BE LOADED = 'E
0061 0   000F        MESS4 DC         15
0062     001E              DMES       'RNUMBER OF OVERFLOW SECTORS = 'E
0071 20  23A17170    CONV  LIBF       TYPE0
0072 0   1000              DC         /1000     READ IN VALUE
0073 1   0080              DC         IO
0074 20  23A17170    WAIT  LIBF       TYPE0
0075 0   0000              DC         /0000     WAIT ON KEYBOARD
0076 0   70FD              B          WAIT
0077 0   C00E              LD         OUT1      MOVE PLUS SIGN TO
0078 0   D007              STO        IO        IO AREA FOR DCBIN
0079 20  040C2255          LIBF       DCBIN     CONVERT FROM IBM CARD CODE
007A 1   0080              DC         IO        TO BINARY VALUE IN ACC
007B 01  D500008C          STO   L1   VTAB1     STORE IN VALUE TABLE
007D 0   71FF              MDX    1   -1        DECREMENT COUNT
007E 0   7089              B          IN        IF COUNT IS NON-ZERO BRANCH
007F 0   7011              B          CAL
0080 0   0005        IO    DC         5         INPUT AREA FOR KEYBOARD
0081     0005        NO    BSS        5
0086 0   80A0        OUT1  DC         /80A0     CONVERSION AREA
0087     0005        OUT   BSS        5
008C 0   1000        VTAB1 NOP                   TABLE OF INPUT VALUES
008D 0   0000        OVRSC DC         *-*       NO. OF OVERFLOW SECTORS
008E 0   0000        RECRD DC         *-*       NO. OF RECORDS
008F 0   0000        LENGR DC         *-*       RECORD LENGTH
0090 0   0000        LENGI DC         *-*       INDEX ENTRY LENGTH
0091 0   C022        CAL   LD         SCTLG     DIVIDE SECTOR LENGTH BY
0092 0   1890              SRT        16        INDEX ENTRY LENGTH TO
0093 0   A8FC              D          LENGI     CALCULATE THE NUMBER OF
0094 0   D027              STO        IEPS      INDEX ENTRIES PER SECTOR
0095 0   C0F9              LD         LENGR     CREATE DIVISOR BY ADDING
0096 0   801B              A          TWO       TWO TO THE RECORD SIZE AND
0097 0   D01D              STO        WORK      STORING IN HOLD AREA
0098 0   C01B              LD         SCTLG     DIVIDE RECORD LENGTH+2
0099 0   1890              SRT        16        INTO THE SECTOR LENGTH TO
009A 0   A81A              D          WORK      CALCULATE THE NUMBER OF
009B 0   D01F              STO        RCDPS     RECORDS PER SECTOR
009C 0   C0F1              LD         RECRD     DIVIDE THE TOTAL NUMBER OF
009D 0   801D              A          RCDPS     RECORDS PLUS NO OF REC.
009E 0   9012              S          ONE       PER SECTOR MINUS ONE
009F 0   1890              SRT        16        BY THE NUMBER OF
00A0 0   A81A              D          RCDPS     RECORDS PER SECTOR TO FIND
00A1 0   D017              STO        NOPDS     NO OF PRIME DATA SECTORS
00A2 0   8010              A          SEVEN     ADD CONSTANT OF SEVEN
00A3 0   1803              SRA        3         DIVIDE BY 8 TO DETERMINE
00A4 0   D015              STO        NOPDC     NO OF PRIME DATA CYLINDERS
00A5 0   8016              A          IEPS      ADD INDEX ENTRIES/SECTOR
00A6 0   900A              S          ONE       MINUS ONE
00A7 0   1890              SRT        16        DIVIDE BY NO OF INDEX
00A8 0   A813              D          IEPS      ENTRIES/SECTOR TO FIND NO
00A9 0   D00E              STO        NOISC     OF INDEX SECTORS
00AA 0   800E              A          NOPDS     ADD NO OF INDEX SECTORS+
00AB 0   80E1              A          OVRSC     NO OF PRIME DATA SECTORS+
00AC 0   8004              A          ONE       NO OF OVERFLOW SECTORS +
00AD 0   D009              STO        TOTSC     1 FOR LABEL
00AE 00  65000006          LDX   L1   6         SET COUNT
00B0 0   7013              B          ROUT
00B1 0   0001        ONE   DC         1         CONSTANT OF ONE
00B2 0   0002        TWO   DC         2         CONSTANT OF TWO
```

```
00B3 0   0007        SEVEN DC        7          CONSTANT OF SEVEN
00B4 0   0140        SCTLG DC        320        NO WORDS PER SECTOR
00B5 0   0000        WORK  DC        *-*        TEMPORARY HOLD AREA
00B6 0   1000        VTAB2 NOP                  TABLE OF OUTPUT VALUES
00B7 0   0000        TOTSC DC        *-*        TOTAL NO OF SECTORS
00B8 0   0000        NOISC DC        *-*        NO OF INDEX SECTORS
00B9 0   0000        NOPDS DC        *-*        TOTAL NO OF PRIME DATA SCTR
00BA 0   0000        NOPDC DC        *-*        NO OF PRIME DATA CYLINDERS
00BB 0   0000        RCDPS DC        *-*        NO OF RECORDS PER SECTOR
00BC 0   0000        IEPS  DC        *-*        NO OF INDEX ENTRIES/SECTOR
00BD 0   1000        MTAB  NOP                  MESSAGE TABLE CONTAINS
00BE 1   00F5              DC        MS5P       ADDRESS IN THE MESSAGES
00BF 1   0107              DC        MS6P       WHERE THE VALUES ARE TO
00C0 1   011B              DC        MS7P       BE PLUGGED
00C1 1   0130              DC        MS8P
00C2 1   0144              DC        MS9P
00C3 1   0158              DC        MS10P
00C4 01  C50000BD  ROUT    LD    L1  MTAB       MOVE ADDR OF CORRECT
00C6 01  D40000CF          STO   L   ADDR       MSG TO CONVERT ROUTINE
00C8 01  C50000B6          LD    L1  VTAB2      LOAD VALUE TO CONVERT
00CA 20  02255103          LIBF      BINDC      CONVERT FROM BINARY TO
00CB 1   0086              DC        OUT1       IBM CARD CODE
00CC 20  292570D6          LIBF      ZIPCO      CONVERT FROM IBM CARD CODE
00CD 0   1100              DC        /1100      TO CONSOLE CODE AND
00CE 1   0087              DC        OUT        PLACE VALUE IN
00CF 0   0000        ADDR  DC        *-*        MESSAGE
00D0 0   0005              DC        5
00D1 30  085930D7          CALL      HOLCP
00D3 01  C50000DF          LD    L1  BTAB2      LOAD ADDR OF MESSAGE
00D5 0   D002              STO       MESSP      STORE ADDR FOR SUBROUTINE
00D6 20  23A17170          LIBF      TYPE0
00D7 0   2000              DC        /2000      TYPE OUTPUT MESSAGE
00D8 0   0000        MESSP DC        *-*
00D9 20  23A17170    WAIT3 LIBF      TYPE0
00DA 0   0000              DC        /0000      WAIT FOR CONSOLE
00DB 0   70FD              B         WAIT3
00DC 0   71FF        CHECK MDX   1   -1         DECREMENT COUNT
00DD 0   70E6              B         ROUT       IF COUNT NON-ZERO BRANCH
00DE 0   6038              EXIT
00DF 0   1000        BTAB2 NOP                  TABLE OF ADDRESSES
00E0 1   00E6              DC        MS5        FOR OUTPUT MESSAGES
00E1 1   00F8              DC        MS6
00E2 1   010A              DC        MS7
00E3 1   011E              DC        MS8
00E4 1   0133              DC        MS9
00E5 1   0147              DC        MS10
00E6 0   0011        MS5   DC        17
00E7     001C              DMES      'RTOTAL NUMBER OF SECTORS = '
00F5     0006        MS5P  DMES           'E
00F8 0   0011        MS6   DC        17
00F9     001C              DMES      'RNUMBER OF INDEX SECTORS = '
0107     0006        MS6P  DMES           'E
010A 0   0013        MS7   DC        19
010B     0021              DMES      'RNUMBER OF PRIME DATA SECTORS = '
011B     0005        MS7P  DMES           'E
011E 0   0014        MS8   DC        20
011F     0022              DMES      'RNUMBER OF PRIME DATA CYLINDERS = '
0130     0006        MS8P  DMES           'E
0133 0   0013        MS9   DC        19
0134     0021              DMES      'RNUMBER OF RECORDS PER SECTOR = '
0144     0005        MS9P  DMES           'E
0147 0   0013        MS10  DC        19
0148     0021              DMES      'RNUMBER OF INDEXES PER SECTOR = '
0158     0005        MS10P DMES           'E
015C     0000              END       START
```

APPENDIX K. BASIC DIFFERENCES BETWEEN 1130 DISK MONITOR SYSTEM, VERSIONS 1 AND 2

Many of the differences between Monitor 1 and Monitor 2 are listed below.

- Lowest allowable origin with:

	Version 1		Version 2	
	Dec.	Hex.	Dec.	Hex.
DISKZ	450	/01C2	510	/01FE
DISK0	610	/0262	690	/02B2
DISK1	880	/0370	690	/02B2
DISKN	1080	/0438	960	/03C0

NOTE: All version 2 disk subroutines provide multiple disk support and accommodate word counts exceeding 320. There is no DISK0 subroutine in version 2; a LIBF to DISK0 is interpreted as a LIBF to DISK1.

- Version 2 does not allow an initial ORG to an odd location in mainlines that require DISKZ. An ORG to an even location followed by a BSS or BES of an odd number of words equivalent to an ORG to an odd location.

- Version 2 may require more core than Version 1, especially FORTRAN core loads.

- Defective cylinders are taken into account in the Version 2 incremental seek and write immediate functions. In other words, it is not possible to seek to or write immediate on a defective cylinder.

- The object code produced by the FORTRAN compiler is slightly longer in Version 2 than Version 1.

- The calling sequence for DISKZ in Version 2 is different from Version 1.

- The LIST DECK, LIST DECK E, and PUNCH SYMBOL TABLE Assembler Options are not allowed with 1134 input.

- ILS02 and ILS04 are part of the Resident Monitor. (The user may write his own and store them in the User Area for use with user programs.)

- Some reprogramming is necessary for cases in which INT REQ key has some special meaning. Word $IREQ may still be used to point to a customer-written subroutine for servicing INT REQ, but the return from this subroutine must be compatible with the coding in ILS02 (see coding at $I200 Appendix H).

- The entire Resident Monitor, with the exception of $LINK, $EXIT, $IOCT, $PRET, and $IREQ, has been relocated. Certain parameters that were formerly in COMMA in Version 1 are in DCOM in Version 2.

- The Core Image header for Disk Core Image format (DCI) has been revised and relocated.

- The *FILE Assembler Control Record has been replaced by the pseudo-operation FILE. *FILE (not to be confused with the Supervisor Control Record *FILES) is not recognized in Version 2.

- On a DUP DUMP using the 1442-6 or -7, blank cards following the punched cards are not selected to stacker 2.

- Version 2 does not reset the NON-XEQ switch when a //ASM record is processed.

- Version 2 requires that all cartridges have a 4-character ID.

- In Version 2, a displacement in the operand field of a WAIT instruction is ignored, and does not remain in the generated opcode.

- There are certain diagnostics in Version 2 that are not in Version 1. Thus, some conditions are detected as errors in Version 2 that are not in Version 1.

- The Version 2 System Loader does not bypass the loading of ISSs for devices not defined on the REQ records. Such subroutines may, however, be deleted if desirable.

- Disk organization is different in the two versions.

- Version 2 requires 14 sectors more disk storage than Version 1, i.e., the address of Working Storage in Version 2 is 14 greater than in Version 1.

- Some FORTRAN programs that can be compiled by the DM1 compiler are too large to be compiled by the DM2 compiler.

- The ISS number for disk I/O has been changed from 4 (DM1) to 5 (DM2).

I-Field Type

This field type describes FORTRAN integer conversion; input is an integer field. The specification is

m-Iw.t(P)

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- I identifies the field type,
- w is the field length og the converted field (maximum of 14),
- t is the number of decimal positions reserved in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Example 1: The integer field /3A7E (14974 decimal) is converted using the field specification 15-I8.2 to the following RPG field.

Record Word:	8	9	10	11
Content:	F0F1	F4F9	F7F4	F0F0

Example 2 (truncation): The integer field of Example 1 is converted using the field specification 15-I6.2 to the following RPG field.

Record Word:	8	9	10
Content:	F4F9	F7F4	F0F0

Example 3 (packed format): The integer field of Example 1 is converted using the field specification 15-I8.2(P) to the following RPG field. The number is converted as in Example 1. The zone portions of each character are then removed and the digit portions are packed two per byte. The sign is added as a trailing hexadecimal digit (F=positive; D=negative).

Record Word:	8	9	10
Content:	0014	9740	0F40

Note: Since field length does not account for sign, incorrect alignment exists if packed mode is specified and field length is an even number. In order to align the data correctly, a leading zero is added to the field. This is true in all field types which accept packed mode conversion.

Example 4: The integer field /C582 (-14974 decimal) is converted using the field specification 15-I8.2 to the following RPG field.

Record Word:	8	9	10	11
Content:	F0F1	F4F9	F7F4	F0D0

J-Field Type

This field type describes two-word integer conversion; input is a two-word integer. The specification is

m-Jw.t(P)

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- J identifies the field type,
- w is the field length of the converted field (maximum of 14),
- t is the number of decimal positions reserved in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Example: The two-word integer field /7FFF /FFFF is converted using the field specification 7-J13.1(P) to the following RPG field.

Record Word:	4	5	6	7
Contnet:	0021	4748	3647	0F40

R-Field Type

This field type describes FORTRAN real-variable conversion. The specification is

m-Rw.t(P)

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- R identifies the field type,
- w is the field length of the converted field (maximum of 14),
- t is the number of decimal positions reserved in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Example 1: The standard precision real field /BC00 /0080 (-0.53125 decimal) is converted using

the field specification 25-R7.5(P) to the following RPG field.

Record Word:	13	14
Content:	0053	125D

Example 2: The real field of Example 1 is converted using the field specification 25-R7.5 to the following RPG field.

Record Word:	13	14	15	16
Content:	F0F0	F5F3	F1F2	D540

Example 3: The standard precision real field /7A12 /0097 (eight million decimal) is converted using the field specification 39-R7.0(P) to the following RPG field.

Record Word:	20	21
Content:	8000	000F

Example 4: If the field specification in Example 3 were 39-R7.2(P), then the resulting RPG field would be set to nines since the input field is too large to yield any signigicant digits in the RPG field.

Record Word:	20	21
Content:	9999	999F

If column 33 of the File Description card contained a W, a warning message would be printed when the above conversion took place.

Example 5: The extended precision real field /0047 /6250 /0000 (10^{-12} decimal) is converted using the field specification 17-R9.9 to the following RPG field.

Record Word:	9	10	11	12	13
Content:	F0F0	F0F0	F0F0	F0F0	F040

The RPG field is set to zeroes since the input field is too small to yield any significant digits in the RPG field. A number whose first signigicant digit is more than nine decimal places to the right of the decimal point cannot be expressed in RPG. If column 33 of the File Description card contained a W, a warning message would be printed when above conversion took place.

B-Field Type

This field type describes FORTRAN A-conversion for integer data and CSP A1 and A2 conversion. The specification is

m-Bw.n

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- B identifies the field type,
- w is the number of characters in the field (maximum of 255),
- n is the number of characters in each unit of the input field (n=1 or 2).

Example: The CSP field POSITIVE appears on a disk record in A2 format as follows.

Record Word:	n	n+1	n+2	n+3
Content:	E5C5	E3C9	E2C9	D7D6
	VE	TI	SI	PO

This field is converted using the field specification 21-B8.2 to the following RPG field.

Record Word:	11	12	13	14
Content:	D7D6	E2C9	E3C9	E5C5
	PO	SI	TI	VE

C-Field Type

This field type describes FORTRAN A-conversion for real data. The specification is

m-Cw.n

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- C identifies the field type,
- w is the number of characters in the field (maximum of 255),
- n is the number of characters in each unit (2 or 3 words) of the input field. For standard precision, n may range from 1 through 4; for extended precision, from 1 through 6.

Example: The FORTRAN field WASHINGTON, D.C. appears on a disk record in A4 format, extended precision, beginning at word 221 as follows.

Record Word:	210	211	212	213	214	215
Content:	4BC3	4B40	4040	D6D5	6BC4	4040
	.C	.		ON	,D	

Record Word:	216	217	218	219	220	221
Content:	C9D5	C7E3	4040	E6C1	E2C8	4040
	IN	GT		WA	SH	

This field is converted using the field specification

35-C15.4 to the following RPG field.

Record Word:	18	19	20	21	22	23
Content:	E6C1	E2C8	C9D5	C7E3	D6D5	6BC4
	WA	SH	IN	GT	ON	,D

Record Word:	24	25
Content:	4BC3	4B40
	.C	

D-Field Type

This field type describes CSP D1 conversion. The specification is

$$m\text{-}D1_1.\ j{=}1_2.k(P)$$

where
- m is the column of the RPG record in which the converted field begins (1 through 640)
- D identifies the field type,
- 1_1 is the length of the CSP field (maximum of 255),
- j is the number of decimal positions in the CSP field,
- 1_2 is the length of the RPG field (maximum of 14),
- k is the number of decimal positions in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Example: The CSP field +00946.88 appears on a disk record beginning at word 78 in D1 format as follows:

Record Word:	72	73	74	75	76	77	78
Content:	0008	0008	0006	0004	0009	0000	0000

This field is converted using the field specification 25-D7.2=6.3 to the following RPG field.

Record Word:	13	14	15
Content:	F9F4	F6F8	F8F0

E-Field Type

This field type describes CSP D4 conversion. The specification is

$$m\text{-}E1_1.\ j1_2.k(P)$$

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- E identifies the field type,
- 1_1 is the length of the CSP field (maximum of 255),
- j is the number of decimal positions in the CSP field,
- 1_2 is the length of the RPG field (maximum of 14),
- k is the number of decimal positions in the RPG field (maximum of 9),
- (P) is optional and is present only if the RPG field is to be packed.

Example: The CSP field - 00946.88 appears on a disk record beginning at word 103 in D4 format as follows.

Record Word:	101	102	103
Contents:	FFF7	68FF	0094

This field is converted using the field specification 25-E7.2=7.2(P) to the following RPG field.

Record Word:	13	14
Content:	0094	688D

F-Field Type

This field type describes CSP A3 conversion. It requires a 40 character translation table. The specification is

m-Fw

where
- m is the column of the RPG record in which the converted field begins (1 through 640),
- F identifies the field type,
- w is the number of characters in the field (not to exceed input record size in characters).

Example: Suppose that a 40 character translation table with W as the 23rd position relative to the last position of the A3 table (card column 40) H as the eight relative position, and Y as the 25th relative position is used to form the CSP field WHY in A3 format. This field is represented by the integer /1419 on a disk record. This integer is derived using the following formula.

$$I=1600\ (N_1-20)+40(N_2)+N_3$$

where N_1, N_2 and N_3 represent the positions in the table of the 1st, 2nd and 3rd characters respectively.

The above field is converted using the field specification 21-F4 to the following RPG field.

Record Word:	11	12
Content:	E6C8	E840
	WH	Y

X-Field Type

This field type allows fields on the input record to be bypassed. The specification is

Xw

where X identifies the field type,
w is the number of words to be bypassed (not to exceed input record size).

Example: The field specification used to bypass an array of 10 real numbers when standard precision has been specified (each real number is 2 words in length) is X20.

Composite Example: FORTRAN file FORFL containing 1,000 records, each record 10 words long, with standard precision and one word integers specified, is to be converted to the RPG file RPGFL containing records 40 characters in length.

One such FORTRAN record appears as follows.

Word:	1	2	3	4	5	6	7	8	9	10
Content:	3A7E	D64B	40D5	D540	D4C1	BC00	0080	03C8	C000	0083

If the File Description card, beginning in column 1, contains

Column	1	7	13	19	23	27	29	72
	FORFL	RPGFL	01000	010	040	S	1	D

and the Field Specification card, beginning in column 1, contains

Column:	1	72
	1-R3.0,5-I4.1,12-R7.5,21-B8.2,30-I8.2	S

and a / * card follows the control cards, the record described above will be converted and stored on disk as follows:

Word:	1	2	3	4	5	6	7	8	9	10
Content:	F0F0	D440	F9F6	F8F0	4040	40F0	F0F5	F3F1	F2D5	4040

Word:	11	12	13	14	15	16	17	18	19	20
Content:	D4C1	D540	40D5	D64B	40F0	F1F4	F9F7	F4F0	F040	4040

Absolute Address. An address that either should not be incremented or has already been incremented by a relocation factor.

Absolute Program. A program which, although stored in disk system format, has been written in such a way that it can be executed from only one core location.

Assembler Core Load. A core load that was built from a mainline written in Assembler language.

CALL Subprogram. A subprogram that must be referenced with a CALL statement. The type codes for subroutines in this category are 4 and 6.

CALL TV. The transfer vector through which CALL subroutines are entered during execution. See the section on the Core Load Builder for a description of this transfer vector.

Card Core Image Format (abbr. CDC). The format in which a program stored in disk core image format is dumped to cards.

Card Data Format (abbr. CDD). The format in which a data file is dumped to cards.

Card System Format (abbr. CDS). The format in which absolute and relocatable programs are punched into cards. In this format, columns 73-80 are used only to contain the card ID and sequence number.

CDC. (See Card Core Image Format.)

CDD. (See Card Data Format.)

CDS. (See Card System Format.)

Checksum. The two's complement of the logical sum of the record count (the position of the record within the program) and the data word(s). The logical sum is obtained by summing the data word(s) and the record number arithmetically, with the addition of one each time a carry occurs out of the high-order position of the Accumulator. The first record is record 1, not record 0.

This term (record number) should not be confused with the sequence number that appears in columns 73-80 in card formats.

CIB. (See Core Image Buffer.)

Cold Start Card. The card that contains the coding necessary for initial program loading (IPL), that is, fetching the Cord Start Program.

Cold Start Program. The disk-resident program that initializes the Monitor system by reading the Resident Monitor into core from the disk.

COMMA. (See Core Communications Area.)

Comment. The text contained on a Monitor control record with an asterisk in column 4, an Assembler language source record with an asterisk in column 21, a FORTRAN source record with a C in column 1, or an RPG specification with an asterisk in column 7.

Control Record. One of the records (card or paper tape) that direct the activities of the Monitor system. For example, the DUP Monitor control record directs the Monitor to initialize DUP, the DUMPLET DUP control record directs DUP to initialize the DUMPLET program; the EXTENDED PRECISION FORTRAN control record directs the Compiler to allot three words instead of two for the storage of variables.

Core Communications Area (abbr. COMMA). The part of core which is reserved for work areas and parameters that are required by the Monitor programs. In general a parameter is found in COMMA if it is required by two or more Monitor programs and is required to load a program stored in disk core image format. Otherwise the parameter is found in DCOM. COMMA is initialized by the Supervisor during the processing of a JOB record.

Core Image Buffer (abbr. CIB). The buffer on which most of the first 4K of core are saved while a core load is being built. It is also used to save any part of COMMON defined below location 4096 during a link-to-link transfer of control. See the section on the Core Load Builder for a description of the CIB and its use.

Core Image Header Record. A part of a core image program including such parameters as the word count of the core load, the ITV, and the setting for index register 3.

Core Image Program. A mainline that has been converted, along with all of its required subroutines, to disk core image format. Included in the core image program are any LOCALs and/or SOCALs that are required. This term should not be confused with "core

load", which refers to only that part of a core image program that is read into core just prior to execution.

Core Load. A mainline, its required subroutines, and its interrupt, CALL, and LIBF transfer vectors. This term should not be confused with "core image program".

CSF Block. A group of not more than 51 data words of a program in card system format. In this format, the first six data words of every CSF block are indicator words. These six words are always present, even though all six are not needed. A CSF block is equivalent to words 4-54 of the CSF module (Data card) of which it is a part.

CSF Module. A group of words consisting of a data header and CSF blocks for a program in card system format. A CSF module is equivalent to a Data card in card system format. A new CSF module is created for every data break. A data break occurs (1) whenever there is an ORG, BSS, BES, or DSA statement, (2) whenever a new Data card is required to store the words comprising a program, and (3) at the end of the program.

Data Break. (See DSF Module.)

Data File. An area in either the User Area or the Fixed Area in which data is stored. "Data file" may also refer to the data itself.

Data Header. The first pair of words in a module for a program in disk system format. The first word contains the loading address of the module; the second the total number of words contained in the module. The data header for the last module contains the effective program length, followed by a word count of zero.

DCI. (See Disk Core Image Format.)

DCOM. (See Disk Communications Area.)

DDF. (See Disk Data Format.)

DEFINE FILE Table. The table which appears at the beginning of every mainline that refers to defined files. There is one 7-word entry for each file that has been defined.

Disk Block. One sixteenth of a disk sector, that is, 20 disk words. The disk block is the smallest distinguishable increment for programs stored in disk system format. Thus, the Monitor system permits packing of disk system format programs at smaller intervals than the hardware would otherwise allow.

Disk Communications Area (abbr. DCOM). The disk sector that contains the work areas and parameters for the Monitor programs.

Disk Core Image Format (abbr. DCI). The format in which core image programs are stored on the disk prior to execution.

Disk Data Format (abbr. DDF). The format in which a data file is stored in either the User Area or the Fixed Area.

Disk System Format (abbr. DSF). The format in which mainlines and subprograms are stored on the disk as separate entities. It is not possible to execute a program in disk system format; it must first be converted to disk core image format as a result of either an XEQ Monitor control record or a STORECI DUP control record.

Disk System Format Program. A program that is stored in disk system format. It is sometimes called a DSF program.

DSF. (See Disk System Format.)

DSF Block. A group of not more than nine data words of a program in disk system format. In this format, the first data word of every DSF block is an indicator word. Normally every DSF block in a DSF module consists of nine data words, including an indicator word; but if the DSF module contains a number of data words that is not a multiple of nine, then the next-to-last DSF block contains less than nine data words.

DSF Module. A group of words consisting of a data header and DSF blocks for a program in disk system format. A new DSF module is created for every data break. A data break occurs (1) whenever there is an ORG, BSS, BES, or DSA statement, (2) whenever a new sector is required to store the words comprising a program, and (3) at the end of the program.

Effective Program Length. The terminal address appearing in a relocatable program. For example, in Assembler language programs, this address is the last value taken on by the Location Assignment Counter and appears as the address assigned to the END statement.

Entry Point. Either (1) the symbolic address (name) of a place at which a program is entered, (2) the absolute core address at which a program is to be entered, or (3) the address, relative to the address of the first word of the subprogram, at which it is to be entered.

Execution. The execution of the program specified on an XEQ Monitor control record and any subsequent links executed via CALL LINK statements. The execution is complete when a CALL EXIT is executed.

Fetching. The process of reading something into core storage, usually from disk.

Fixed Area (abbr. FX). The area on disk in which core image programs and data files are stored if it is desired that they always occupy the same sectors. No programs in disk system format may be stored in this area. No packing ever occurs in the Fixed Area.

FLET. (See LET/FLET.)

FORTRAN Core Load. A core load that was built from a mainline written in the FORTRAN language.

Function. A subprogram that evaluates a mathematical relationship between a number of variables. In FORTRAN, a FUNCTION is a subprogram that is restricted to a single value for the result. This type of subprogram is called by direct reference.

FX. (See Fixed Area.)

IBM Area. That part of disk storage that is occupied by DCOM, the CIB, and the Monitor programs. This area is also known as the System Area.

IBT. (See ILS Branch Table.)

ILS. (See Interrupt Level Subroutine.)

ILS Branch Table (abbr. IBT.) A table consisting of the addresses of the interrupt entry points for each ISS used for an interrupt level. An IBT is required by the ILS for an interrupt level with which more than one device is associated.

In-core Subprogram. A subprogram that remains in core storage during the entire execution of the core load of which it is a part. ILSs are always in-core subprograms, whereas LOCALs and SOCALs never are.

Indicator Word. The first word of a DSF block indicating which of the following data words should be incremented (relocated) when relocating a program in disk system format. This word also indicates which words are LIBF, CALL, and DSA names. Programs in disk system format all contain indicator words. Each pair of bits in the indicator word is associated with one of the following data words -- the first pair with the first data word following the indicator word, etc.

Initial Program Load. The action that occurs when the PROGRAM LOAD key is pressed. One record is read into core, starting at location zero, from the input hardware device that is physically wired to perform this function. The record read, usually a loader, then instructs the system as to the next action to be performed, e.g., load more records.

Interrupt Level Subroutine (abbr. ILS). A subroutine that analyzes all interrupts on a given level; that is, it determines which device on a given level caused the interrupt and branches to a servicing subroutine (ISS) for the processing of that interrupt.

Interrupt Service Subroutine (abbr. ISS). A subroutine that 1) manipulates a given I/O device and 2) services all interrupts for that devicer after they have been detected by an ILS.

Interrupt Transfer Vector (abbr. ITV). The contents of words 8-13, which are the second words of the automatic BSI instructions which occur with each interrupt. In other words, if an interrupt occurs on level zero and if core location eight contains 500, an automatic BSI to core location 500 occurs. Similarly, interrupts on levels 1-5 cause BSIs to the contents of core locations 9-13, respectively.

IOAR Header. The word(s) required by an I/O device subroutine (ISS). They must be the first or the first and second words of the I/O buffer.

IPL. (See Initial Program Load.)

ISS. (See Interrupt Service Subroutine.)

ISS Counter. A counter in COMMA (word $IOCT) that is incremented by 1 upon the initiation of every I/O operation and decremented by 1 upon receipt of an I/O operation complete interrupt.

ITV. (See Interrupt Transfer Vector.)

Job. A group of tasks (subjobs) that are to be performed by the Monitor system and which are interdependent; that is, the successful execution of any given subjob (following the first) depends upon the successful execution of at least one of those that precede it.

LAC. (See Location Assignment Counter.)

LET/FLET (the Location Equivalence Table for the User Area/ the Location Equivalence Table for the Fixed Area). The disk-resident table through which the disk addresses of programs and data files stored

in the User/Fixed Area may be found. On a system cartridge, LET occupies the cylinder preceding the User Area. If a Fixed Area has been defined, FLET occupies the cylinder preceding it; otherwise, there is no FLET.

LIBF Subroutine. A subprogram that must be referenced with an LIBF statement. The type codes for subroutines in this category are 3 and 5.

LIBF TV. The transfer vector through which LIBF subprograms are entered at execution time. See the section on the Core Load Builder for a description of this transfer vector.

Link. A link is a core image program that is read into core for execution as a result of the execution of a CALL LINK statement.

Loading Address. The address at which a mainline, subprogram, core load, or DSF module is to begin. For mainlines and DSF modules, the loading address is either absolute or relative. For subprograms, it is always relative, whereas, for core loads, it is always absolute.

Load-On-Call (abbr. LOCAL) Subroutine. A subprogram in a core image program that is not an in-core subprogram. It is read from the disk into a special overlay area in core only when it is called during execution. LOCALs, which are specified for any given execution by the user, are a means of gaining core storage at the expense of execution time. The Core Load Builder constructs the LOCALs and all linkages to and from them.

Load-Although-Not-Called (abbr. NOCAL) Subprogram. A subprogram that is to be included in a core image program although it is never referenced in that core image program by an LIBF or CALL statement. Debugging aids such as a trace or a dump fall into this category.

LOCAL. (See Load-On-Call Subroutine.)

Location Assignment Counter. A counter maintained in the Assembler for assigning addresses to the instructions it assembles. A similar counter is maintained in the Core Load Builder for loading purposes.

Long Instruction. An instruction that occupies two core storage locations.

Low COMMON. Words 896 - 1215 if DISKZ is in core, words 1216 - 1535 if DISK1 is in core, or words 1536 - 1855 is DISKN is in core. This area exists even if there is no COMMON.

Mainline. The program about which a core image program is built. The mainline is normally the program in control. It calls subprograms to perform various functions.

Master Cartridge. The cartridge residing on logical drive zero. The master cartridge must be a system cartridge.

Modified EBCDIC Code. A six-bit code used internally by the Monitor programs. In converting from EBCDIC to Modified EBCDIC, the leftmost two-bits are dropped. (See Name Code.)

Monitor. A synonym for the entire 1130 Disk Monitor System, Version 2, which is also known as the Monitor system or the Disk Monitor.

Monitor Control Record. (See Control Record.)

Monitor Program. One of the following parts of the Monitor system: Supervisor (SUP), Core Image Loader (CIL), Core Load Builder (CLB), Disk Utility Program (DUP), Assembler (ASM), FORTRAN Compiler (FOR), or RPG Compiler (RPG).

Name Code. The format in which the names of subprograms, entry points, labels, etc., are stored for use in the Monitor programs. The name consists of five characters, terminal blanks being added if necessary to make five characters. Each character is in Modified EBCDIC code, and the entire 30-bit representation is right-justified in two 16-bit words. The leftmost two bits are used for various purposes by the Monitor.

Naturally Relocatable Program. A program that may be executed from any core storage location without first being relocated. The only absolute addresses in such a program refer to parts of the Resident Monitor, which, of course, are fixed.

NOCAL. (See Load-Although-Not-Called Subprogram.)

Non-system Cartridge. A cartridge that does not contain the Monitor programs, although it does contain DCOM, LET, etc. A non-system cartridge may be used only as a satellite cartridge.

NOP. An acronym used to denote the instruction, No operation.

Object Program. The output from either the Assembler, FORTRAN Compiler, or the RPG Compiler.

Packing. The process of storing programs in the User Area to the nearest disk block, thus reducing the average wasted disk space from 160 disk words/program to 10 disk words/program.

Padding. Areas in the User/Fixed Area required to permit core image programs and data files to start on a sector boundary. The length of the padding, which is reflected in LET/FLET with a dummy entry, is from 1 to 15 disk blocks.

Principal I/O Device. The device used for stacked job input to the Monitor system. The 2501/1442, 1442/1442, or 1134/1055 may be assigned as the principal I/O device. The Keyboard may be assigned temporarily as the principal input device (see // TYP under Monitor Control Records). The System Loader considers the fastest device defined on the REQ records to be the principal I/O device.

Principal Print Device. The device used by the Monitor system for printing system messages. Either the 1403, 1132, or Console Printer may be assigned as the principal print device. The System Loader considers the fastest print device defined on the REQ records to be the principal print device.

Program. The highest level in the hierarchy describing various types of code. Subprograms and mainlines are subsets of this set.

Program Header Record. The part of a program stored in disk system format that precedes the first DSF module. Its contents vary with the type of program with which it is associated. It contains the information necessary to identify the program, to describe its properties, and to convert it from disk system format to disk core image format.

Quintuples. Five-word tables in COMMA/DCOM that contain cartridge-related parameters. There is one table for each cartridge on the system. These tables are updated during JOB processing or by a user callable subprogram (SYSUP) if cartridges are changed during a job.

Relocatable Program. A program that can be executed from any core location. Such a program is stored on the disk in disk system format. It is relocated by the Core Load Builder.

Relocation. The process of adding a relocation factor to address constants and to those long instructions whose second words are not (1) invariant quantities, (2) absolute core addresses, or (3) symbols defined as absolute core addresses. The relocation factor for any program is the absolute core address at which the first word of that program is found.

Relocation Indicator. The second bit in a pair of bits in an indicator word. If the data word with which this bit is associated is not an LIBF, CALL, or DSA name, then it indicates whether or not to relocate the data word. If the relocation indicator is set to 1, the word is to be relocated. Pairs of relocation indicators indicate LIBF, CALL, or DSA names. The combinations are 1000, 1100, and 1101, respectively.

Remark. An explanation of the use or function of a statement or statements. A remark is a part of a statement, whereas a comment is a separate statement.

Resident Image. The mirror-image of the Resident Monitor minus the disk I/O subroutine. It resides on disk and is read into core by the Cold Start Program.

Resident Monitor. The area required in core by the Monitor system for its operation. This area is generally unavailable to the user for his own use. The Resident Monitor consists of COMMA, the Skeleton Supervisor, and one of the disk I/O subroutines, nominally DISKZ.

RPG Core Load. A core load that was built from a mainline which was generated from an RPG language program.

Satellite Cartridge. A cartridge residing on a drive other than logical drive zero. A satellite cartridge can be either a system or a non-system cartridge.

Short Instruction. An instruction that occupies only one core storage location.

Skeleton Supervisor. The part of the Supervisor that is always in core and that is, essentially, the logic necessary to process CALL DUMP, CALL EXIT, and CALL LINK statements. Certain traps are also considered to be part of the Skeleton Supervisor.

SOCAL. (See System Overlay to be Loaded-On-Call.)

Subjob. A Monitor operation to be performed during a job. Each subjob is initiated by a Monitor control record such as ASM or XEQ. It may also be initiated by a CALL LINK.

Subprogram. A synonym used mainly in FORTRAN for both FUNCTIONs and SUBROUTINEs. This term

is equivalent to subroutine when subroutine is used in its broadest sense.

Subroutine. A subset of the set "program". In FORTRAN, a SUBROUTINE is a type of subprogram that is not restricted to a single value for the result and that is called with a CALL statement.

Supervisor Control Record Area (abbr. SCRA). The cylinder in which the Supervisor control records are written. The first two sectors are reserved for LOCAL control records, the next two for NOCAL control records and the next two for FILES control records. See the Supervisor section for the formats of these records.

System Area. (See IBM Area.)

System Cartridge. A cartridge that contains the Monitor programs. A system cartridge may be used as either a master or a satellite cartridge.

System Overlay to be Loaded-On-Call (abbr. SOCAL). One of two or three overlays automatically prepared by the Core Load Builder under certain conditions when a core load is too large to fit into core storage. See the section on the Core Load Builder for an explanation.

System Working Storage. The Working Storage area to be used during a job by the Monitor programs, not user programs. The cartridge to be used for System Working Storage is defined on the JOB record. System Working Storage need not be on the master cartridge.

Transfer Vector (abbr. TV). A collection of both the LIBF TV and the CALL TV.

TV. (See Transfer Vector.)

UA. (See User Area.)

User Area (abbr. UA). The area on the disk in which all programs in disk system format are found. Core image programs and data files may also be stored in this area. All IBM-supplied programs are found here. This area occupies as many sectors as are required to store the programs and files residing there.

User Programs. Mainlines, subprograms, or core loads that have been written by the user and stored in the User/Fixed Area.

Working Storage (abbr. WS). The area on disk immediately following the last sector occupied by the User Area. This is the only one of the three major divisions of disk storage (IBM Area, User/Fixed Area, Working Storage) that does not begin at a cylinder boundary.

WS. (See Working Storage.)

IBM 1130 Printed in U. S. A. C26-3717-4

IBM

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENT FORM

IBM 1130 Disk Monitor System, Version 2
Programming and Operator's Guide

Form C26-1717-4

- How did you use this publication?

  As a reference source ..... ☐
  As a classroom text ..... ☐
  As a self-study text ..... ☐

- Based on your own experience, rate this publication . . .

As a reference source:	..... Very Good	..... Good	..... Fair	..... Poor	..... Very Poor
As a text:	..... Very Good	..... Good	..... Fair	..... Poor	..... Very Poor

- What is your occupation? .....

- We would appreciate your other comments; please give specific page and line references where appropriate. If you wish a reply, be sure to include your name and address.

- **Thank you for your cooperation. No postage necessary if mailed in the U.S.A.**

C26-3717-4

YOUR COMMENTS, PLEASE . . .

This SRL manual is part of a library that serves as a reference source for systems analysts, programmers and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Fold Fold

FIRST CLASS
PERMIT NO. 1359
WHITE PLAINS, N. Y.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
112 East Post Road
White Plains, N. Y. 10601

Attention: Department 813 L

Fold Fold

IBM

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

CUT ALONG THIS LINE

IBM 1130 Printed in U. S. A. C26-3717-4

a Program Library
1 Japan, Ltd.
tems Engineering Dept.
1 Chome Nagata-cho
yoda-ku
:yo, Japan

Canadian Program Library
IBM Canada Ltd.
1150 Eglinton Ave. E.
Don Mills 402, Ont.
Canada

European Program Library
IBM France
23, Allée-Maillasson
F.92-Boulogne-Billancourt
France

Société Anonyme Au Capital de
347. 424. 000 F-R.C.
(Seine 55B-11 846)

Program Information Dept.
IBM Corporation
40 Saw Mill River Road
Hawthorne, New York 10532
United States

South American
Program Library
IBM do Brasil, Ltda.
Avenida Presidente
Vargas 642, 4 Andar
Caixa Postal 1830-ZC-00
Rio de Janeiro, Brazil

South Pacific
Program Library
IBM Australia, Ltd.
Box 3318 G.P.O.
Sydney, N.S.W.
Australia

SEPTEMBER 1969

Memorandum To: Users of IBM 1130 Disk Monitor Programming System, Card Input, 1130-OS-005

Subject: Error in Version 2, Modification Level 6

Four (4) REQ cards were erroneously included in the System Loader deck following the SCON card (ID/Sequence Number G0000001). Replace these with REQ cards which reflect your configuration.

The Load Mode Control Card which follows card DP105063 contains an R in column 14. If you are using RPG, this column must be changed to a blank before performing the reload to install modification level 6.

We hope the error has not caused any inconvenience.

cc: SE Managers
FE Managers