SPERRY UNIVAC

# PUBLICATIONS RELEASE

**9400/9480 Systems**

**Processors**

**Programmer Reference**

**UP-8080**

**This SPERRY UNIVAC™ 9400/9480 Systems Library Memo announces the release and availability of "SPERRY UNIVAC 9400/9480 Systems Processors Programmer Reference", UP-8080. This is a Standard Library Item.**

This manual provides hardware information to familiarize system programmers with the overall operation of the SPERRY UNIVAC 9400 and 9480 System processors. Information provided includes overall description of the processor, input/output channels, main storage, and system console.

Detailed information includes data and instruction formats, fixed-point and variable length decimal arithmetic instructions, interrupts, the program status word, and timer control word. I/O channel information includes multiplexer and selector channels, functional interfaces, start I/O instruction, initial loading, and I/O priorities and timing. Main storage characteristics for each system are described in detail, along with coverage of the operation of the system console and associated printer.

This manual supersedes Section 2 of the "UNIVAC 9400 System Assembler/Central Processor Unit Programmer Reference " UP-7600 (current version).

Additional copies may be ordered by your local Sperry Univac Representative.

| | | |
|---|---|---|
| Mailing Lists 217, 630 and 692 | Mailing Lists 60 and 61 (Covers and 95 pages) | Library Memo for UP-8080 |

RELEASE DATE

July, 1974

UD1-0251 Rev. 0/73

# SPERRY UNIVAC 9400/9480 Systems Processors

Programmer Reference



SPERRY UNIVAC COMPUTER SYSTEMS

This document contains the latest information available at the time of publication. However, Sperry Univac reserves the right to modify or revise its contents. To ensure that you have the most recent information, contact your local Sperry Univac representative.

Sperry Univac is a division of Sperry Rand Corporation.

FASTRAND, MATED-FILM, PAGEWRITER, SPERRY UNIVAC, UNISCOPE, UNISERVO, and UNIVAC are trademarks of the Sperry Rand Corporation.

©1974 — SPERRY RAND CORPORATION

PRINTED IN U.S.A.

# PAGE STATUS SUMMARY

**ISSUE: UP-8080**

| Part/Section | Page Number | Update Level |
|---|---|---|
| Cover/Disclaimer | | |
| PSS | 1 | |
| Contents | 1 thru 5 | |
| 1 | 1 thru 10 | |
| 2 | 1 thru 22 | |
| 3 | 1 thru 22 | |
| 4 | 1 thru 8 | |
| 5 | 1 thru 14 | |
| Appendix A | 1, 2 | |
| Appendix B | 1, 2 | |
| Index | 1 thru 7 | |
| User Comment Sheet | | |

| Part/Section | Page Number | Update Level |
|---|---|---|

| Part/Section | Page Number | Update Level |
|---|---|---|

*All the technical changes are denoted by an arrow (➔) in the margin. A downward pointing arrow ( ↓ ) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow ( ↑ ) is found. A horizontal arrow (➔) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.*

# Contents

# 1. Introduction

## 1.1. GENERAL

The information contained in this manual is designed to familiarize system programmers with the overall operation of the UNIVAC 9400 and UNIVAC 9480 Systems Processors (Figure 1–1) and their relation to system programming.

The manual is organized in five sections, each containing a description of each system hardware component, plus appendixes, as follows:

- Section 1

  This section contains an overall description of the processor, input/output (I/O) channels, main storage, and system console, in addition to the possible system configurations, minimum and maximum, with expansion features.

- Section 2

  Data and instruction formats, fixed-point and variable-length decimal arithmetic instructions, the program status word (PSW), interrupts, and the purpose and use of the timer control word (TCW) are described in this section.

- Section 3

  The I/O sections of the processors, including the multiplexer channel, functional interfaces, optional selector channels, and the optional communications adapter are described in this section, as well as the start-I/O instruction, initial loading, I/O priorities, and timing.

- Section 4

  Main storage characteristics for each system are described in this section.

- Section 5

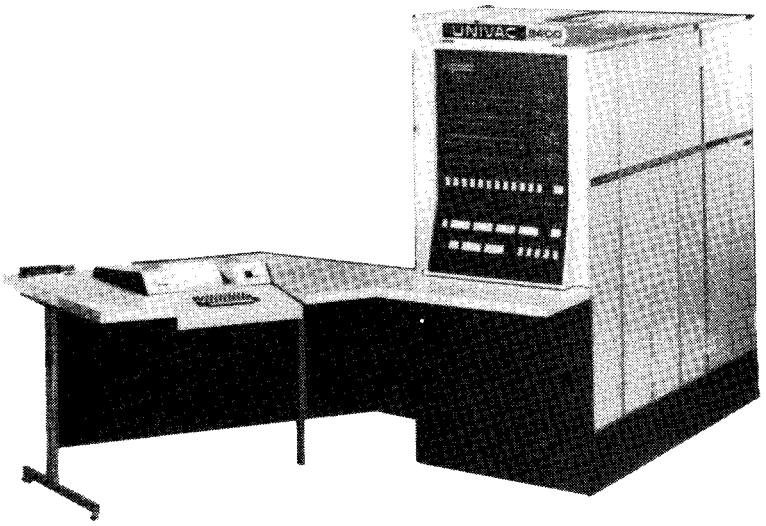
  This section contains descriptions of the system consoles, including operation of printers.

- Appendix A

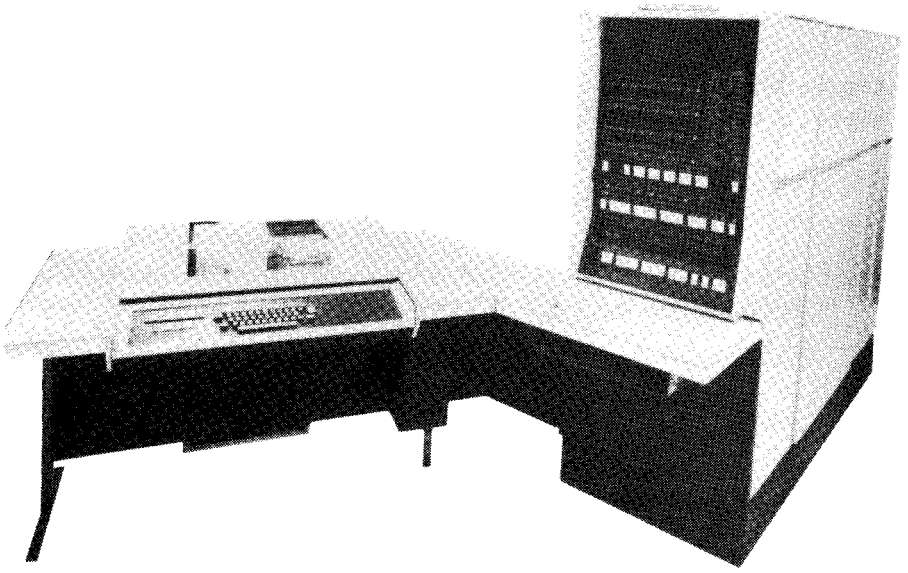
  Appendix A contains table listings (alphabetical by name) of instructions, including hexadecimal opcodes, and execution times.

- Appendix B

  Appendix B is a glossary of acronyms.



a. Serial numbers between 100 and 204



b. Serial numbers 205 and above

*Figure 1–1. UNIVAC 9400/9480 Systems Processors*

## 1.2. FUNCTIONAL DESCRIPTION

Inasmuch as the UNIVAC 9400 and 9480 Systems Processors are very nearly identical, information presented in this manual is for both processors, with only the differences between the two processors being specified separately.

### 1.2.1. Processor

The processor consists mainly of a control section, arithmetic section, and I/O channels. The control section controls the sequence in which instructions are executed, and interprets and controls the execution of each instruction. The arithmetic section processes binary and decimal integers and logical information. Also, it performs data comparisons, shifting, and single or double indexing of operand addresses. It is shared with the multiplexer input/output channel. (Each selector channel contains its own arithmetic section.) All binary operations are performed in the twos complement form. This is accomplished by taking the ones complement of the number and adding one.

| | |
|---|---|
| 11000110 | original number |
| 00111001 | ones complement |
| 1 | add one |
| 00111010 | twos complement |

The I/O channels consist of one multiplexer channel and up to two selector channels which permit peripheral subsystems to be connected to main storage and utilize instructions from the processor.

#### 1.2.1.1. Registers

The processor has access to two sets of general registers, 16 supervisor and 16 problem, located in low-order main storage. These registers are used to eliminate storing of register content when the supervisor intercepts the program to process interrupts. Register storage is required only when switching between problem programs.

#### 1.2.1.2. Status

Overall processor status is determined by the contents of a 64-bit program status word (PSW) register. All interrupts cause the contents of the PSW register to be transferred to storage allocated for that class of interrupt. The area of storage for each interrupt type is called the old PSW. The PSW register is then loaded with new corresponding PSW unique to the type of interrupt. This information is located in the storage area assigned to new PSWs. An old PSW area and a new PSW area are available for each class of interrupt.

#### 1.2.1.3. Interrupts

Seven classes of interrupts are available in the processor. In all cases, each interrupt causes the contents of the PSW register to be stored in the old PSW area and the new PSW to be loaded into the PSW register. The seven classes of interrupts are:

- Supervisor call

  This interrupt is used for communicating between the problem program and the supervisor program. The interrupt is caused by the execution of the supervisor-call instruction (SVC).

- Program exception

  This interrupt is caused by illegal operation codes, privileged instruction execution in problem mode, storage protection violation, addressing and specification exceptions, binary and decimal overflow, and divide check.

- Timer

  This interrupt is caused by the interrupt count field of the timer word being decremented from 1 to 0.

- Selector channel 1

  This interrupt is caused by an I/O interrupt on selector channel 1.

- Selector channel 2

  This interrupt is caused by an I/O interrupt on selector channel 2.

- Multiplexer channel (shared subchannel)

  This interrupt is caused by an I/O interrupt on the shared subchannels of the multiplexer channels.

- Multiplexer channel (nonshared subchannel)

  This interrupt is caused by an I/O interrupt on one of the nonshared multiplexer subchannels.

### 1.2.1.4. Timer

The timer is standard equipment and provides variable-length, periodic interrupts and an interval timer. The timer is used to measure elapsed time over 2-minute (short) intervals and maintains the time of day by providing the necessary data for logging job functions. The timer is decremented in the interrupt count field and incremented in the running time counter (interval timer) field of the timer word each millisecond. The range of the interrupt count is 1 to $(2^{12}-1)$ milliseconds (4096 – 1). The range of the interval timer is 0 to $(2^{17}-1)$ milliseconds (131,072 – 1).

### 1.2.1.5. Instruction Types

The instructions are:

- register-to-indexed-storage and indexed-storage-to-register (RX);
- storage-to-immediate-operand (SI);
- register-to-register (RR);
- storage-to-storage (SS); and
- register-to-storage and storage-to-register (RS).

The basic RX execution (add) time is 6.0 microseconds. For exact execution times for all instructions, see Appendix A.

## 1.2.2. I/O Section

The I/O section consists of the I/O channels and the I/O interface connecting the channels with the subsystem controllers. This section initiates, directs, and monitors the transfer of data between main storage and the peripheral subsystems. One multiplexer channel is provided as part of both the 9400 and 9480 systems. Up to two selector channels may be added as optional features to the 9400 system, while the 9480 system is provided with one selector channel as part of its system, with the capability of a second selector channel being added as an optional feature.

### 1.2.2.1. Multiplexer Channel

Data transfer for the multiplexer channel is a maximum of 85k bytes per second for restricted applications, the transfer mode being a single byte.

The channel is capable of operating with eight shared subchannels and 128 nonshared subchannels, the 128 nonshared subchannels being part of the communications adapter optional feature. The adapter enables:

- special-character recognition;
- odd or even parity generation and checking;
- data chaining; and
- status tabling.

### 1.2.2.2. Selector Channels

Data transfer for the selector channel is a maximum of 333k bytes per second. This data rate can be sustained on one selector channel so long as no interference is encountered from a second selector channel.

The channel is capable of operating with eight subchannels (not simultaneously) in burst mode with command chaining.

## 1.2.3. Main Storage

Main storage access is a 600-nanosecond cycle time: using plated-wire type storage modules expandable from 32K bytes to 256K bytes (UNIVAC 9400 System); using semiconductor type storage modules expandable from 64K bytes to 256K bytes (UNIVAC 9480 System). Main storage contains a ninth bit associated with each byte for parity. Locations are consecutively numbered in main storage, with each address specifying one byte of information. Bytes may be accessed separately or in groups. The address of a group of bytes is the leftmost byte of the group. Main storage is specified in terms of K bytes of capacity, the K term being defined as $2^{10}$ or 1024 bytes.

### 1.2.3.1. Parity Checking

The parity bit is checked as data, is read out of storage, and regenerated when data is altered as a result of processing or when data is introduced into the processor from peripheral devices. Parity also is checked and generated at the I/O channel interface to a bus to verify the transmission between the channel and device controls. Odd parity is the correct parity; even parity (parity error) causes the processor to stop.

## 1.2.4. System Console

The system console consists of a printer, keyboard, and associated I/O interface logic. The keyboard and printer, which are housed in a desk-like cabinet attached to the processor, derive power from the processor.

The system console control is housed in the processor cabinet.

The system console keyboard generates 8-bit EBCDIC characters, each being transferred to the control with an uppercase or lowercase strobe pulse.

When the control holds an active read command, the keyboard transmits characters to the printer and channel in an online mode, or to the printer only in an offline mode. Once the control receives a character, another character is not accepted until the first is printed.

Two types of printers are provided for the system console. For processors with serial numbers between 100 and 204, a 10-character-per-second print rate is provided. For processors with serial numbers 205 and above, a printer with a 25-character-per-second print rate is provided. The control required to translate data bytes coded in EBCDIC to the interface signals required by the printer is provided by printer control located in the processor.

## 1.3. CONFIGURATIONS

The minimum and expanded configurations of the systems processors are illustrated in Figures 1–2 and 1-. The minimum configuration for each system processor is defined by the solid-line boxes. The dashed-line boxes define the possible expanded configuration for each system. All expansions for each system are optional features. The features available for each system are listed, and briefly described, in Table 1–1.

### 1.3.1. Minimum Configurations

- Processor
  - UNIVAC 9400 System Processor
  - UNIVAC 9480 System Processor
- Internal main storage
  - UNIVAC 9400 System Processor: minimum of 32K bytes of plated-wire main storage
  - UNIVAC 9480 System Processor: minimum of 64K bytes of semiconductor main storage
- Multiplexer channel
- Selector channel 1
- System console, keyboard, and printer

### 1.3.2. Expanded Configurations

The optional expansion features are:

- Selector channel 2
- Communications adapter
- Storage protection
- Main storage
  - UNIVAC 9400 System Processor: maximum of two 16K-byte, two 32K-byte, and two 64K-byte storage expansion modules, resulting in a maximum of 256K bytes of main storage.
  - UNIVAC 9480 System Processor: maximum of two 32K-byte and two 64K-byte storage expansion modules, resulting in a maximum of 256K bytes of main storage.

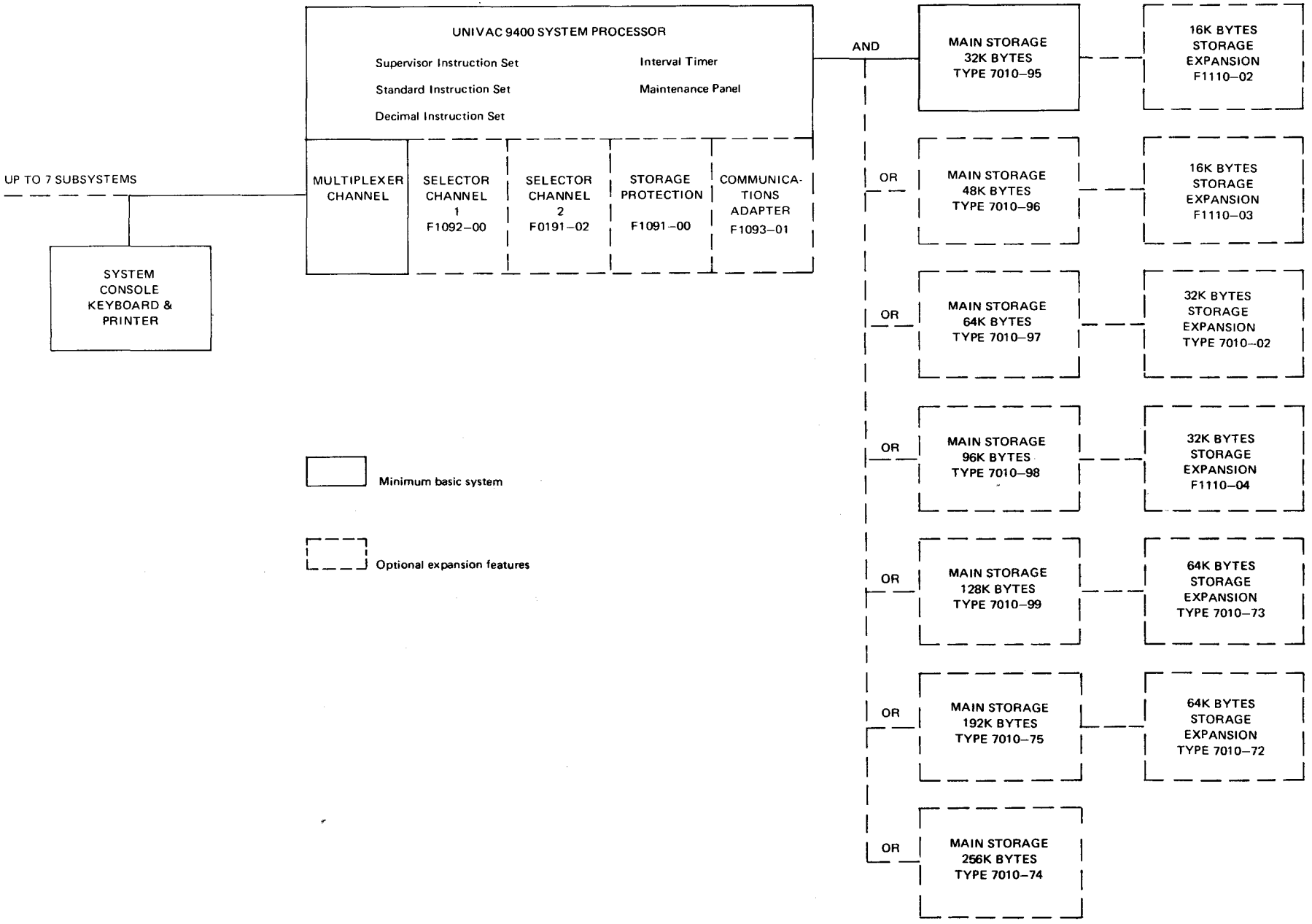



*Figure 1–2. UNIVAC 9400 System Processor Configuration*

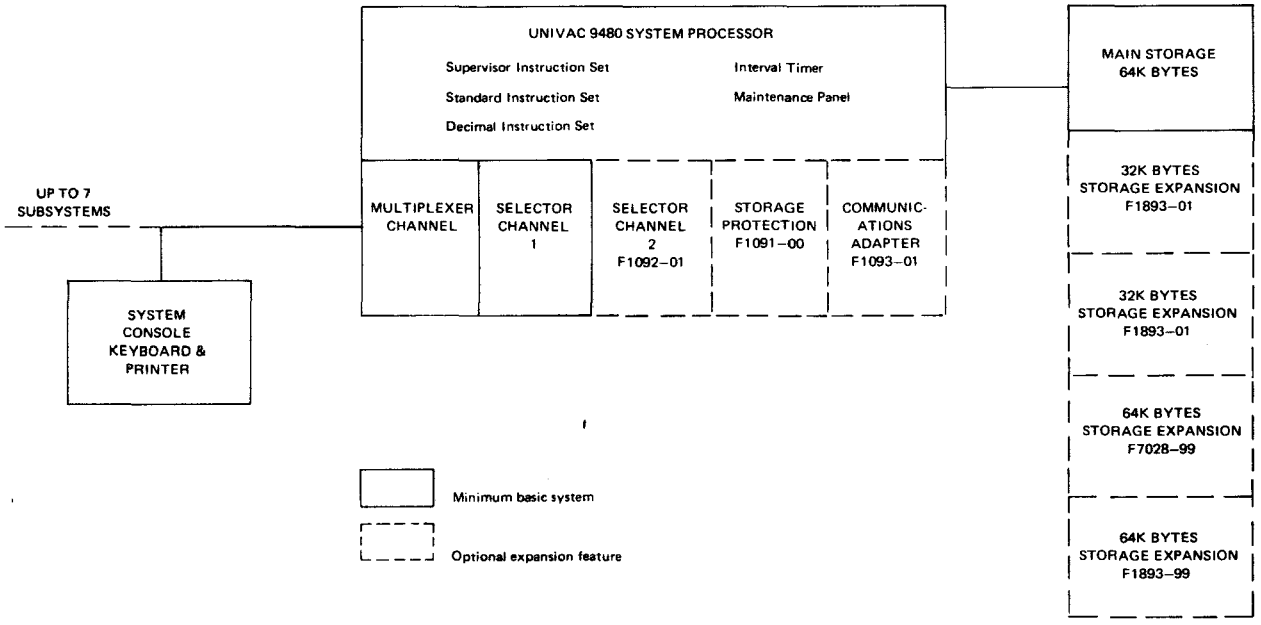



*Figure 1–3. UNIVAC 9480 System Processor Configuration*

*Table 1–1. Optional Hardware Expansion Features (Part 1 of 2)*

| Feature/Type Number | Name | Description |
|---|---|---|
| F1092–00 | Selector channel 1 | Provides additional I/O capability for eight high-speed subsystems. Maximum transfer rate is 333k bytes (includes command chaining). |
| F1092–01 | Selector channel 2 | Provides second selector channel with same characteristics as selector channel 1. |
| F1091–00 | Storage protection | Provides program write protection over contiguous increments of 512 bytes for a system up to 131,072 bytes or 1024 bytes for systems with more than 131,072 bytes. |
| F1093–00 | Communications adapter | Provides interfaces and control required to coordinate the transfer of data between this feature and a maximum of 12 line adapters (communications line terminals).<br><br>Provides 128 nonshared multiplexer subchannels for the Data Communications Subsystem (DCS). |

*Table 1–1. Optional Hardware Expansion Features (Part 2 of 2)*

| Feature/Type Number | Name | Description |
|---|---|---|
| Storage Expansion, UNIVAC 9400 System Processor | | |
| 7010–96 | Main storage (48K bytes) | Provides 49,152 bytes of main storage expandable to 65,536 bytes. |
| 7010–97 | Main storage (64K bytes) | Provides 65,536 bytes of main storage expandable to 98,304 bytes. |
| 7010–98 | Main storage (96K bytes) | Provides 98,304 bytes of main storage expandable to 131,072 bytes. |
| 7010–99 | Main storage (128K bytes) | Provides 131,072 bytes of main storage expandable to 196,608 bytes. |
| 7010–75 | Main storage (192K bytes) | Provides 196,608 bytes of main storage expandable to 262,144 bytes. |
| 7010–74 | Main storage (256K bytes) | Provides maximum of 262,144 bytes of main storage. |
| F1110–02 | Storage expansion (16K bytes) | Provides 16,384 bytes of storage to be added to type 7010–95 main storage for a total of 49,152 bytes. |
| F1110–03 | Storage expansion (16K bytes) | Provides 16,384 bytes of storage to be added to type 7010–96 main storage for a total of 65,536 bytes. |
| 7010–02 | Storage expansion (32K bytes) | Provides 32,768 bytes of storage to be added to type 7010–97 main storage for a total of 98,304 bytes. |
| F1110–04 | Storage expansion (32K bytes) | Provides 32,768 bytes of storage to be added to type 7010–98 main storage for a total of 131,072 bytes. |
| 7010–73 | Storage expansion (64K bytes) | Provides 65,536 bytes of storage to be added to type 7010–99 main storage for a total of 196,608 bytes. |
| 7010–72 | Storage expansion (64K bytes) | Provides 65,536 bytes of storage to be added to type 7010–75 main storage for a maximum total of 262,144 bytes. |
| Storage Expansion, UNIVAC 9480 System Processor | | |
| F1893–01 | Storage expansion | Provides a 32,768-byte storage increment to be added to minimum main storage (65,532 bytes) for a total of 98,304 bytes of storage. |
| F1893–01 | Storage expansion | Provides a 32,768-byte storage increment to be added to a 98,304-byte storage for a total of 131,072 bytes of storage. |
| F7028–99 | Storage expansion | Provides a 65,536-byte storage increment to be added to a 131,072-byte storage for a total of 196,608 bytes of storage. |
| F1893–99 | Storage expansion | Provides a 65,536-byte storage increment to be added to a 196,608-byte storage for a maximum of 262,144 bytes of storage. |

## 1.4. CHARACTERISTICS

The processors are small-to-medium scale in size and capabilities, handling random or sequential batch processing, and communication-oriented data processing with a maximum repertoire of 70 instructions.

The characteristics, capabilities, and optional features available for the processor, main storage, I/O channels, and some typical peripheral devices are listed in Table 1–2.

*Table 1–2. UNIVAC 9400 and 9480 Characteristics*

| | | |
|---|---|---|
| **Processor** | | |
| Data organization | 8-bit byte | |
| Decimal multiply and divide instructions | Included as part of standard instruction repertoire | |
| Edit instructions | Included as part of standard instruction repertoire | |
| Add (binary) instruction time (two 32-bit words) | 6 microseconds | |
| Registers | 16 for problem program functions<br>16 for supervisor functions | |
| **Main Storage** | | |
| | **UNIVAC 9400** | **UNIVAC 9480** |
| Type | Plated wire | Semiconductor (chip) |
| Storage access time | 600 nanoseconds per 2 bytes | 600 nanoseconds per 2 bytes |
| Storage capacity | Minimum 32K bytes; maximum 256K bytes in 16K-, 32K-, and 64K-byte increments | Minimum 64K bytes; maximum 256K bytes in 32K- and 64K-byte increments |
| Storage protection | Optional feature | Optional feature |
| I/O Channels | | |
| Multiplexer channel | One – standard | One – standard |
| Multiplexer channel transfer rate | 85,000 bytes per second | |
| Selector channels | Two – maximum as optional features | One – standard, one as an optional feature |
| Selector channel transfer rate | 333,000 bytes per second | |
| Communications adapter | Up to 64 duplex lines as an optional feature | |
| Typical Peripheral Devices | | |
| Card reader | 600 or 1000 cards per minute | |
| Card punch | 250 cards per minute | |
| Read/punch | Optional feature to card punch | |
| Printer | 840 to 1600 lines per minute, depending on print drum | |
| Magnetic tape | Transfer rate of 34,160 to 192,000 bytes per second | |
| Disc storage | Auxiliary storage for 29.18 or 58.35 million bytes per disc drive | |

# 2. Processor

## 2.1. INFORMATION FORMATS

Data and instructions are transmitted in single or multiple 8-bit increments called bytes. Up to four bytes of information may be transmitted in parallel between various sections of the system. Instructions are made up of half-word lengths and can be 1, 2, or 3 half words long.

A half word is defined as a field containing two consecutive bytes and is basic for the instructions. A full word is a field containing four consecutive bytes. Figure 2–1 illustrates the various word formats and bit numbering.

Fixed-length fields such as half words and full words are located in main storage on a integral boundary for the unit of information. Instructions must be located on half-word boundaries. An integral boundary is defined as main storage address for a unit of information that is a multiple of the length of the unit in bytes, as follows:

Half word (two bytes) — multiple of 2

Full word (four bytes) — multiple of 4

Storage addresses are expressed in binary form within the processor. Integral boundaries for half words and full words are given as binary addresses in which 1 or 2 of the low-order bits are 0, respectively.

Variable-length fields are not limited to integral boundaries and can start at any byte address.

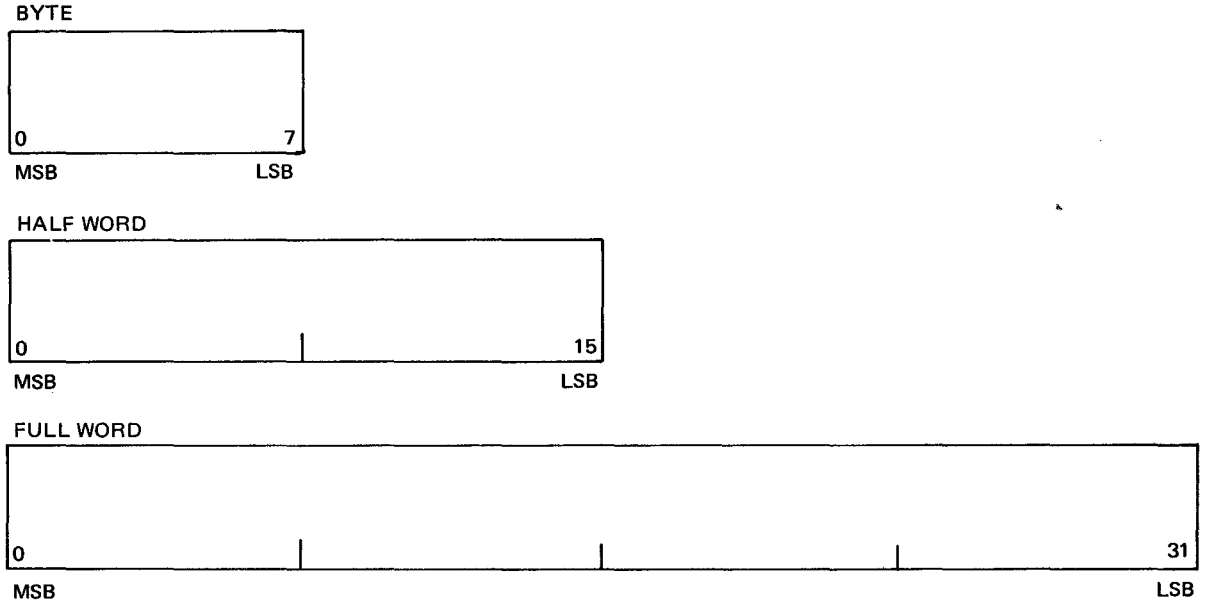



*Figure 2–1. Information Formats*

### 2.1.1. Data Formats

Data is represented in several different formats, depending on the type of instruction that is to manipulate the data.

#### 2.1.1.1. Fixed-Point Numbers

Fixed-point numbers are represented in one of three fixed-length formats consisting of one sign bit (bit 0) followed by a binary integer field. When the sign bit is set to 1, the integer represents a negative value; when set to 0, the integer represents a positive value. Negative integers are represented in the twos complement notation.

The half-word and full-word formats are as follows:

HALF WORD


FULL WORD

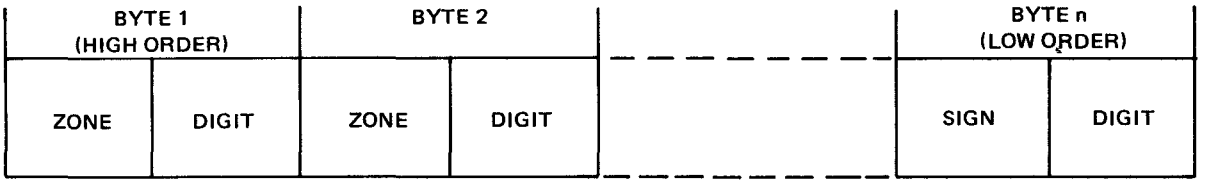

When held in one of the 16 general registers, a fixed-point number is treated as a 32-bit operand.

#### 2.1.1.2. Decimal Numbers

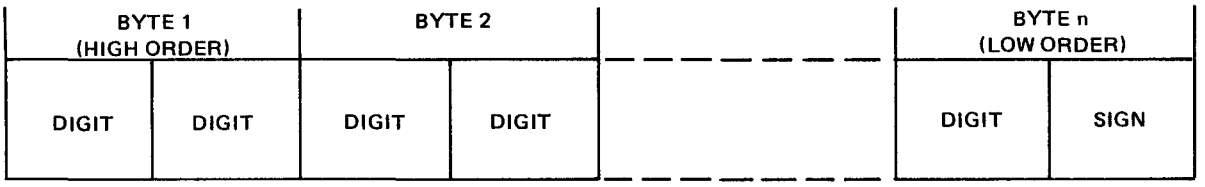
Decimal numbers are represented in signed absolute form and in either packed or unpacked formats with variable lengths. These are used in decimal arithmetic and logical operations. For decimal operations, the numbers are again processed from data stored in the general purpose registers or in main storage, and the results are returned to the general purpose registers or sent to main storage, requiring a field area of from 1 to 16 eight-bit bytes.

The packed and unpacked decimal formats are:

UNPACKED DECIMAL FORMAT




PACKED DECIMAL FORMAT



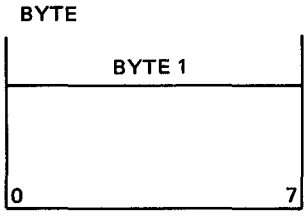
### 2.1.1.3. Logical Information

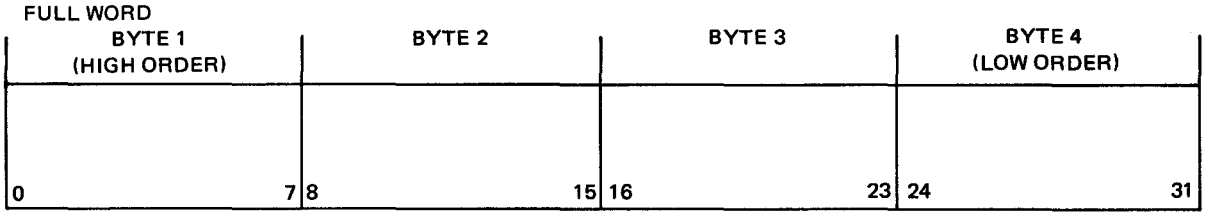
Logical information is processed as fixed-length or variable-length data. Operations using this type of information include moving, comparing, translating, editing, bit testing, bit manipulation, and shifting.

Fixed-length data, which consists of 1 or 4 bytes, is processed from data stored in the general purpose register or in main storage, and the results are returned to these registers or to main storage.

Data in the general registers normally occupies all 32 bits. In logical operations, all bits are treated uniformly, with no distinction made between sign and numeric bits. In some operations, only the low-order eight bits of a register are used; the remaining 24 bits are unchanged.

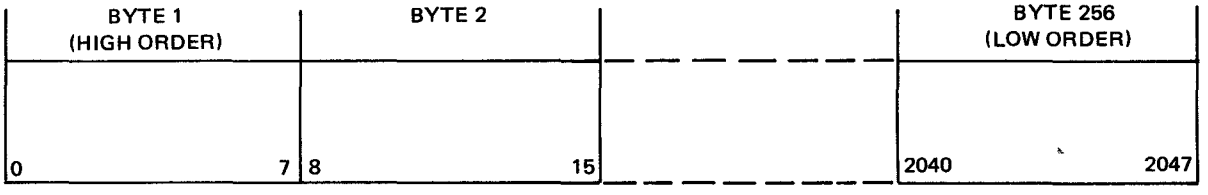
Logical data in fixed-length format is:




Variable-length data contains up to 256 bytes of alphabetic or numeric character codes (alphanumeric data). This information is again processed from data stored in the processor or in main storage, and the results are returned to the processor or to main storage; the information occupies fields that may start at any byte address. The processing order of variable-length data is from high-order (most significant) byte to the low-order (least significant) byte.

Logical data in variable-length format is:



## 2.1.2. Instruction Formats

Instructions are identified by an opcode, which is the most significant byte (eight bits) of the instruction. Two general categories of instructions related to a processor state are available to the programmer:

- Supervisor (Privileged) Instructions

  Privileged instructions can be executed only when the processor is in the supervisor state. The current PSW specifies the operating state of the processor. Processor states may switch as a result of an interrupt condition that causes a new PSW to be obtained from storage or from a load-program-status-word (LPSW) instruction. If the program in program mode attempts to execute a privileged instruction, a program exception interrupt occurs. It should be noted that the supervisory routines use the privileged instructions, and the problem programmer should avoid these instructions.

- Problem (Nonprivileged) Instructions

  The problem instructions can be executed either in the supervisor or in the program state.

The format used for programmed instructions for directing peripheral devices and processing data may vary in format and length. The format used is dictated by the operation to be performed and the operand location. The length of the instruction is dictated by the format and is either 1, 2, or 3 half words.

The five instruction formats are shown in Figure 2–2 and are identified as follows:

- register-to-register (RR);
- register-to-indexed storage (RX);
- register-to-storage (RS);
- storage-to-immediate operand (SI); and
- storage-to-storage (SS).

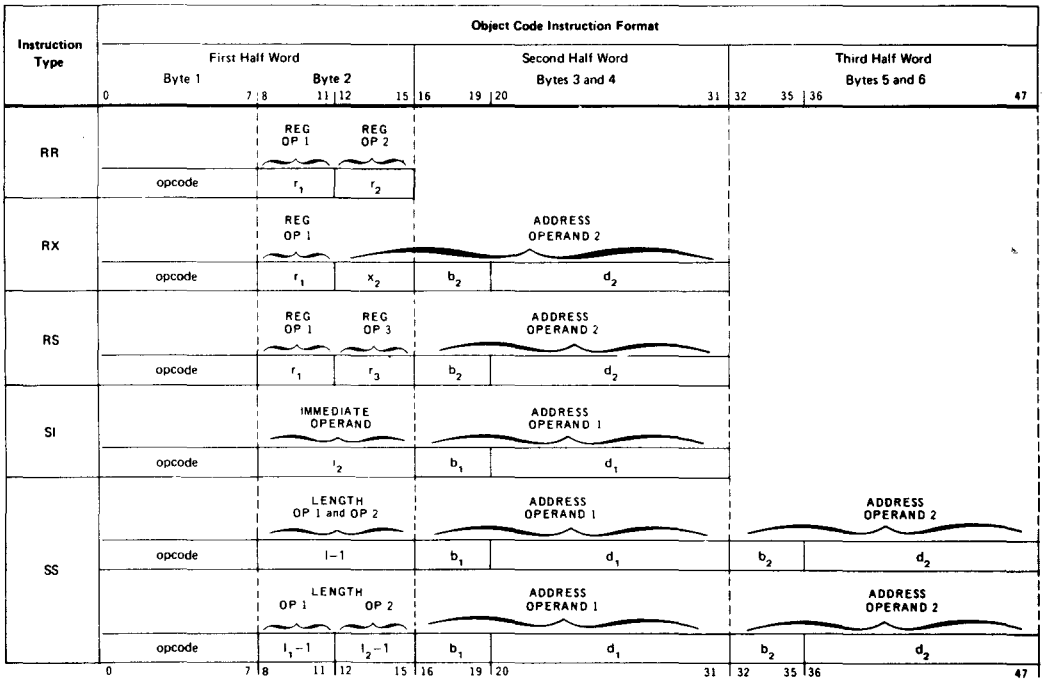



*Figure 2–2. Instruction Formats (Part 1 of 2)*

| Symbol | Definition |
|---|---|
| opcode | Instruction operation code |
| $r_1$ | Number of a general register containing the first operand. If an arithmetic compilation, $r_1$ field specifies where result is to be stored. For RS instruction, $r_1$ field contains either the first operand or the number that represents a boundary for general register usage. For RR and RX instructions, the $r_1$ field is used as a mask in the branch-on-condition (BC) instruction. |
| $r_2$ | Number of a general register containing the second operand. Both $r_1$ and $r_2$ may be the same if the same operand is used. |
| $r_3$ | Number of a general register containing the third operand, or the number representing a boundary for general register usage. This field is ignored when executing shift instructions. |
| $x_2$ | The number of a general register containing an index number. When field is 0, index number is 0. |
| $i_2$ | Immediate data used as operand 2 of an SI instruction. |
| $l_1$ | For instructions with opcodes Fx (where x is any hexadecimal digit 0 through F), contains a 4-bit number specifying the number of additional bytes in the first operand field that are to the right of the leftmost byte. |
| $l_2$ | For instructions with opcodes Fx (where x is any hexadecimal digit 0 through F), contains a 4-bit number specifying the number of additional bytes in the second operand field that are to the right of the leftmost byte. |
| l | For instructions with opcodes Dx (where x is any hexadecimal digit 0 through F), $l_1$ and $l_2$ are combined as l to form an 8-bit number specifying the number of additional bytes that are to the right of the operand address. Operands 1 and 2 may overlap except for the decimal-multiply and decimal-divide instructions. |
| $b_1$ | The number of a general register containing an index number representing the base address of the first operand. When 0, base address is 0. |
| $b_2$ | The number of a general register containing an index number representing the base address of the second operand; when 0, the base address is 0. |
| $d_1$ | Contains a 12-bit number that, when added to the contents of $b_1$, represents the address of the leftmost (high order) byte of the first operand. |
| $d_2$ | Contains a 12-bit number that, when added to the contents of $B_2$, represents the address of the second operand. For shift instructions, it represents the number of bits of shifting to be performed. For an SS instruction, it represents the address of the leftmost (high order) byte of the second operand. |
| op 1 | Operand 1 |
| op 2 | Operand 2 |
| op 3 | Operand 3 |

*Figure 2–2. Instruction Formats (Part 2 of 2)*

## 2.1.3. Fixed-Point Arithmetic

Fixed-point arithmetic operations include binary arithmetic on operands serving as addresses, indexed quantities, counts, and fixed-point data. In general, both operands are signed (negative or positive) and 32 bits long. Sixteen-bit operands may be specified to conserve storage locations. Negative quantities are held in the two's-complement form. One operand is always in one of the 16 general registers; the other can be in storage or in another general register. The fixed-point instructions (RR, RX, and RS) provide for loading, adding, subtracting, comparing, storing, and shifting of fixed-point operands. The condition code (2.1.3.3) is set as a result of all add, subtract, compare, and shift operations.

### 2.1.3.1. Fixed-Point Data Formats

Fixed-point numbers occupy a fixed-length format consisting of a sign bit followed by the integer field. When held in one of the general registers, a fixed-point quantity consists of a 31-bit integer field and occupies all 32 bits of the register. In register-to-register operations, the same register may be specified for both operand locations. Fixed-point data in storage occupies a 32-bit full word or a 16-bit half word with a binary integer field of 31 or 15 bits, respectively. A half word in storage is extended to a full word as the operand is retrieved from storage. Subsequently, the operand participates as a full-word operand.

### 2.1.3.2. Fixed-Point Number Representation

All fixed-point operands are treated as signed integers. Positive numbers are true binary notation with the sign bit set to 0. Negative numbers are represented in the twos complement notation with a 1 in the sign bit position. The twos complement of a number is obtained by inverting each bit of the number and adding a 1 in the low-order bit position. This type of number representation can be considered the low-order portion of an infinitely long representation of the number. When the number is positive, all bits to the left of the most significant bit of the number, including the sign bit, are 0's. When the number is negative, all these bits, including the sign bit, are 1's. When an operand must be extended with high-order bits, the expansion is achieved by prefixing a field in which each bit is set equal to the high-order bit of the operand.

The twos-complement notation does not include a negative 0. It has a number range in which the set of negative numbers is one larger than the set of positive numbers. The maximum positive number consists of a 1's integer field with a sign bit of 0; the maximum negative number consists of 0's integer field with a 1 bit for the sign.

The processor cannot represent the complement of the maximum negative number. When an operation (such as a subtraction from 0) produces the complement of the maximum negative number, the number remains unchanged, and a binary overflow exception interruption is recognized. An overflow does not result when the number is complemented and the final result is within the representable range. An example of this is a subtraction from minus 1.

The sign bit is leftmost in a number. An overflow carries into the sign bit and does not change even if the significant high-order bits are shifted out. The twos-complement notation is particularly suited to address computation and multiple-precision arithmetic. The twos-complement representation of a negative number may be considered the sum of the integer part of the field, taken as a positive number, and the maximum negative number. Therefore, in multiple-precision arithmetic, the low-order fields should be treated as positive numbers. Also, when negative numbers are shifted to the right, the resulting rounding off, if any, is toward minus infinity and not toward 0.

### 2.1.3.3. Fixed-Point Condition Code

The results of fixed-point add, subtract, compare, and shift operations are used to set the condition code (bits 34 and 35) in the program status word (2.2.1). Loading and storing operations leave this code undisturbed. The condition code can be used for decision-making by subsequent branch-on-condition instructions.

The condition code can be set to reflect three types of results for fixed-point arithmetic. For most operations, $00_2$, $01_2$, or $10_2$ condition codes indicate a zero, less-than-zero, or greater-than-zero content of the result register, respectively; an $11_2$ condition code is used when the result overflows. For a comparison, the $00_2$, $01_2$, or $10_2$ condition codes indicate that the first operand is equal to, less than, or greater than, the second operand, respectively.

## 2.1.4. Decimal Arithmetic

Decimal arithmetic operations on data occur in the packed format. In this format, two decimal digits are placed in one 8-bit byte. Inasmuch as data is often communicated to or from external devices in the zoned (unpacked) format, the necessary format conversion operations also are provided in the decimal instruction (SS) group.

Data is interpreted as integers, right-aligned in their fields, and is kept in true notation with a sign in the low-order 4-bit byte.

Processing occurs from right to left between storage locations. All decimal arithmetic instructions are a 2-address SS format. Each address specifies the leftmost byte of an operand. Associated with this address is a length field indicating the number of additional bytes that the operand extends beyond the first byte.

The decimal arithmetic instruction set provides for adding, subtracting, comparing, multiplying, and dividing, as well as the format conversion of variable-length operands.

The condition code is set as a result of all addition, subtraction, and comparison operations.

### 2.1.4.1. Decimal Data Formats

Decimal operands reside in main storage only. They occupy fields that may start at any byte address and consist of 1 to 16 bytes.

Lengths of the two operands specified in an instruction need not be the same. They are considered to be extended with 0's to the left of the high-order digits. Results never exceed the limits set by addresses and length specification. Lost carries or lost digits from arithmetic operations are signaled as a decimal overflow exception interruption.

Operands are either in the packed or unpacked (zoned) format. In the packed format, two decimal digits are normally placed adjacent in a byte, except for the rightmost byte of the field. A sign is placed to the right of decimal digits in the rightmost byte. Both digits and a sign are encoded and occupy four bits each. In the unpacked (zoned) format, the low-order four bits of the byte, the numeric, are normally occupied by a decimal digit. The four high-order bits of a byte are called the zone, except for the rightmost byte of the field, where, normally, the sign occupies the zone position.

Arithmetic operations are performed with operands and results are in the packed format. In unpacked format, the digits are represented as part of an alphanumeric character set. A pack instruction is provided to transform unpacked data into packed data and an unpack instruction performs the reverse transformation. The edit instruction is used to change data from packed to unpacked format.

In move-type operations, the operand digits and sign are not checked, and the operand fields may overlap without restrictions. For example, in the move characters (MVC) instruction, the multiple-byte first operand is in storage at address $d_1+(b_1)$ and the length specification is I (SS type instruction). The multiple-byte second operand is in storage at address $d_2+(b_2)$ and the length specification is I. The second operand replaces the first operand. Storage occurs from left to right, one byte at a time.

### 2.1.4.2. Decimal Number Representation

Numbers are represented as right-aligned true integers with a plus or minus sign. The digits 0 through 9 are binary-coded 0000 through 1001. The codes $1010_2$ through $1111_2$ are invalid as digits. This set of codes is interpreted as sign codes with $1010_2$, $1100_2$, $1110_2$, and $1111_2$ recognized as plus, and with $1011_2$ and $1101_2$ recognized as minus. The codes $0000_2$ through $1001_2$ are invalid as sign codes. The zones are not tested for valid codes, inasmuch as they are eliminated in changing data from the unpacked to packed format.

The sign and zone codes generated for all decimal arithmetic results differ for EBCDIC and ASCII. The choice between the two codes is determined by the a field (bit 12) of the PSW (2.2.1). When the a field is 0, the preferred EBCDIC codes are generated. These are plus, $1100_2$; minus, $1101_2$; and zone, $1111_2$. When the a field is 1, the preferred ASCII codes are generated. These are plus, $1010_2$; minus, $1011_2$; and zone, $0101_2$.

### 2.1.4.3. Decimal Condition Code

The results of all addition, subtraction, and comparison operations are used to set the condition code. All other decimal arithmetic operations leave the code unchanged. The condition code can be used for decision making by subsequent branch-on-condition instructions.

The condition code can be set to reflect two types of results for decimal arithmetic operations. For most operations, $00_2$, $01_2$, and $10_2$ condition codes indicate a zero, less-than-zero, and greater-than-zero for the content of the result field, respectively. The $11_2$ condition code is used when the result of the operation overflows.

For the comparison operation, $00_2$, $01_2$, or $10_2$ condition codes indicate that the first operand is equal to, less than, or greater than the second operand field, respectively.

## 2.1.5. Logical Operations

A set of instructions is provided for the logical manipulation of data, with the operands treated generally as 8-bit bytes. In a few cases, the left or right four bits are treated separately or operands are shifted a bit at a time. The operands are either in storage or in a general register. Some operands are introduced from the instruction stream (SI instructions).

Processing of data in main storage proceeds from left to right through fields that can start at any byte position. In the general registers, the processing generally involves the entire register contents.

Except for the edit instruction, data is not treated as numbers. Editing provides a transformation from packed decimal to alphanumeric characters. The set of logical operations includes moving, comparing, bit setting, bit testing, translating, editing, and shift operations.

The condition code is set as a result of all logical comparing, connecting, testing, and editing operations.

### 2.1.5.1. Logical Data Formats

Data resides in general registers or in storage, or is introduced from the instruction stream. The data size can be a single word, a single character, or a variable length. When two operands are involved, they are of equal length, except in the edit instruction. Data in the general registers normally occupies all 32-bits. Bits are treated uniformly, and no distinction is made between sign and numeric bits. In a few operations, only the low-order eight bits of a register are involved, leaving the remaining 24 bits unchanged.

When the processor has access to less than 128K bytes of main storage, the load-address instruction introduces a 17-bit address into a general register, and the high-order 15 bits of the register are made 0. If the processor has access to a main storage greater than 128K bytes, the load-address instruction introduces a 16-bit address into a general register, and the high-order 14 bits of the register are made 0.

In RS operations, the storage data occupies either a full word of 32 bits or a byte of eight bits. The word must be located on full-word boundaries; that is, the two low-order bits of its address must be 0's.

In SS operations, data is of variable-length field format, starting at any byte address and continuing to a total of 256 bytes, processing again being from left to right.

Operations introducing data from the instruction stream into main storage as immediate data are restricted to an 8-bit byte. Only one byte is introduced from the instruction stream, and only one byte in storage is involved.

Editing requires a packed decimal field and generates zoned decimal digits. These digits, signs, and zones are recognized and generated the same as for decimal arithmetic. Therefore, no internal data structure is required and all bit configurations are considered valid.

The translating operations use a list of arbitrary values. A list provides a relationship between an argument (the quantity used to reference the list) and the function (the contents of the location related to the argument). The purpose of the translation may be to convert data from one code to another or to perform a control function.

A list is specified by an initial address that designates the leftmost byte location of the list. The byte from the operand to be translated is the argument. The actual address used to address the list is obtained by adding the argument to the low-order positions of the initial address. As a result, the list contains 256 eight-bit function bytes. In cases where it is known that not all 8-bit argument values occur, it may be possible to reduce the size of the list.

In SS operations, the operand fields can be defined in such a way that they overlap. The effect of this overlap depends upon the operation. When the operands remain unchanged, as in the compare instruction, overlapping does not affect the execution of the instruction. For the edit and translate instructions, one operand is replaced by new data, and the execution of the operation may be affected by the amount of overlap and the manner in which data is fetched and stored. For purposes of evaluating the effect of overlapped operands, consider that the data is handled one byte at a time. All overlapping fields are considered valid, but in editing, overlapping fields give unpredictable results.

### 2.1.5.2. Logical Condition Code

The results of most logical operations are used to set the condition code in the PSW. The load-address, insert-character, store-character, and translate instructions, along with the moving and shift operations, leave the condition code unchanged. The condition code can be used for decision-making by subsequent branch-on-condition instructions.

The condition code can be set to specify the following results for logical operations:

- Compare-logical instructions

  The $00_2$, $01_2$, or $10_2$ values indicate that the first operand is equal to, less than, or greater than the second operand, respectively.

- Test-under-mask instruction

  The $00_2$, $01_2$, or $10_2$ values indicate that the selected bits are 0's, mixed 0's and 1's, or 1's, respectively.

- Editing instructions

  The $00_2$, $01_2$, or $10_2$ values indicate a zero, less-than-zero, or greater-than-zero content, respectively, of the last result field.

## 2.1.6. Branching

Instructions are executed primarily in the sequential order of their locations. A departure from this normal sequential operation may occur when branching is performed. The branching instructions provide a means for making a 2-way choice:

- to reference a subroutine; or
- to repeat a segment of coding, such as a loop.

Branching is performed by introducing a branch address as a new instruction address. The branch address may be obtained from one of the general registers or it may be the address specified by the instruction. The branch address is independent of the updated instruction address.

Operational details of branching are determined by the condition code, or by the results in the general registers that are specified in the loop-closing operations.

During a branching operation, the rightmost half of the PSW, including the updated instruction address, may be stored before the instruction address is replaced by the branch address. The stored information may be used to link the new instruction sequence with the preceding instruction sequence.

### 2.1.6.1. Normal Sequential Operation

Normal operation of the processor is controlled by the sequence of instructions. An instruction is retrieved from a location specified by the instruction address field of the PSW. The instruction address is incremented by the number of the bytes of the instruction in order to address the next instruction in sequence. This new instruction-address value, called the updated instruction address, replaces the previous contents of the instruction-address field in the PSW. The current instruction is executed, and the same steps are repeated, using the updated instruction address to fetch the next instruction.

Instruction length is a half word or multiple of a half word. An instruction can be composed of up to three half words. The number of half words in an instruction is specified by the instruction length code (ILC), bits 32 and 33 of the PSW. An $01_2$ code specifies a half word instruction; a $10_2$ code specifies a 2-half-word instruction, and an $11_2$ code specifies a 3-half-word instruction. Code $00_2$ specifies length is not available.

Main storage wraps around from the maximum addressable storage location, byte location 262,143 to byte location 0. An instruction with a last half word at the maximum storage location is followed by the instruction at address 0. Also, a multiple half-word instruction may straddle the upper storage boundary; no special indication is given in these cases.

Actually, an instruction is retrieved from main storage after the preceding operation is compiled and before execution of the current operation, even though physical storage-word size and overlap of instruction execution with storage access may cause the retrieved instruction to be different.

A change in the sequential operation may be caused by branching, interruption, or manual intervention. Sequential operation is initiated and terminated from the system console and maintenance panel. (It is possible to modify an instruction in main storage with an immediately preceding instruction.)

Exceptional instruction addresses or operation codes cause a program interruption. When the interruption occurs, the current PSW is stored as an old PSW, and a new PSW is obtained. The interrupt code in the old PSW identifies the cause of the interruption. The following program interruptions can occur in normal instruction sequencing, independently of the instruction performed.

- Illegal operation exception

  Caused by an operation code not having been assigned.

- Addressing exception

  Caused by a reference to main storage outside the problem program boundaries when the processor is in the problem state.

- Specification exception

  Caused when reference to main storage is not on the proper boundary.

In each case, the operation is suppressed; therefore, the condition code and the data in main storage and in the register remains unchanged. The instruction address stored as part of the old PSW is updated by the number of half words indicated by the ILC in the old PSW.

It should be noted that an addressing exception address can occur when normal instruction sequencing proceeds from a valid storage region into an unavailable region or following a branching operation.

When the last location in available main storage contains an instruction that again introduces a valid instruction address, no program interruption is caused, even though the updated address designates an unavailable location. The main storage or register address specification of an instruction with an unassigned operation code can cause an addressing or specification exception interruption when the requirements for the particular instruction class are not met.

### 2.1.6.2. Decision-Making

Branching can be conditional or unconditional. Unconditional branches replace the updated instruction address with the branch address. Conditional branches can use the branch address or can leave the updated instruction address unchanged. When branching occurs, the instruction is called successful; otherwise, it is called unsuccessful. Whether a conditional branch is successful depends on the result of operations concurrent with the branch or preceding the branch. The successful branch is represented by the branch-on-count instruction. The unsuccessful branch is represented by a branch-on-condition instruction which inspects the condition code that reflects the result of a previous arithmetic, logical, or I/O operation.

### 2.1.6.3. Branching Condition Code

A condition code provides a means of data-dependent decision-making. The code is inspected to qualify the execution of the condition-branch instructions. This code is set by some operations to reflect the result of the operation independently of the previous setting of the code and remains unchanged for all other operations. The condition code occupies bit positions 34 and 35 of the PSW and is stored as part of the rightmost half of the PSW in a branch and link operation. A new condition code is obtained by a load-PSW or set-program-mask instruction, or by a new PSW being loaded as a result of an interruption.

The condition code indicates the outcome of some of the arithmetic, logical, or I/O operations. It is not changed for any branching operation.

## 2.2. CONTROL SECTION

The control section of the processor controls the sequence that instructions are executed, interprets and controls the execution of each individual instruction, initiates cycling of main storage, maintains the program address location counter, provides processor state of operation, and handles all hardware aspects of interrupt handling, error checking, and main storage protection. The processor can operate in one of two program states: supervisor state or problem state. These two states are controlled by the contents of the current program status word register (PSW register).

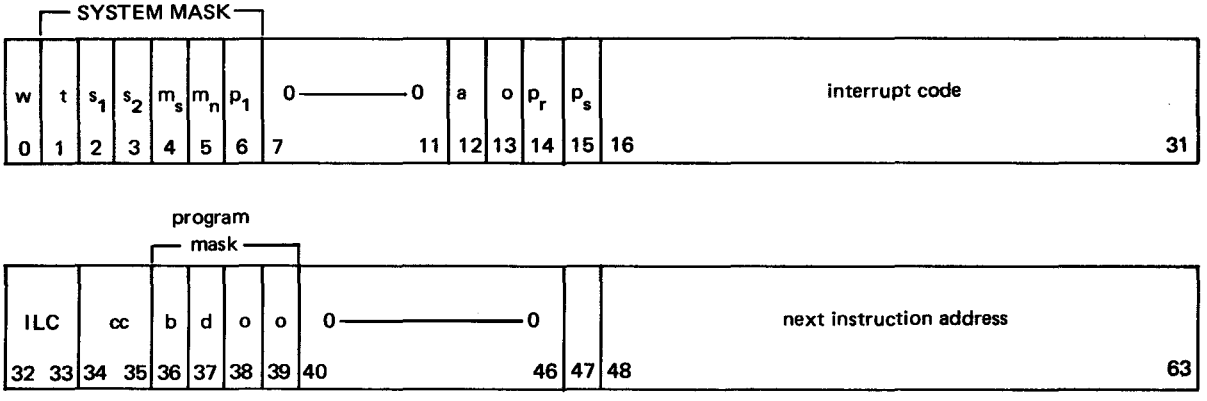
The processor status, processor control, and operating conditions are determined by and reflected in the seven pairs of program status words (PSW) stored in the PSW registers (Figure 4–1). These status words are associated with seven classes of interrupts (2.2.2).

### 2.2.1. Program Status Word

The contents of the program status word (PSW) registers contain, at any given time, one of seven PSWs that control the operation of the processor. Any interrupt causes this current PSW to be transferred to the old PSW storage area for that class of interrupt, and the new PSW is loaded into the PSWR from main storage. The detailed status of the processor when the interrupt occurred is thereby stored for further interrogation, and a new operational state for the processor is established.

The format and description of both the old and new PSWs are identical.

Format:




Bit functions:

w (bit 0)

Write protection bit (if feature is installed)

w=1, write limits check is performed.

w=0, check is ignored.

System mask (bits 1–6)

Permits one of six interrupt levels in the processor to be masked. If a mask bit is set to 1, the interrupt corresponding to the mask bit is allowed. If set to 0, a corresponding interrupt is inhibited. The individual bit masks are defined as:

t (bit 1) – timer interrupt

$s_1$ $s_1$ (bit 2) – selector channel 1 interrupt

$s_2$ $s_2$ (bit 3) – selector channel 2 interrupt

$m_s$ $m_s$ (bit 4) – shared multiplexer subchannel interrupt

$m_n$ $m_n$ (bit 5) – nonshared multiplexer subchannel interrupt

$p_1$ $p_1$ (bit 6) – program exception interrupt

Bits 7–11, 13, 38 and 39, and 40–46

Unassigned and must be set to 0 by the software.

a (bit 12)

Internal operating mode

a=1, operates in ASCII mode.

a=0, operates in EBCDIC mode.

$p_r$ (bit 14)

Register designation

$p_r$=1, hardware uses problem general registers.

$p_r$=0, hardware uses supervisor general registers.

$p_s$ (bit 15)

Processor state control

$p_s$=1, operates in problem state, nonprivileged instruction usage, and reserved low-order main storage protection.

$p_s$=0, operates in supervisor state, and all instructions are valid with no reserved low-order main storage protection.

Interrupt codes (bits 16–31)

Contains the interrupt code that is used in software analysis of interrupts. (See 2.2.2 for a complete definition of the contents of this field.)

ILC (bits 32, 33)

Instruction length code, specifies the length of the instruction just completed or terminated by the interrupt.

| Code | Description |
|---|---|
| $00_2$ | length not available |
| $01_2$ | 1 half word (RR) |
| $10_2$ | 2 half words (RS, RX, SI) |
| $11_2$ | 3 half words (SS1, SS2) |

CC (bits 34, 35)

Condition code, summarizes the result of arithmetic instructions. This code is used in branch-on-condition instructions.

| Code | Description |
|---|---|
| $00_2$ | 0 or equal |
| $01_2$ | negative, less than or mixed |
| $10_2$ | positive or greater |
| $11_2$ | all 1's |

Program mask (bits 36–37)

b (bit 36)

Program mask binary overflow:

b=1, binary overflow interrupt is permitted.

b=0, binary overflow interrupt is inhibited.

d (bit 37)

Program mask decimal overflow:

d=1, decimal overflow interrupt is permitted.
d=0, decimal overflow interrupt is inhibited.

Next instruction address (bits 46–63 or 47–63)

Contains instruction address. When a new PSW is loaded into the current PSW registers, this address (after being located) points to the first instruction in a program to be executed. Thereafter, this field is updated as each instruction is executed. When the current instruction address field of the old PSW is stored in low-order main storage, it is pointing to the instruction that would have been executed next had an interrupt or a branch instruction not occurred. Branch instructions replace the instruction address field in the current PSW in the case of a successful branch.

Bit 46 is used as the most significant bit for the next instruction address if main storage exceeds 128K bytes. When main storage does not exceed 128K bytes, this bit is is set to 0.

## 2.2.2. Interrupts

The interrupt hardware provides the processor with a means of reaction to external and internal error or monitoring conditions. The interrupt hardware also allows the processor to change state by storing the contents of the current PSW register in one of the seven (old) PSW main storage areas. A new PSW is then moved into the current PSW register for continued operation. Testing the interrupt code in each old PSW identifies the cause of that interrupt. Seven conditions have been assigned an interrupt priority to allow for handling two or more simultaneous interrupts. The following is a list of the interrupts in order of decreasing priority:

- Supervisor call — 1 (highest)
- Program exception — 2
- Timer — 3
- Selector channel 1 — 4
- Selector channel 2 — 5
- Shared multiplexer channel — 6
- Nonshared multiplexer channel — 7 (lowest)

### 2.2.2.1. Supervisor Call Interrupt

This interrupt occurs as a result of execution of a supervisor-call instruction (SVC). The interrupt is used to initiate communications between the problem program and the supervisor control program. (This interrupt cannot be masked.) During execution of an SVC instruction, the contents of bit positions 8 through 15 are stored in bit positions 24 through 31 of the interrupt code field in the old SVC PSW. Bits 16 through 23 of the old SVC PSW are set to 0. The highest byte of the field (bits 0 through 7) are set to 0. The instruction length code is set to $01_2$ (two bytes).

### 2.2.2.2. Program Exception Interrupt

Certain exceptions are recognized by the processor hardware during the execution of instructions. These exceptions are interrogated by the hardware and cause program interruptions. When an interrupt of this type occurs, the interrupt code field of the PSWR is set to the code identifying the source or sources of the interrupt. The contents of the PSWR are then stored in the old PSW location. The PSWR is then loaded with the contents of the new PSW location. The interrupt code fields are:

| Interrupt Code | Condition |
|---|---|
| 0000 0000 0001 0000 | Illegal operation |
| 0000 0000 0010 0000 | Privileged operation |
| 0000 0000 0011 0000 | Not used |
| 0000 0000 0100 0000 | Write protection* |
| 0000 0000 0101 0000 | Addressing exceptions |
| 0000 0000 0110 0000 | Specification exception |
| 0000 0000 0111 0000 | Not used |
| 0000 0000 1000 0000 | Binary overflow* |
| 0000 0000 1001 0000 | Not used |
| 0000 0000 1010 0000 | Decimal overflow* |
| 0000 0000 1011 0000 | Decimal divide exception |

**These interrupts may be masked*

The eight causes of the program exception interrupt conditions are:

- Illegal operation

  This interrupt is caused by an attempt to execute an instruction whose operation code is 00, 2x, 3x, 6x, 7x, Ax, Cx, Ex, or Fx. The ILC of the PSW is set to 1, 2, or 3.

- Privileged operation

  This interrupt occurs when a privileged operation is detected while the processor is in the problem state (PSW bit 15 set to 1). The ILC field of the PSW is $10_2$ during this interrupt.

- Write protection

  This interrupt occurs when the processor is in the problem state (PSW bit 15 set to 1), bit 0 of the current PSW is 1, and a write operation specifies addresses outside the bounds held in the limits register. The ILC field of the PSW is $10_2$ or $11_2$ during this interrupt.

  *NOTE:*

  *The storage protection feature must be installed for this interrupt to occur.*

- Addressing exception

  This interrupt is caused by a reference (read or write) to the reserved low-order 512 bytes of main storage addresses 0 through 511. The ILC field of the PSW is $10_2$ or $11_2$ during this interrupt.

- Specification exception

  This interrupt is caused by an operand address that does not specify the appropriate integral boundary for the operation in progress. The ILC field of the PSW is $10_2$ or $11_2$ during this interrupt.

- Binary overflow

  This interrupt occurs when performing fixed-point arithmetic operations and the carry-out of the high-order numeric bit does not agree with the carry-out of the sign bit position. The results are valid except for sign truncation. This interrupt occurs only if PSW bit 36 (binary overflow) is set to 1. The ILC field of the PSW is $01_2$ or $10_2$ during this interrupt.

- Decimal overflow

  This interrupt occurs when the results of a decimal arithmetic operation (other than multiply or divide) exceed the operand 1 field length (operand 1 being the receiving field). The initial length (ILC) specification must be such that $I_1$ is greater than or equal to $I_2$ for the overflow detection to be sensed. The results are valid even when truncated to the length of operand 1. The decimal overflow (PSW bit 37) must be set to 1 for the interrupt to occur. The ILC field of the PSW will be $11_2$ during this interrupt.

- Decimal divide exception

  This interrupt occurs when a quotient digit greater than 9 is formed. A quotient digit greater than 9 occurs when the absolute value of $I_2$+1 most significant bytes of operand 1 is not less than the absolute value of operand 2. It follows, that when operand 2 is greater than 0, and the most significant digit of operand 1 must be equal to zero.

  *NOTE:*

  *The illegal operation, privileged operation, binary overflow, decimal overflow, and decimal divide exception interrupt codes are mutually exclusive of any other.*

The write protection, addressing exception, and specification exception interrupt codes are not mutually exclusive. It must be understood that when multiple conditions arise, the code reflects the OR effects on the conditions (codes). A close examination of the various conditions reveals:

- Write protection is limited to instructions involving write;

- Addressing exception applies to all instructions specifying an operand address (read or write) and is not a privileged instruction. Instructions of the RR type and shifts, for example, are exempt.

The following are the combinations to be considered when multiple conditions arise. The connection lines shown demonstrate the ambiguities that exist when multiple conditions arise.

Combinations

WP=0100
SE=0110
AE=0101
BO=1000
DO=1010
DDE=1011
WP OR AE=0101
WP OR SE=0110
WP OR AE OR SE=-111
AE OR SE=0111

where:

WP = write protection
SE = specification exception
AE = addressing exception
BO = binary overflow
DO = decimal overflow
DDE = decimal divide exception
OR = OR operation

The following concerning the program exception interrupt also should be noted:

- The only program interrupt conditions recoverable (control returned to the program) are:
  - binary overflow = $1000_2$
  - decimal overflow = $1010_2$
  - decimal divide exception = $1011_2$

  All three are unique in identification and, therefore, the terminal versus nonterminal error decision in the supervisor is not affected by the possible existence of ambiguities.

- When a program has caused a terminal error, examine the possibility of the previously mentioned ambiguities and resolve by using the flow chart in Figure 2–3.

### 2.2.2.3. Timer Interrupt

This interrupt occurs when the interrupt count in the interval timer is decremented to 0. The interrupt count field is located in the first 12 bits of the timer word (2.2.4.2). The timer word is located at the address specified in the timer address field of the timer control word (TCW) (2.2.4.1). The count field is decremented by 1 every millisecond. The timer interrupt also can be inhibited by masking the proper interrupt field (bit 1 of the PSW). The interrupt code of the old PSW is not changed.

### 2.2.2.4. Multiplexer Channel Interrupt

This interrupt (channel or device) occurs where an interrupt condition has been generated on either a shared or nonshared multiplexer subchannel. The affected shared or nonshared PSW is exchanged; that is, the current PSW in the register is transferred to the old PSW main storage area and the new PSW is loaded into the register from main storage. The shared subchannel status is stored in the appropriate SCW location of reserved low-order main storage. The nonshared subchannel status is tabled under control of the status table SCW and the associated BCWs. The status identifies the source and cause of the interrupt. A system mask bit has been assigned for both shared and nonshared multiplexer channel interrupts. The device number (shared subchannel subsystem) is stored in the old PSW as the least significant eight bits (bits 24 through 31) of the interrupt code. The interrupt code is not conditioned when a nonshared subchannel status is stored.

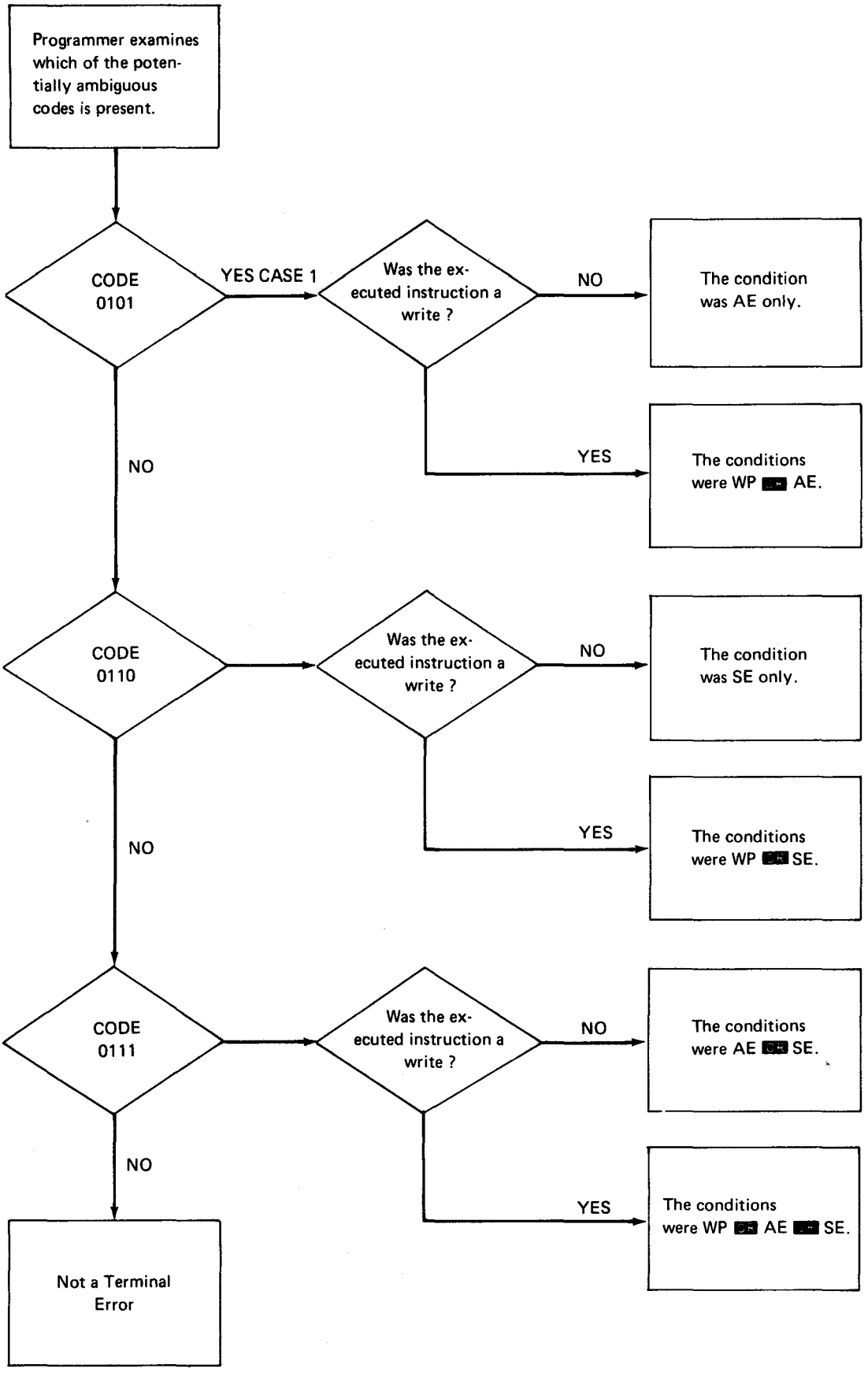



*Figure 2–3. Terminal Error Flowchart*

### 2.2.2.5. Selector Channel Interrupt

Any channel or device error status on either selector channel 1 or 2 causes this type of interrupt. Ending status signals (CHANNEL END, DEVICE END, etc.) also cause an interrupt if command chaining is not indicated. The selector channel status is stored in the appropriate channel status word (CSW) when the interrupt is processed (3.4.3). The PSWs are again exchanged and the stored status identifies the cause of the interrupt.

A system mask field (bit 2 or bit 3 of the PSW) is assigned for the selector channels. The device number of the subsystem is stored in the old PSW as the lower eight bits of the interrupt code.

### 2.2.3. Interrupt Status

All multiplexer states (channel and subchannel) cause an interrupt, except status generated during a start-I/O sequence. All error states on the selector channel cause an interrupt, except status generated during a start-I/O instruction. Ending status is not indicated.

### 2.2.3.1. Subchannel

The subchannel generates an interrupt and presents a status byte upon detection of any of the following:

- Buffer control interrupt

  This interrupt is generated to indicate a multiplexer nonshared subchannel buffer termination.

- Control byte interrupt

  This interrupt is generated when the flag bit c is set to 1 in the nonshared SCW and the control byte in this SCW is equal to the data byte being transferred.

### 2.2.3.2. Device

The control unit and/or the device generates an interrupt and presents a status byte upon detection of any of the following conditions:

- Attention

  This bit is set by the I/O device to indicate a condition other than those associated with the initiation, execution, or termination of an I/O operation. It also can indicate the transition of a device to a ready state, if the capability is equipped with a device.

- Status modifier

  This bit and the busy bit are set together to distinguish between control unit busy and device busy in multidevice subsystems. It is also set, with device end status bit, to indicate special ending conditions for command chaining operations (skip designation).

- Control unit end

  This bit can be generated by a control unit, when it has been freed for another operation. It can accompany channel end or device end, or may occur at a time in between if the control is interrogated by the initiation of another command.

- Busy

  This bit is set when the device control or control unit cannot execute the command because of a previous command in progress or a pending status condition.

- Channel end

  This bit is set when the channel is ready for a new operation. The time of occurrence of the channel end is dependent upon the type of control.

- Device end

  This bit is set when the device is ready for a new operation. If an unusual condition is detected during the initiation of a command, error status is presented without device end status.

- Unit check

  This bit is set when errors are detected during an operation. Unit check is a summary notification of a unit-associated malfunction. Detailed status is obtained by initiating a sense command.

- Unit exception

  This bit is set for abnormal conditions that may or may not be errors. Its meaning is applicable to the type of command and device involved in an operation.

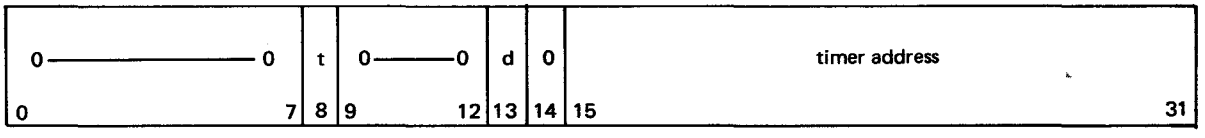
### 2.2.4. Timer

The operation of the timer is similar to the operation of the nonshared SCW and BCW. The timer consists of a timer control word (TCW) and its associated timer word. The TCW is stored in reserved low-order main storage (location 400); the timer word can be stored in any address in the lower 128K bytes of main storage on a full-word boundary.

#### 2.2.4.1. Timer Control Word

The format and the bit functions of the timer control word (TCW) are:

Format:




Bit functions:

bits 0–7, 9–12, 14
Unassigned and must be set to zero.

t (bit 8) – terminate flag
t = 1, timer operation stops
t = 0, timer operation continues

d (bit 13) – decrement field

d = 1, running time counter to be decremented by 1 every millisecond
d = 0, running time counter to be incremented by 1 every millisecond

timer address (bits 15–31)

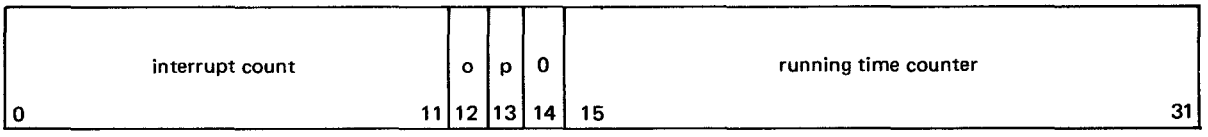
Contains an address of the timer word (on full-word boundary); timer address is limited to the lower 128K bytes of main storage.

### 2.2.4.2. Timer Word

The timer word contains two count fields, the interrupt count and the running time counter. This word is similar in function to the nonshared BCW.

The format and bit functions of the timer word are:

Format:




Bit functions:

interrupt count (bits 0–11)

Interrupt count decremented by 1 every millisecond.

o,p (bits 12, 13) – operation field

$00_2$ = stop
$01_2$ = chain (contiguous)
$10_2$ = link (noncontiguous)
$11_2$ = stop immediate

bit 14

Is normally 0 and can be used as an overflow extension to the running time counter. This automatically will be the case if the running time counter field is not reset prior to the overflow condition.

running time counter (bits 15–31)

Incremented (or decremented) every millisecond.

*NOTES:*

1. *The t field (bit 8 of the TCW) is not set to 1 when the interrupt count of the timer word is decremented to 0 and the stop operation code is set (bits 12 and 13 of the timer word).*

2. *Bit 14 of the timer word can be used as an extension of the running time counter. This occurs automatically if the running time counter is not set to 0 prior to an overflow condition. If bit 14 and the running time counter are not reset to 0 prior to an overflow into bit 13, a clock malfunction occurs, because a bit in bit position 13 indicates a different operation code (chain).*

3. *If the same full word is to be used repeatedly for the timer, the stop ($00_2$) operation code must be indicated in bits 12 and 13 of the timer word.*

### 2.2.4.3. Timer Operation

The operation of the timer is similar to the multiplexer sequence that updates the shared BCW. The running time counter is a 17-bit count field capable of counting to a maximum of 131,071 milliseconds (approximately 2.2 minutes). The initial count can be set to any value under control of the supervisor control program. This count field can be incremented or decremented by setting the d control field of the TCW to 0 or 1, respectively. An interrupt does not occur as a result of passing 0 or the maximum count. To retain main storage integrity, bit 14 of the timer word must be reset to 0 if it is set to 1 when the maximum count ($1FFF_{16}$ if d is 0, or $0000_{16}$ if d is 1) is passed. The counting sequence is initiated between instruction sequences or at specific intervals (instruction hold times) of any instruction sequence. This permits the count values to be valid to within 1-millisecond increments. The interrupt count is a 12-bit count field capable of being decremented for a maximum of 4096 milliseconds. The initial count is set to any value under control of the supervisor control program. The count is decremented at 1-millisecond intervals. A decrement to 0 causes the initiation of a timer interrupt, which causes the processor to exchange the PSW if the appropriate mask bit is set to 1. The interrupt count field is decremented through 0 to a maximum count.

The interrupt count can be loaded by the supervisor control program with a maximum count of 4096. This count is decremented by 1 each millisecond. When 0 is passed, the field resets and a timer interrupt (2.2.2.3) occurs, depending on the setting of the system mask. The time of day is generated by software. The user has no control over the contents of the timer control word or the timer word.

# 3. Input/Output Section

## 3.1. GENERAL

The input/output section of the processor initiates, directs, and monitors the transfer of data between main storage and the peripheral subsystems, and consists of one multiplexer channel and up to two selector channels. A communications adapter also is available, by feature addition, providing up to 128 nonshared subchannels for servicing communications line terminals.

The processor initiates all I/O operations by issuing I/O instructions to a selected channel and a selected subsystem connected to that channel. Once operation is successfully initiated, the channel maintains control of data transfers between main storage and the subsystem, independently of the processor. Upon completion of the I/O operation, the state of the channel and subsystem is presented to the software by way of the appropriate status words and I/O interrupts.

## 3.2. CHANNEL AND SYSTEM FUNCTIONAL INTERFACES

Both the multiplexer and selector channels communicate with the other components of the system by way of the following relatively independent types of functional interfaces.

- I/O interface
- Instruction interface
- Main storage interface
- I/O interrupt interface

### 3.2.1. I/O Interface

The I/O interface accommodates an 8-bit byte, plus parity, and is a compatible interface for all communications between the channel and subsystems. As many as eight subsystems may be connected to the channel by way of this interface.

### 3.2.2. Instruction Interface

The instruction interface enables the processor to issue I/O instructions to a selected channel and subsystem; the channel returns a condition code at the completion of the initiation of the operation. The processor begins the execution of an I/O instruction by transferring selected bits of the instruction opcode field, the contents of the instruction subsystem address field, and a request to process the instruction to the selected channel. Depending on the state of the channel, subchannel, and subsystem, the operation may or may not be initiated; the channel then returns the appropriate condition code and the signal acknowledging receipt of the instruction request signal.

### 3.2.3. Main Storage Interface

This interface enables a channel to request the use of main storage, and enables transfers between main storage and a selected channel. Each channel may independently request the use of main storage by sending a main storage request signal to the processor. Based on the storage priority scheme, the processor determines whether a channel is to be allowed use of storage and issues a signal to the selected channel. The selected channel retains use of main storage for one storage cycle. This interface also is used for the transfer of main storage error indications, such as addressing exception, protection exception, address check, and storage parity check from main storage and the processor to the channel.

### 3.2.4. I/O Interrupt Interface

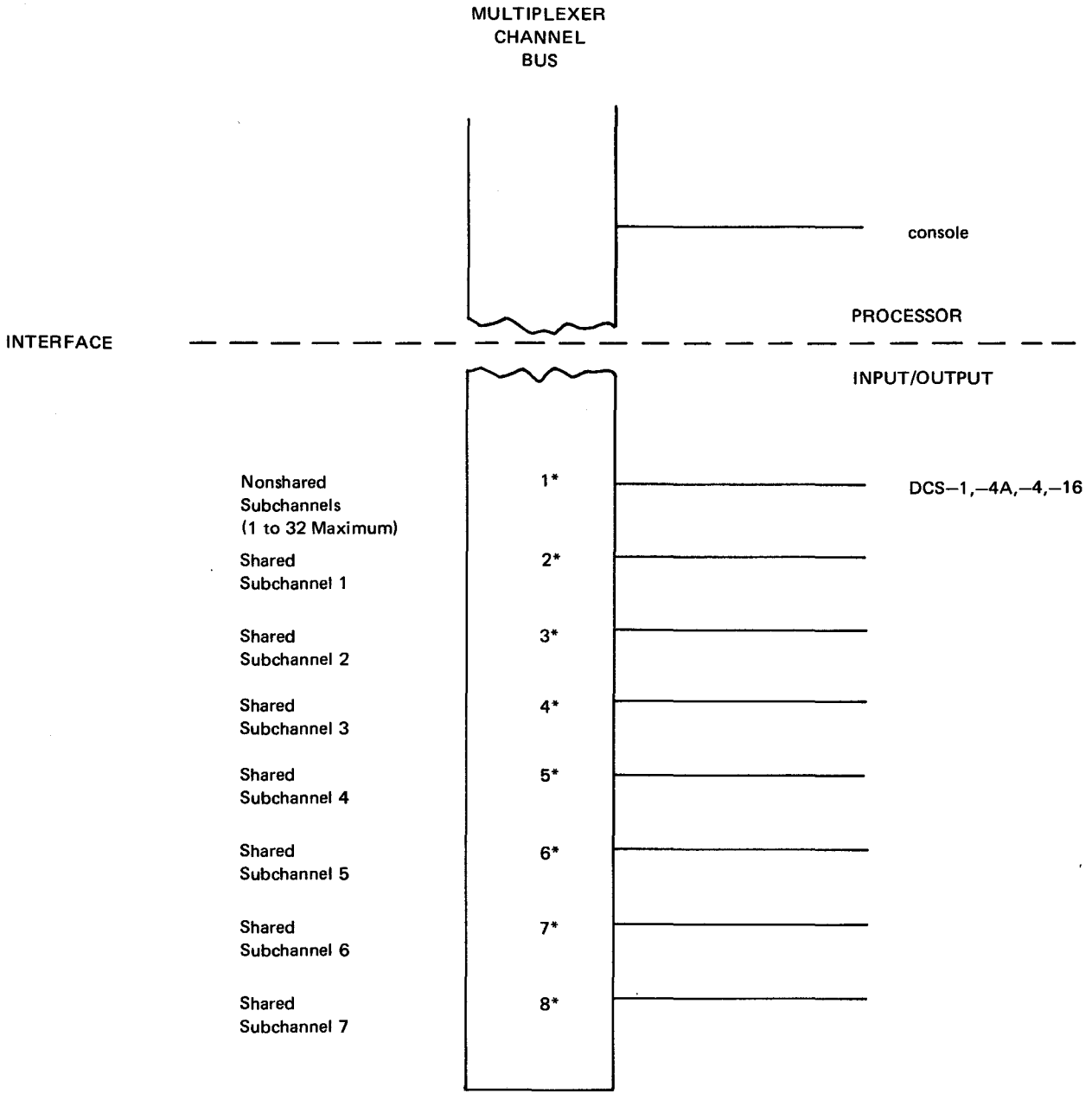
This interface enables a channel to request and to execute I/O interrupts. Each channel may independently request an I/O interface by sending an interrupt request signal to the processor. Based on the interrupt priority scheme, the processor determines whether a channel is to be allowed to interrupt and issues a priority signal to the selected channel. The selected channel then takes the appropriate action, depending on the interrupt level (priority).

## 3.3. MULTIPLEXER CHANNEL

The processor contains one multiplexer channel to operate low-speed peripheral devices, data communications subsystems, and up to 132 subchannels (128 nonshared subchannels plus four shared subchannels). It is capable of storing immediate status in the channel address word (CAW) area of low-order main storage, storing the ending status in the associated subchannel control word, or storing nonshared subchannel status in a defined table area. Also, it interrupts the processor when an error or ending status is stored. Physically, it is possible to connect up to eight peripheral subsystems to the multiplexer channel. However, at least one of these subsystems must be communications oriented. Because the console is an integral component of the processor and is connected before the interface driver circuits, it is not considered a physical drop and does not use one of the eight drops available.

### 3.3.1. Shared and Nonshared Subchannels

Functionally, it is possible to sustain concurrent operation on up to 8 shared subchannels and from 1 to 32 nonshared subchannels, depending upon the number of line terminals connected to a physical drop, such as drop 1 shown in Figure 3–1, and provided the maximum transfer rate of 79,000 bytes per second is not exceeded. Additional nonshared subchannels may be specified to a maximum number of 128. In a maximum configuration, the multiplexer channel could consist of, and would support concurrent operation on, 5 shared subchannels and 128 nonshared subchannels through the connection of 4 DCS-16's. Three DSC-16's are connected to physical drops normally reserved for shared subchannel operation. If a DCS-1 and/or a DCS-4A are specified, the number of shared and the number of nonshared subchannels decrease accordingly. (DCS equipment may be connected to a maximum of five physical drops normally reserved for shared subchannel operation. This is in addition to the connection to physical drop 1.)




*denotes physical drop on the multiplexer channel bus.

*Figure 3–1. Multiplexer Channel Simplified Physical and Functional Diagram*

A shared subchannel accommodates one or more subsystem devices connected to it through a common control, but only one device at a time can transfer data. Thus, the shared subchannel is defined as that logic circuitry required to sustain a data transfer between one device and main storage, by way of the control and arithmetic sections of the processor. A nonshared subchannel accommodates just one device (DCS) connected to it and may transfer data concurrently with other subchannels. All subchannel activity is directed by a buffer control word (BCW). These BCWs are restricted to the lower 128K bytes of main storage. The BCW addresses are specified by SCWs at fixed locations. Shared SCWs for shared subchannels are located in low-order main storage locations $100_{16}$ through $170_{16}$ (noncontiguous location). Nonshared SCWs for the nonshared subchannels are located at locations $200_{16}$ through $3FC_{16}$. The SCWs are specified by the device number provided by a peripheral control or by the b and d fields of the start-I/O (SIO) instruction. Subsystems connected to the multiplexer channel are addressed as follows:

| Address | (Binary Subsystem No.) | Subchannel |
|---|---|---|
| 1000 | 0000 | Shared subchannel 0 (System console only) |
| 1001 | xxxx | Shared subchannel 1 |
| 1010 | xxxx | Shared subchannel 2 |
| 1011 | xxxx | Shared subchannel 3 |
| 1100 | xxxx | Shared subchannel 4 |
| 1101 | xxxx | Shared subchannel 5 |
| 1110 | xxxx | Shared subchannel 6 |
| 1111 | xxxx | Shared subchannel 7 |
| 0000 to 0111 | 0000 to 1111 | 128 nonshared subchannels |

*NOTE:*

*x indicates that any binary subsystem number between 0 and 15 can exist on that particular shared subchannel.*

Table 3–1 summarizes the operational aspects of each subchannel type.

*Table 3–1. Operational Aspects of Each Multiplexer Subchannel Type*

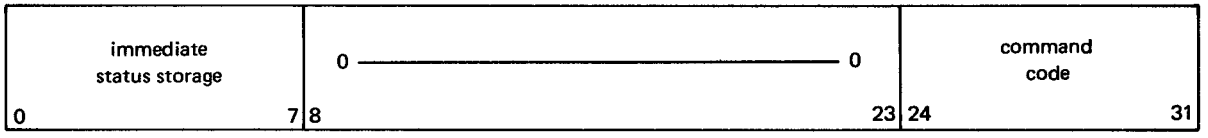
| | Shared Subchannels 1–7 | Nonshared Subchannel | Console or Timer or Status SCW |
|---|---|---|---|
| Shared subchannel control word | Yes | No | Yes |
| Nonshared subchannel control word | No | Yes | No |
| Shared buffer control word | Yes | No | No |
| Nonshared buffer control word | No | Yes | Yes |
| Data chaining | No | Yes | Yes |
| Transfer to non-contiguous BCW (link) | No | Yes | Yes* |
| Character parity check/ generation | No | Yes | No |
| Control byte compare | No | Yes | No |
| Buffer control interrupt | No | Yes | No |

*Console is designated to operate in the nonshared mode for test purposes only.

### 3.3.2. Channel Address Word (Multiplexer Channel)

The channel address word (CAW) is located in supervisor general register 0 during an SIO instruction. The format and bit functions of the CAW for both shared and nonshared subchannels are:

Format:




Bit functions:

immediate status storage (bits 0–7)

Provides status storage for immediate response or pending status on execution of an SIO instruction. Available status codes are:

| Status Codes | Meaning |
|---|---|
| 1000 0000 | Attention |
| 0100 0000 | Status modifier |
| 0010 0000 | Control unit end |
| 0001 0000 | Busy |
| 0000 1000 | Channel end |
| 0000 0100 | Device end |
| 0000 0010 | Unit check |
| 0000 0001 | Unit exception |

bits (8–23)

Unassigned and must be set to 0 by the software.

command code (bits 24–31)

Specifies operation to be performed. This 8-bit command is transmitted to the addressed device (3.4.2.1). The command must correspond to the d and w fields of the associated SCW (3.3.3). Available command codes are:

| Command Codes | Meaning |
|---|---|
| 0000 0000 | Test-I/O |
| DDDD DD01 | Write |
| DDDD DD10 | Read |
| DDDD 0100 | Sense |
| DDDD 1100 | Read-backward |
| DDDD DD11 | Control |

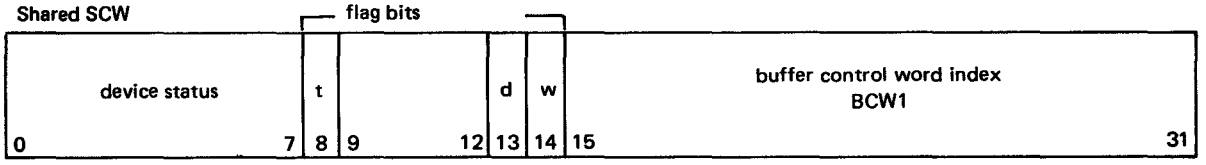
D = Variable detail bit, 1 or 0.

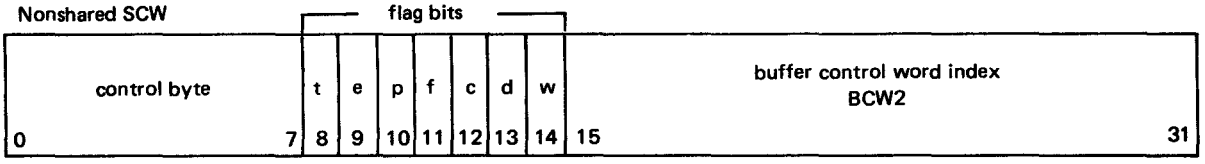
These commands are initiated by the channel. The least significant four bits are decoded by the channel to initiate the appropriate sequence.

### 3.3.3. Subchannel Control Word

The subchannel control word (SCW) contains flags and an index for locating the associated BCW for each type of subchannel. The upper eight bits of the SCW either hold a control byte or are used for device status storage, depending on whether the subchannel is nonshared or shared, respectively. The format and bit functions of the shared and nonshared subchannel control words are:

Format:




*NOTE:*

*The buffer control word index (BCWI) field of the SCW is only 17 bits, thus restricting the BCWs to be allocated only within the 128K bytes of main storage on full-word boundaries. Subchannel control words for nonshared subchannels are located in main storage starting at address of 512 (outside reserved low-order storage) when the optional communications system is utilized.*

Bits 9 through 12 for the shared SCW are ignored by the hardware.

Bit function:

device status (bits 0–7)

Device status storage for interrupts on shared subchannel. Control byte is associated with input and output on nonshared subchannel.

flags (bits 8–14)

t (bit 8)

Terminate flag – upon completion of the I/O operation using the BCW, the channel sets the t flag to 1 when either a stop operation code ($00_2$) or a stop immediate code ($11_2$) is contained in the current BCW. Normal completion of a BCW is caused by:

- the byte count decremented from 1 to 0, or
- the control byte compares equally to the input data byte being transferred.

When the t flag is 1, it causes the channel to terminate the device when the next service request is issued by the subsystem.

e (bit 9)

Parity error flag is set to 1 when the communications adapter detects a character parity error on a nonshared subchannel. The operation is not terminated and the parity bit of the data byte is set to 1. In a normal input operation, bit 0 of the data byte is set to 0 by the channel before it is stored. The e flag is ignored by the hardware for shared subchannels.

p (bit 10)

Character parity flag – when set to 1, the seven low-order data bits are used to check the data byte parity and for generation of output parity in data bit position 0. When set to 0, no character parity operation occurs. The p flag is ignored by the hardware for shared subchannels.

f (bit 11)

Odd parity flag – when set to 1, the 7-level special character parity check (input) or parity generation (output) is performed for odd parity.

When set to 0, the parity check or parity generation is performed for even parity. The f flag is ignored by the hardware for shared subchannels.

c (bit 12)

Control byte compare flag – when set to 1, causes the channel to compare the data byte with the control byte. If the two bytes are equal, an interrupt is generated and subchannel status is stored. The operation indicated by the operation code in the current BCW (stop, chain, link, or stop immediate) is executed. The c flag is ignored by the hardware for shared subchannels.

d (bit 13)

Decrement flag – when set to 1, the data address is decremented by 1 for each data byte transferred. When set to 0, the data address is incremented by 1 for each data byte transferred. (In the case of nonshared status SCW, the address is changed by plus or minus 4.)

w (bit 14)

Write output flag – when set to 1, specifies an output operation to the channel; when set to 0, specifies input operation to the channel.

buffer control word index – BCW1, BCW2 (bits 15–31)

Holds the address of the current BCW (must be located at a full-word boundary).

### 3.3.3.1. Parity Generation and Checking

When the p field (bit 10 of the nonshared SCW) is set to 1, the channel generates parity on output and checks parity on input. Parity is the eighth-level bit (bit 0). The f field (bit 11 of nonshared SCW) specifies odd or even parity generation or checking on the data transfer.

If both the p and f fields are set to 1, odd parity is generated on the seven data bits transferred during the output operation or checked on those transferred during the input operation. If the p field is set to 1 and the f field is set to 0, even parity is generated on the low-order seven data bits transferred during the output operation or checked on those transferred during the input operation. During the input operation, with character parity check, the eighth level of each character is normally forced to 0 by the channel before it is stored (most significant bit). Detection of a character parity error during an input data transfer causes the e field of the SCW and bit 0 of the data byte to be set to 1.

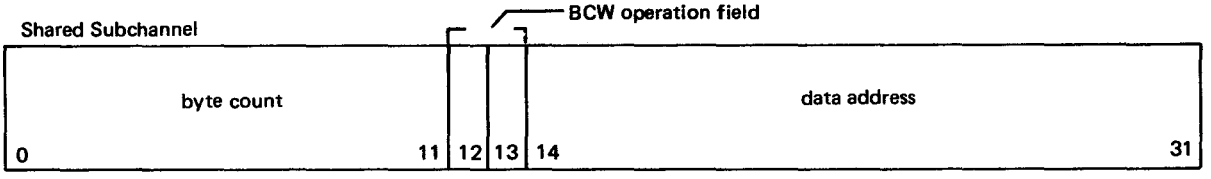
### 3.3.3.2. Special Character Recognition

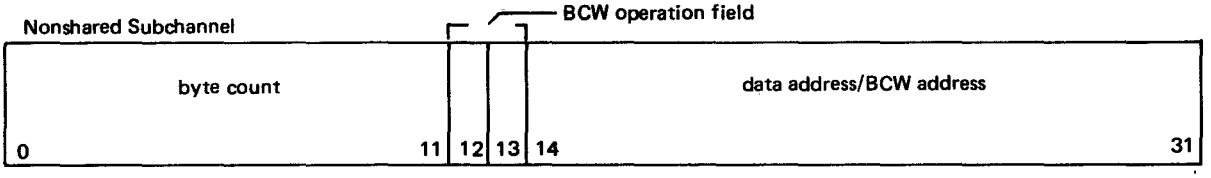
If the c field (bit 12 of the nonshared SCW) is set to 1, each byte of data transferred by the subchannel is compared to the control byte of the nonshared SCW. When equality is found, a control byte status is tabled and an interrupt request is generated. The BCW operation executed by the channel, following character recognition (stop, chain) is defined by bits 13 and 14 of the current BCW. Special character recognition operates on both input and output.

## 3.3.4. Buffer Control Word

The buffer control word (BCW) associated with each subchannel can be located anywhere in the lower 128K bytes of main storage other than reserved low-order storage and must be located on full-word boundaries. The BCW contains a byte count field, a BCW operation field, and a data address field. The data address field acts as a new BCW address when a link BCW operation occurs on the nonshared subchannel. The following two formats define the BCW for the shared and nonshared subchannels. The format and bit functions of the BCW are:

Format:




Bit functions:

byte count (bits 0–11)

A 12-bit field allowing a transfer count up to and including 4096 bytes.

BCW operation field (bits 12, 13)

Except for the console, must be set to 0 by software for shared subchannel operation. For nonshared subchannel:

- $00_2$ stop (terminates SCW when count = 0)
- $01_2$ chain to next contiguous BCW
- $10_2$ link to noncontiguous BCW (address contained in low-order 17 bits of the SCW)
- $11_2$ stop immediate (terminates SCW immediately)

data address/BCW address (bits 14–31)

Contains data address when used for a shared subchannel. For nonshared subchannel use, the contents are either a data address or a new BCW address.

### 3.3.5. Status Handling

When I/O status conditions arise, either from the channel (internal) or the control units (external), they are handled as either start I/O status, nonshared subchannel status, or shared subchannel status.

#### 3.3.5.1. Start I/O Status

If the pending or busy status conditions exist during the execution of an I/O instruction on a shared or nonshared subchannel, the device status byte is stored in the high-order eight bits of the CAW, and the PSW condition code is set to 1; no interrupt sequence is initiated. If an immediate operation is executed, the channel stores the device status in the high-order eight bits of the CAW and sets the condition code to 1; again no interrupt sequence is initiated. An operation is called immediate when the I/O device immediately signals the channel end condition upon receipt of the command code.

#### 3.3.5.2. Nonshared Subchannel Status

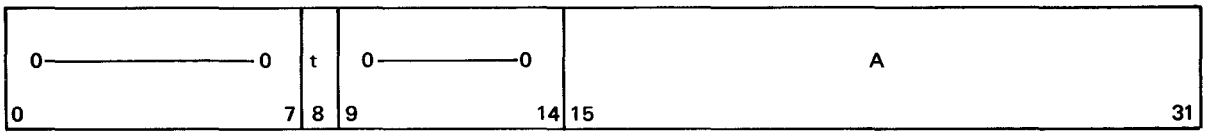
All other status conditions on the nonshared subchannel cause a full interrupt status word to be stored at a location specified by the address field of the current BCW specified by the status table SCW. An interrupt request is generated and the actual interruption occurs as permitted by the system mask and interrupt priority. The amount of storage allocated for interrupt tabling is variable and is controlled by the byte count contained in the status table BCWs. The storage of interrupt status for nonshared subchannels is the same as for the storage of data (chaining and linking included), with the following exceptions:

- the buffer control interrupt is not generated when the byte count field is zero;
- the address field of the BCW is incremented by 4 upon the storage of each status word; and
- the count field is decremented by 4.

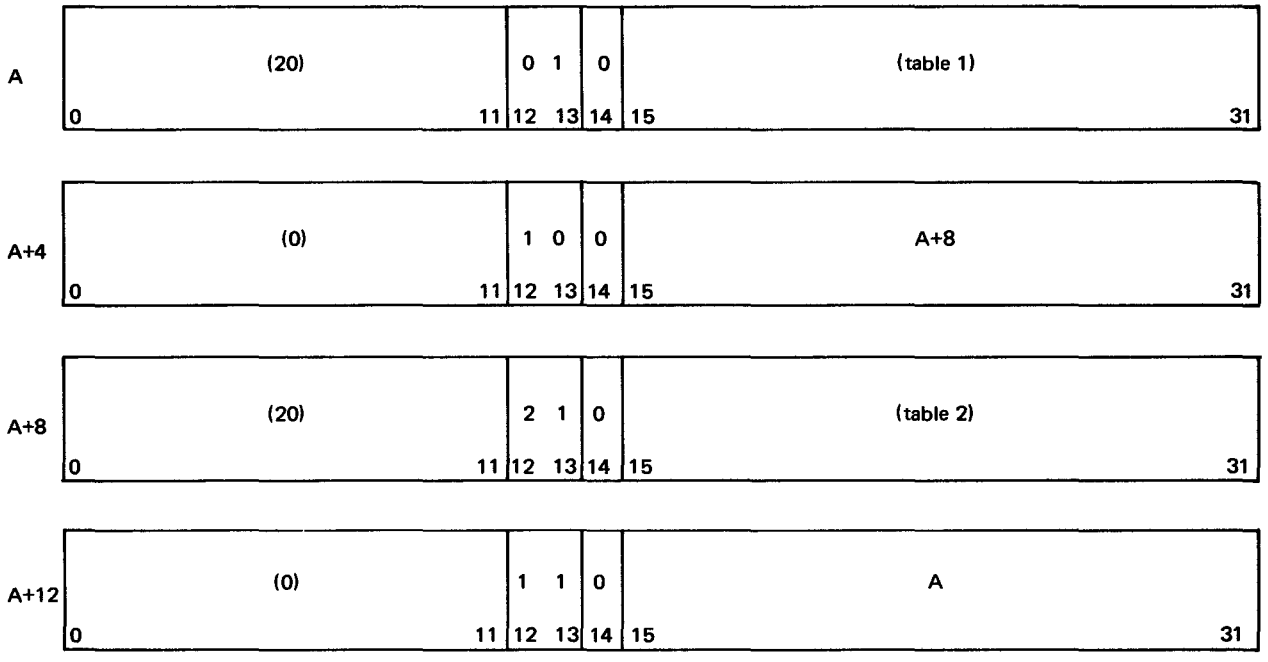
The SCW and associated BCW for status storage are formatted similarly and are programmed the same as the shared data SCWs and associated nonshared BCWs. No start-I/O instruction is required to activate the status table SCW.

It should be noted that inasmuch as a buffer control interrupt is not generated upon exhaustion of a BCW count field, and prestore checks are not made upon the status table contents, it is the responsibility of the software to ensure status table integrity. This is performed by a BCW operation, such as:

**STATUS SCW**




**STATUS BCWs**




| | | | |
|---|---|---|---|
| STATUS TABLE 1 | STATUS | 1 | 11 |
| | | 2 | 12 |
| | | 3 | 13 |
| | | 4 | 14 |
| | | 5 | 15 |
| STATUS TABLE 2 | | 6 | 16 |
| | | 7 | 17 |
| | | 8 | 18 |
| | | 9 | 19 |
| | | 10 | 20 |

The status BCWs define the status storage as:

BCW 1 at A
: Specifies store five entries and chain to next contiguous BCW.

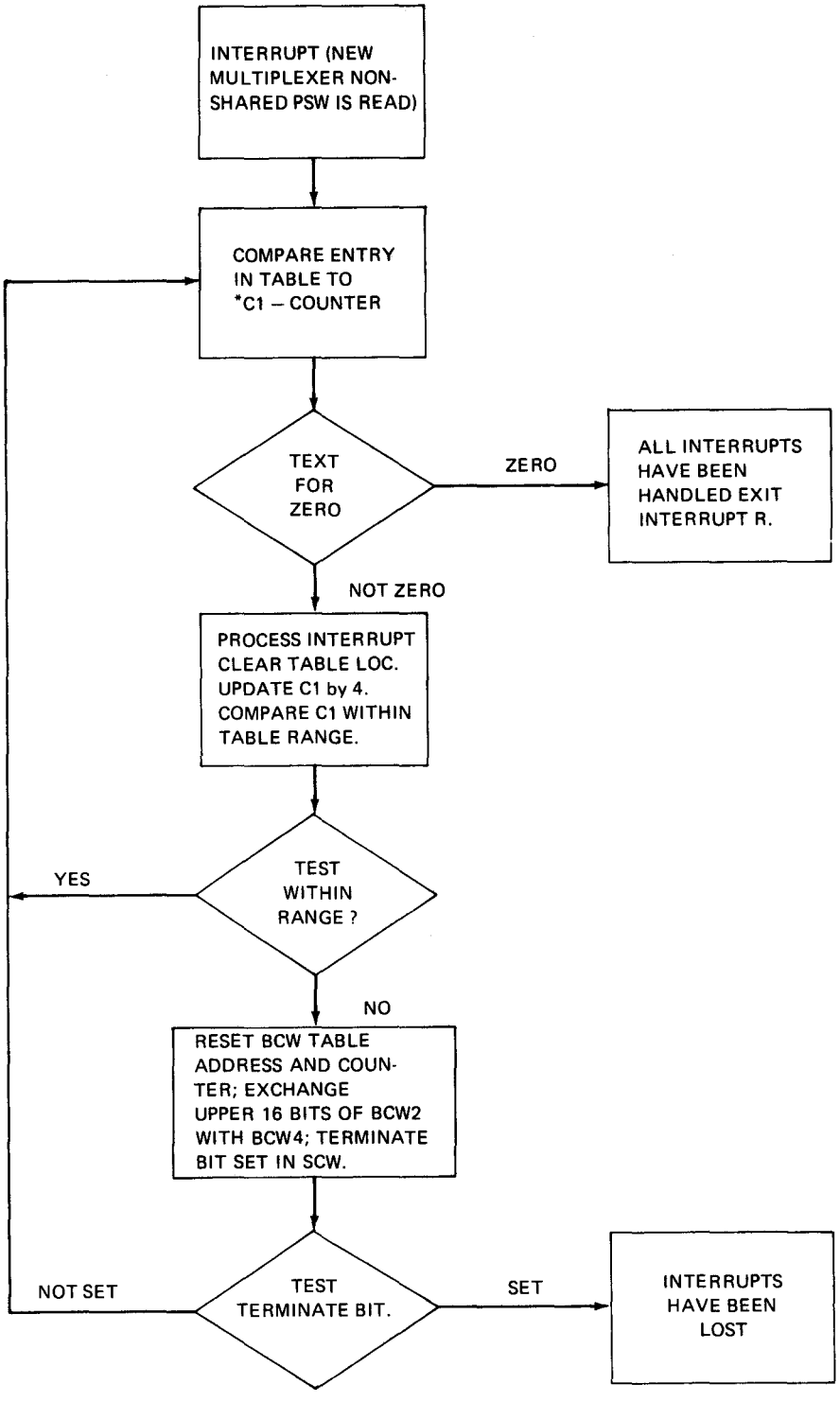
BCW2 at A + 4
: Specifies replace BCW address field of the SCW with the address A + 8 (link).

BCW3 at A + 8

Specifies store five entries and chain to next contiguous BCW.

BCW at A + 12

Specifies stop immediate. Set t bit to 1 in SCW. No more status words are stored. The program control of the BCW is shown in Figure 3–2.




*CI is a software counter initially set to status table 1 address.

*Figure 3–2. Program Control of BCWs*

With receipt of each service request, the BCW operation code is examined for $11_2$ (stop immediate) and $10_2$ (link). The stop immediate causes the t field (bit 8, SCW) to be set to 1 and issues a terminate signal to the subsystem for that request.

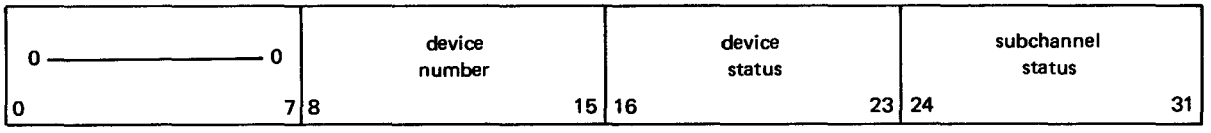
The link operation causes replacement of the current BCW address in the SCW with the new BCW address. The present service request and subsequent requests are processed according to the new BCW. Two conditions cause examination of the BCW operation code for $01_2$ (chain) or $00_2$ (stop):

- The control byte compare operation has established equality. This operation is valid for data transfer, not status transfer.
- The byte count field in the current BCW has gone from 1 to 0.

The stop code ($00_2$) causes the t field to be set to 1 in the SCW. Any subsequent data requests cause a terminate signal to be sent to the subsystem. The chain code ($01_2$) causes the BCW address to be incremented by 4 in the SCW. This represents a chain to the next contiguous BCW. The format and bit functions of the information word are:

Format:

**STATUS TABLE INFORMATION**




Bit functions:

bits 0–7
: Unassigned and must be set to 0 by the software.

device number (bits 8–15)
: Provides the device number of the external status source.

device status (bits 16–23)

bit 16 (attention)
: Indicates a condition not associated with the initiation, execution, or termination of an I/O operation. Set to 0 for nonshared channels.

bit 17 (status modifier)
: When set with busy bit (bit 19), distinguishes between control unit busy and device busy signals in multiple-device subsystems. When set with DEVICE END signal, indicates special ending condition for command chaining operations (skip designation). Set to 0 for nonshared channels.

bit 18 (control unit end)
: Indicates control is free for another operation. Can accompany CHANNEL END or DEVICE END signals or it can occur at a time in between if the control unit is interrogated by the initiation of another command. Set to 0 for nonshared channels.

bit 19 (busy)
: Indicates a subsystem control unit or control cannot execute a command because a previous command is in progress or a pending status condition exists.

bit 20 (channel end)
Indicates the channel is ready for a new operation. The time of occurrence of CHANNEL END signal is dependent upon the subsystem.

bit 21 (device end)
Indicates device in subsystem is ready for a new operation. If an unusual condition is detected during the initiation of a command, error status is presented without the DEVICE END.

bit 22 (unit check)
Indicates errors detected during operation. UNIT CHECK signal is a summary notification of a unit-associated malfunction. Detailed status can be obtained by initiating a sense command.

bit 23 (unit exception)
Indicates an abnormal condition that may or may not be an error. Its meaning is applicable to the type of command and subsystem device involved in the operation.

subchannel status (bits 24–26)

bit 24 (buffer control interrupt, BCI)
Indicates a multiplexer nonshared subchannel buffer termination resulting from the byte count being decremented from 1 to 0.

bit 25 (control byte interrupt, CBI)
Indicates the flag bit c of the nonshared SCW is 1 and that the control byte in this SCW is equal to the input or output data byte being transferred.

bits 26–31
Unassigned and must be set to 0.

### 3.3.5.3. Shared Subchannel Status

The status conditions that may occur on shared subchannels cause the channel to store the device status in the high-order eight bits of the SCW. The device number is stored in the lower half of the interrupt code field in the old PSW location. If interrupts are masked, or if a second interruption condition occurs on the subchannel before the first is cleared (interrupt processed), a stacked status response is initiated (causes COMMAND OUT signal instead of SERVICE OUT in response to STATUS IN signal) and suppresses service requests for status. No subchannel status is associated with the shared subchannel.

## 3.4. SELECTOR CHANNEL

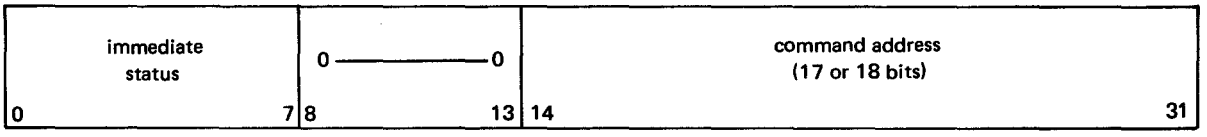
A maximum of two selector channels is available. Each can control up to eight high-speed subsystems, in burst mode, with a maximum of 16 devices attached. The selector channel:

- stores status in a reserved storage location called the channel status word (CSW);
- interrupts the processor when error or ending status is stored;
- executes a command chain sequence;
- operates as an independent transfer mechanism, queueing with the processor at the storage access interface; and
- utilizes a CAW and channel command word (CCW) for control of an addressed I/O device.

### 3.4.1. Channel Address Word (Selector Channel)

The channel address word (CAW) is used during the execution of a start-I/O instruction to point to the location of the first CCW. The CAW is located in supervisor general register 0. The format and bit functions of the CAW are:

Format:




Bit functions:

immediate status (bits 0–7)
: Contains storage of status for immediate response or pending status on execution of start-I/O instruction.

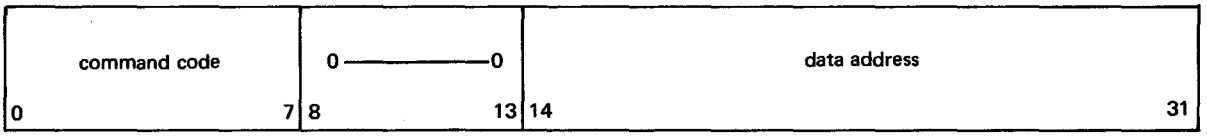
bits 8–13
: Unassigned and must be set by the software.

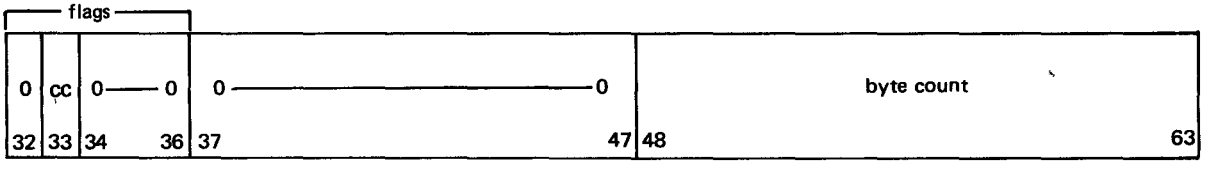
command address (bits 14–31 or 15–31)
: Specifies the location of the first CCW on a double-word boundary. For main storage less than 128K, bit 14 is always set to 0.

### 3.4.2. Channel Command Word (CCW)

The CCW is used during the execution of a start-I/O instruction or command chain sequence to specify control information needed for operating the addressed I/O device. The format and bit functions of the CCW are:

Format:




Bit functions:

command code (bits 0–7)
: Specifies commands that the channel can initiate (3.4.2.1).

bits 8–13, 37–47
: Unassigned and must be set to 0 by the software.

data address (bits 14–31)
Specifies storage address of the first data byte.

flags (bits 32–36)
bits 32, 34–36
Unassigned and must be set to 0 by the software.

cc (bit 33)
Command chain flag – when set to 1 specifies command chaining.

byte count (bits 48–63)
Specifies the number of data bytes to be transferred (65,536 maximum).

### 3.4.2.1. Channel Command Codes

The command codes that the selector channel can initiate are listed and defined as follows. The low-order four bits are decoded by the channel to initiate the appropriate sequence. The x indicates the bit can be either 0 or 1.

| Command | Code | |
|---|---|---|
| Test-I/O | xxxx | 0000 |
| Write | xxxx | xx01 |
| Read | xxxx | xx10 |
| Sense | xxxx | 0100 |
| Transfer-in-channel (TIC) | 0000 | 1000 |
| Read-backward | xxxx | 1100 |
| Control | xxxx | xx11 |

- Test-I/O

  The test-I/O (TIO) command is sent to the device and the resulting status is stored in supervisor general register 0.

- Write

  A write command is sent to the subsystem and the subchannel is enabled to transfer data from storage to the subsystem device when requested.

- Read

  A read command is sent to the subsystem and the subchannel is prepared to transfer data to storage when the subsystem device requests service.

- Sense

  A sense command is sent to the subsystem and the subchannel is prepared to transfer sense bytes from the subsystem device to storage.

- Transfer-in-channel

  The next CCW is fetched from the location designated by the data address field of the CCW specifying transfer-in-channel (TIC) command. The TIC does not initiate an I/O operation at the channel.

- Read-backward

  A read-backward command is sent to the subsystem and the subchannel is enabled to transfer data from the subsystem device to storage. The accessed channel places the bytes in storage in a descending order.

- Control

  A control command is sent to the device and the subchannel is set up to transfer data from storage to the device when the device requests data.

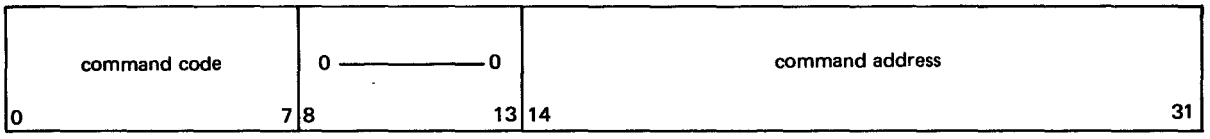
  *NOTE:*

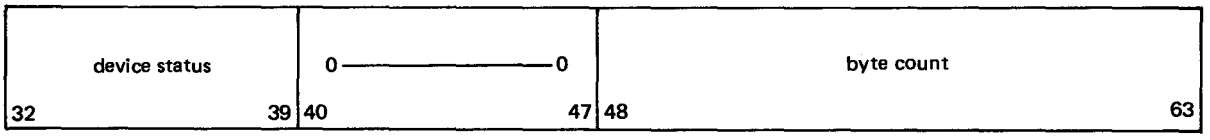
  *The multiplexer channel does not operate with CCWs. It is therefore necessary for the program to establish the SCWs and BCWs associated with the subchannel for the SIO instruction prior to execution of SIO. The SIO on the multiplexer channel sends the command to the device, stores status when appropriate, and sets the condition code (3.6).*

### 3.4.3. Channel Status Word

A double word is used to store interrupt status information for the selector channels. These words are located in reserved low-order storage at addresses $1A0_{16}$ through $1A7_{16}$ and $1A8_{16}$ through $1AF_{16}$ for selector channels 1 and 2, respectively. The format and bit functions of the channel status word (CSW) are:

Format:




Bit functions:

command code (bits 0–7)
: Indicates a command code was executed last.

bits 8–13, 40–47
: Unassigned and is set to 0 by the hardware.

command address (bits 14–31)
: Specifies address pointing to location of next CCW.

device status (bits 32–39)
bit 32 (attention)
bit 33 (status modifier)
bit 34 (control unit end)
bit 35 (busy)
bit 36 (channel end)
bit 37 (device end)
bit 38 (unit check)
bit 39 (unit exception)

*NOTE:*

*See 2.2.3.2 for descriptions of the individual bits.*

byte count (bits 48–63)
Specifies original CCW byte count decremented to some value. When subtracted from the original byte count, it specifies the number of bytes transferred.

### 3.4.4. Command Chaining

The selector channel initiates a command chain sequence if:

- a device end status is received from the device;
- a command chain control field (bit 33) has been set to 1 in the CCW; or
- no error conditions have occurred as a result of the previous operation.

The channel reads contiguous CCWs executing the specified commands. All commands, except for the TIC, are transferred to the device. The device responds by presenting appropriate status. The stored status information modifies the chaining operation as follows:

- Unit check, absence of command chain bit, or unit exception status terminates the chaining operation
- Status modifier, with the device end status, causes the channel to skip the next contiguous CCW specified by the command address. The channel takes the next (the CCW following the skipped CCW) contiguous CCW and performs the indicated operation.
- Channel end status without device end status causes the channel to indicate command chaining to the subsystem device. A multidevice subsystem must use this indication to link the appropriate device end status to this channel end status.
- Device end status, or channel end status and device end status, causes the channel to read the next contiguous CCW specified by the command address.

If a TIC command is decoded during a chain operation, the channel does not transfer this command to the subsystem device. The channel reads another CCW using the data address (read along with the TIC command), instead of the command address, to locate the CCW. The channel performs the indicated operation and continues to chain if directed. Test commands are invalid in command chaining because if an interrupt occurs, chaining is terminated.

### 3.4.5. Status Handling

#### 3.4.5.1. Start-I/O Status

Upon initiation of a start-I/O instruction, various conditions can arise, such as:

- Normal completion

  Upon acceptance of the command by the subsystem, the operation in process is indicated by a condition code value of $00_2$ and no status is stored.

- Status stored

  Status is stored and the condition code value is set to $01_2$ when a subsystem device has pending status, immediate operation is not specified, or the subsystem responds with immediate error status.

- Busy

  If the channel is busy, the condition code is set to $10_2$ and status is not stored. The channel is busy from initiation of an SIO instruction until a channel end status is received from the control unit.

- Not operational

  The condition code is set to $11_2$ and status is not stored if the subsystem does not respond to the selection sequence. This condition occurs when the channel is not installed, the subsystem is not installed, the subsystem is offline, or the subsystem power is off.

#### 3.4.5.2. Other Channel Status

The error status and ending status (when command chaining is not indicated) cause the channel to store the status in the CSW in location $1A0_{16}$ through $1A7_{16}$ or $1A8_{16}$ through $1AF_{16}$, and initiate and interrupt request. If interrupts are masked or a second interrupt condition occurs at the I/O interface before the previous interrupt is processed, the accessed channel initiates a stack status response (provide COMMAND OUT signal instead of SERVICE OUT signal in response to STATUS IN signal). This causes the control unit to hold the status. Ending status is not stored in the CSW when command chaining is indicated. This status causes the channel to execute sequences as specified in 2.4.4.

## 3.5. COMMUNICATIONS ADAPTER

The optional feature communications adapter provides up to 128 nonshared subchannels for servicing commucations line terminals. The program-controlled options provided for each subchannel facilitate the handling of a large number of different communications line terminals. Each simplex line terminal requires one subchannel. The program-controlled options available for each subchannel, independent of other subchannels are:

- Data chaining and linking

  Automatic chaining of one or more contiguous or noncontiguous BCWs in main storage. Each BCW controls a variable-length buffer area.

- Character parity specification

  Odd or even parity checking on input transfers and parity generation on output transfers.

- Control character specification

  Continuous monitoring of input and output data for program-specified control bytes.

- Buffer terminating interruptions

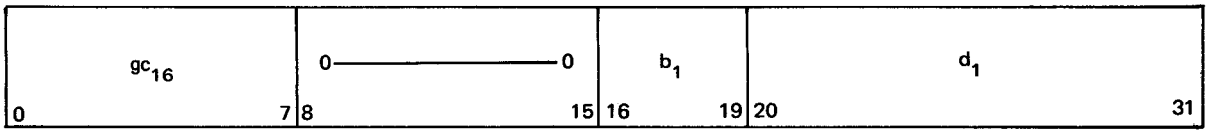
  – Buffer control interrupt, which occurs when the last character of a buffer has been stored or fetched (count decrements from 1 to 0).

  – Control byte interrupt, which occurs when the control byte and data byte are identical (input or output), and the e bit in the SCW is set to 1.

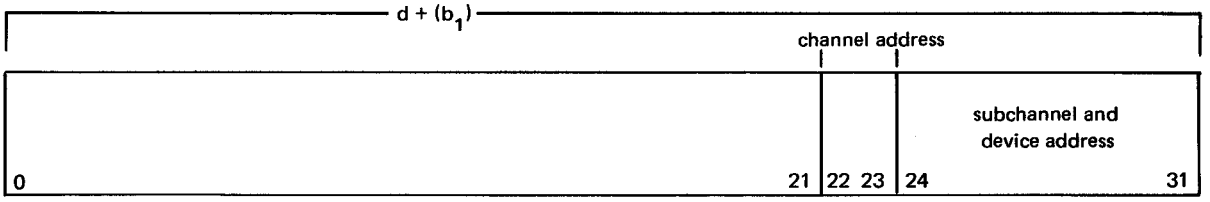
## 3.6. START-I/O INSTRUCTION

One instruction initiates activity on either the multiplexer channel or selector channel: the start-I/O (SIO) instruction. The format and bit functions of the SIO instruction are:

Format:




The channel, subchannel, and device address to which the instruction applies are specified by the low-order 10 bits of the sum of the contents of the register designated by the $b_1$ field added to the $d_1$ field. The CAW is in supervisor general register 0.




If the SIO instruction is for a selector channel, the low-order 18 bits of the CAW contain the address of the first CCW. For the SIO instruction, the high-order eight bits of the CAW always are used for storage of immediate, busy, or pending status. This instruction is initiated on the multiplexer channel after the channel has serviced all outstanding requests for data transfers. Completion of the SIO instruction sets the condition code (bits 34–35) in the PSW (2.2.1) as follows:

| Condition Code | Meaning |
|---|---|
| $00_2$ | Normal completion |
| $01_2$ | Status stored |
| $10_2$ | Channel busy (selector channel only) |
| $11_2$ | Not operational |

## 3.7. INITIAL LOADING

A certain amount of information must be loaded initially into storage before the system can be completely operative under program control. A total of 64K bytes (selector channel) or 4K bytes (multiplexer channel) may be loaded into storage, beginning at address 000. The loading procedure for each channel is identical with one exception: When loading through the multiplexer channel, care must be taken to make certain that the SCW and BCW applicable to the load operation are properly overlaid; that is, the appropriate SCW at $1n0_{16}$ through $1n3_{16}$, where n is equal to or greater than 1 and equal to or less than 7 ($1 \leqslant n \leqslant 7$) and the BCW at $114_{16}$ through $117_{16}$.

Initial loading is performed at the system console by selecting the channel and device number and pressing the LOAD switch. After transferring the initial load block, a device end status response initiates program control.

### 3.7.1. Initial Loading at Multiplexer Channel

Initial loading at the multiplexer channel can be performed by using a subsystem such as a magnetic tape unit connected to one of the shared subchannels. The initial load operation is monitored by an SCW and an associated BCW in low-order main storage. The channel stores the SCW at the location specified by the LOAD switches, and the BCW is stored at address $114_{16}$. The SCW contains 0's in bit positions 0 through 14; the BCWI field is set to $0114_{16}$. The BCW initially is set to 0 to permit 4,096 bytes to be placed into main storage from starting address 000. Inasmuch as the SCW and BCW locations are overwritten during the initial load process, the program must not specify entries at these locations. Also, SCW1 through SCW7 locations, respectively should be set to $00000114_{16}$, and the BCW byte count field should be set to $EEA0_{16}$ if the initial load block size is 4096 bytes. This number may be smaller for a proportionately smaller block size. The BCW data address field should be set to $0118_{16}$ for loading the next byte into the next contiguous location. Addresses $101_{16}$, $181_{16}$, and $191_{16}$ should be set to $80_{16}$ to condition the system console SCW, status table SCW, and TCW, respectively, so that these subchannels do not operate upon completion of the initial load operation. Location $000_{16}$ is used for storing immediate status indication. It is conditioned by the hardware if nonzero status indication is received at the beginning of the initial load operation. Location $000_{16}$ through $003_{16}$ should be set to 0.

The program must provide a new multiplexer-channel-shared PSW, as an interruption occurs upon completion of the initial load operation. This could be 0000 0000 0000 $0400_{16}$ with $0400_{16}$ in the least significant two bytes; the first instruction in the program is fetched at this address. Other addresses may be specified; however, the new multiplexer-channel-nonshared PSW must contain $000104_{16}$ in the least significant three bytes and the halt-and-proceed instruction should be placed into location $104_{16}$ to stop the processor if an erroneous interruption occurs after the initial load operation is completed.

The old multiplexer-channel-shared PSW is changed by the hardware when the interruption occurs. The program should load these locations with 0000 FFFF 0000 $FFFF_{16}$.

### 3.7.2. Initial Loading at Selector Channel

Initial loading at the selector channel can be performed by using subsystems such as magnetic tapes or discs. The operation is similar to the multiplexer channel loading with the following exceptions and considerations.

- When performed on a selector channel, the initial load operation is monitored by a CCW in the channel hardware and a CSW in low-order storage at address $1A0_{16}$ (for selector channel 1) or $1A8_{16}$ (for selector channel 2). The program should set all locations to 0. The contents of the appropriate CSW are changed by the channel upon completion of the initial load operation.

- The program must provide a new selector channel 1 (or 2) PSW. This could be 0000 0000 0000 $0400_{16}$ for the new multiplexer-channel-shared PSW. The old selector channel PSW should have a pattern similar to the old multiplexer-channel-shared PSW.

- If different addresses are specified in the new PSWs for the selector channels and the multiplexer channel, the address to which the program transfers after the interruption indicates which channel was used in the initial load operation.

## 3.8. I/O PRIORITY

### 3.8.1. Selector Channel

The selector channels hold all control information in registers except the updated CCW address, which is held in the CSW during the I/O operation. This channel essentially functions as one subchannel on which eight possible subsystems can operate serially. The selector channels access storage directly, with selector channel 2 as the highest priority channel.

### 3.8.2. Multiplexer Channel

The multiplexer channel holds all SCW and BCW information in storage. The channel is time-shared by all subchannels on the multiplexer channel.

The updated BCW information is read by the channel (byte count and address), updated, and stored again for each data byte transferred to or from storage. The channel can carry out this process concurrently on a maximum of eight shared subchannels, 128 nonshared subchannels, or some combination of both shared and nonshared subchannels.

Three levels of conflict could occur during multiplexer channel operations:

- The first could occur at the I/O interface that is time-shared by all the assigned subchannels. The priority of activity at this level is defined by the cable routing of the SELECT OUT signal through the subsystem.
- The second could occur at the common arithmetic unit, time-shared by the processor and multiplexer channel.
- The third could occur at the common main storage interface, time-shared by the selector channels and the combined multiplexer and processor operation.

I/O priority assignments that resolve these conflicts are:

| | Channel | Priority |
|---|---|---|
| Selector channel 2 | 1 | highest |
| Selector channel 1 | 2 | ↓ |
| Multiplexer channel | 3 | |
| Processor | 4 | lowest |

## 3.9. I/O LOADING AND TIMING

### 3.9.1. Loading Considerations

The loading relationship of any system is defined by the way the conflicts (two or more elements time-sharing some common element) are handled by the system. A definite queueing and priority scheme is implemented in channel control.

### 3.9.2. Timing Considerations

Timing and data transfer rates are dependent on:

- conflicting activities at all levels;
- priority assignment;
- number of storage references per sequence;
- interface delays (including cable lengths); and
- subsystem response times.

The selector and multiplexer channels are capable of transferring data at maximum rates of 333,000 bytes and $79{,}000_5$ bytes per second (85,000 bytes per second if an ideal instruction mix is used), respectively. These rates are achieved by limiting the conflicting activities at all levels. The data transfer rate of the multiplexer channel is degraded when one or two selector channels are operating concurrently with it (Table 3–2).

*Table 3–2. Multiplexer Channel Data Transferring Capability (Bytes per Second)*

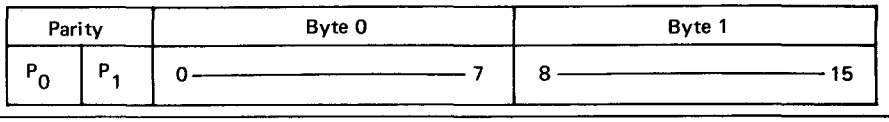
| Storage Reference Time (nanoseconds) | Data Transfer Rate in Bytes per Second | | |
|---|---|---|---|
| | Number of Selector Channels | | |
| | 0 | 1 | 2 |
| 600 | 79,000 | 71,000 | 64,000 |
| 750 | 63,000 | 55,000 | 49,000 |

# 4. Main Storage

## 4.1. MAIN STORAGE CHARACTERISTICS

Main storage characteristics of the UNIVAC 9400 System and the UNIVAC 9480 System are similar, with the exception of the type of hardware used, storage capacities, and cabinet configurations. The major hardware and capacitance differences are listed in Table 4–1.

*Table 4–1. Main Storage Differences*

| | UNIVAC 9400 System | UNIVAC 9480 System |
|---|---|---|
| Hardware | Plated wire | Semiconductor |
| Minimum capacity | 32,768 bytes | 65,536 bytes |
| Maximum optional expandable capacity | 262,144 bytes | 262,144 bytes |
| Expandable incremental capacity | 16,384-byte increments up to 64,536 bytes; 32,768-byte increments between 64,536 and 131,072 bytes; 64,536-byte increments over 131,072 bytes of main storage. | 32,768-byte increments up to 131,072 bytes; 64,536-byte increments over 131,072 bytes of main storage. |
| Cycle time (four bytes) | 600 nanoseconds | 600 nanoseconds with a 2% degradation due to refresh cycle required for semiconductor type of storage. |
| Storage | Nondestructive | Destructive (volatile, must be refreshed or restored) |
| Data Format |  | |

The smallest addressable unit of storage is one byte, which is eight binary bits and a parity bit. Bytes are numbered consecutively from 0 to a maximum of 262,143. These numbers are the byte addresses.

Storage functions are an integral part of the processor even though housed in one or more separate cabinets. For the UNIVAC 9400 System, three cabinets are required to house the maximum of 256K bytes of plated wire main storage. One cabinet houses the necessary power supplies; each of the two remaining cabinets houses up to 128K bytes of main storage.

For the UNIVAC 9480 System, a maximum of two cabinets is required, each containing its own power supply and with up to 256K bytes of semiconductor type main storage.

### 4.1.1. Addressing

Byte locations are numbered consecutively, beginning with 0, and are addressed sequentially. A group of locations (two or more bytes) is referenced by the address of the most significant byte location. The addressing hardware of the processor accommodates an 18-bit number that allows access to 262,144 maximum locations.

## 4.2. INFORMATION POSITIONING

Inasmuch as main storage locations are addressed consecutively, the bits in an information byte are numbered from left to right starting with 0, ending with 7. A half word consists of two consecutive bytes; a full word, four consecutive bytes; a double word, eight consecutive bytes.

Fixed-length fields, such as half words and full words, are on integral boundaries and must be loaded into main storage so that the address is evenly divisible by the field length in bytes. Therefore, a half word must have an address that is a multiple of 2, and a full word must have an address that is a multiple of 4. The binary address of these fields must contain 0's in the low-order bit positions. Variable-length data fields are not restricted by their boundaries. Instructions must begin on half-word boundaries and must have lengths of 2, 4, or 6 bytes.

## 4.3. FIXED STORAGE ASSIGNMENTS

Although most of main storage is for storing instructions, operands, and data, the first 512 byte locations are reserved for special functions and are protected from reading or writing when the processor is in the problem state (1024 locations when specifying the unprotected area that holds the subchannel control words for the multiplexer nonshared subchannels). The functions assigned to these locations are listed in Table 4–2 with a brief explanation of their use. Figure 4–1 shows the physical arrangement for fixed low-order main storage.

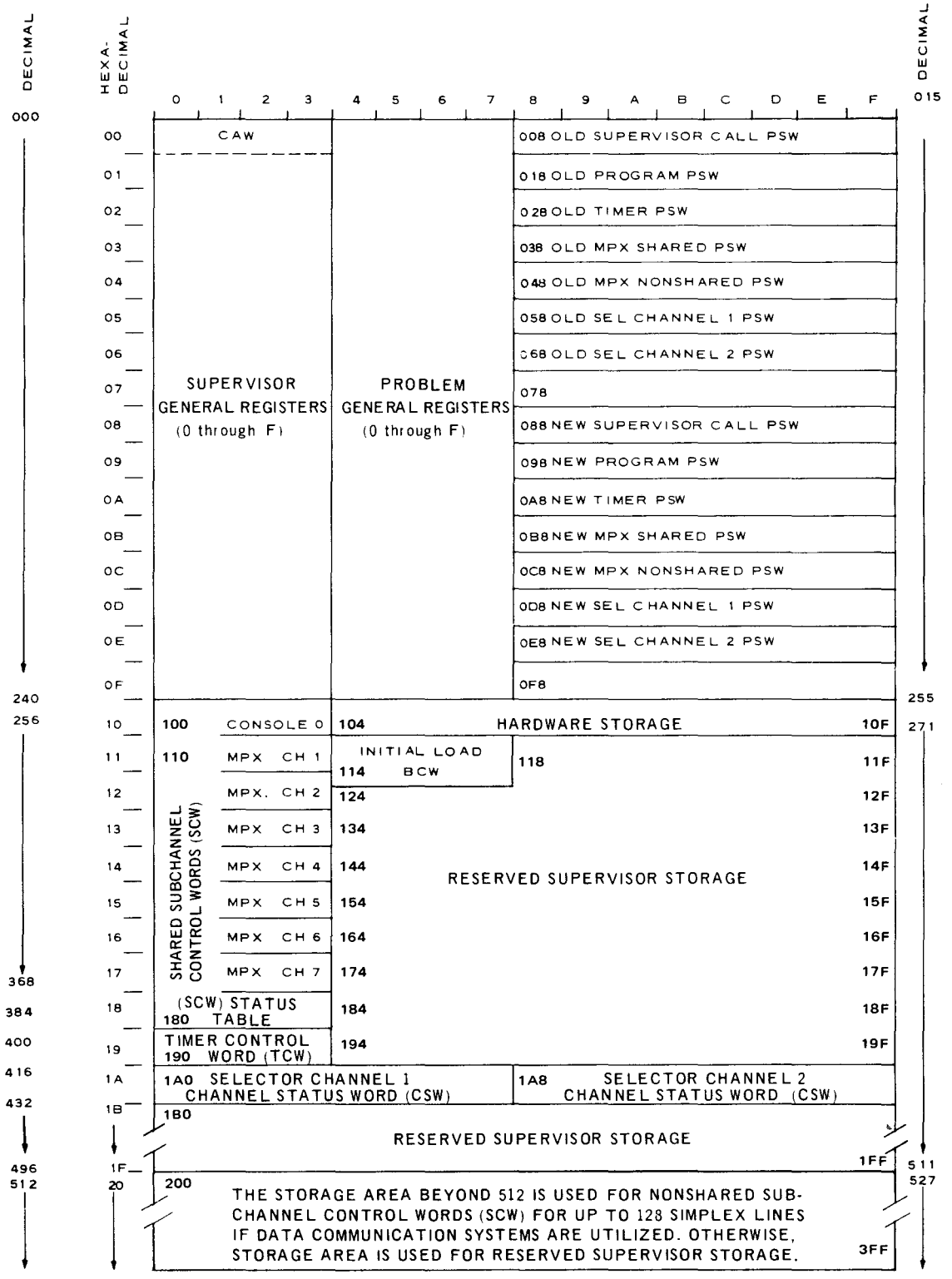



*Figure 4–1. Fixed Main Storage Assignments*

*Table 4–2. Fixed Main Storage Functions (Part 1 of 2)*

| | |
|---|---|
| Supervisor general registers $0_{16} - F_{16}$ ($000_{16} - 0F0_{16}$) | Sixteen word locations for storing operands, index numbers, and the results of arithmetic computations. They can be specified when the processor is in the supervisor state and the $p_r$ field (bit 14) of the current PSW is set to 0. In this state they can be addressed by way of the r, b, or x fields of the instruction, or by way of the normal operand addressing of main storage. (It should be noted the general register 0 cannot be addressed by way of the b or x fields of the instruction.) These locations can be addressed when the processor is in the problem state and the $p_r$ field is 0, but this is not recommended. When the start-I/O instruction is executed, location $000_{16}$ contains the channel address word (CAW). |
| Problem general registers $0_{16} - F_{16}$ ($004_{16} - 0F4_{16}$) | Sixteen word locations for storing operands, index numbers, and the results of arithmetic computations. They may be specified when the processor is in either the supervisor or program state and the $p_r$ field (bit 14) of the current PSW is set to 1. In the supervisor state, these locations can be addressed by way of the r, b, or x fields of the instruction, or by way of the normal operand addressing of main storage. In the problem state, these locations can be addressed only by way of the r, b, or x fields.<br><br>In the supervisor state, the processor can be switched between the supervisor and problem general registers by setting the $p_r$ field to the desired value (1 for the problem general registers; 0 for the supervisor registers). |
| Old supervisor call PSW ($008_{16}$) | One double-word location for storing the current PSW when a supervisor call interrupt occurs. |
| Old program PSW ($018_{16}$) | One double-word location for storing the current PSW when a program interrupt occurs. |
| Old timer PSW ($028_{16}$) | One double-word location for storing the current PSW when a timer interrupt occurs. |
| Old mpx shared PSW ($038_{16}$) | One double-word location for storing the current PSW when an interrupt occurs on a multiplexer shared subchannel. |
| Old mpx nonshared PSW ($048_{16}$) | One double-word location for storing the current PSW when an interrupt occurs on a multiplexer nonshared subchannel. |
| Old sel channel 1 PSW ($058_{16}$) | One double-word location for storing the current PSW when an interrupt occurs on selector channel 1. |
| Old sel channel 2 PSW ($068_{16}$) | One double-word location for storing the current PSW when an interrupt occurs on selector channel 2. |
| Location $078_{16}$ | Not assigned. |
| New supervisor call PSW ($088_{16}$) | One double-word location containing a new PSW that becomes the current PSW when a supervisor call interrupt occurs. |
| New program PSW ($098_{16}$) | One double-word location containing a new PSW that becomes the current PSW when a program interrupt occurs. |
| New timer PSW ($0A8_{16}$) | One double-word location containing a new PSW that becomes the current PSW when a timer interrupt occurs. |

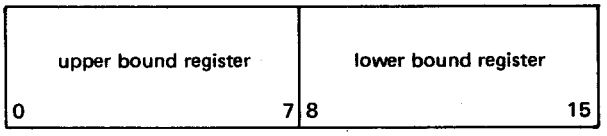
*Table 4–2. Fixed Main Storage Functions (Part 2 of 2)*

| | |
|---|---|
| New mpx shared PSW ($0B8_{16}$) | One double-word location containing a new PSW that becomes the current PSW when an interrupt occurs on the multiplexer shared subchannel. |
| New mpx nonshared PSW ($0C8_{16}$) | One double-word location containing a new PSW that becomes the current PSW, when an interrupt occurs on the multiplexer non-shared subchannel. |
| New sel channel 1 PSW ($0D8_{16}$) | One double-word location containing a new PSW that becomes the current PSW when an interrupt occurs on selector channel 1. |
| New sel channel 2 PSW ($0E8_{16}$) | One double-word location containing a new PSW that becomes the current PSW when an interrupt occurs on selector channel 2. |
| Location $0F8_{16}$ | Not assigned. |
| Console 0 ($100_{16}$) | One word location for storing a subchannel control word for the console connected to multiplexer shared subchannel 0. |
| Hardware storage ($104_{16}$) | Storage area that may serve as holding registers when SS-type instructions are executed. |
| Init load BCW ($114_{16}$) | One word location for storing the subchannel control word during an initial load operation. |
| Shared subchannel control words (SCW) Mpx ch 1 – mpx ch 7 ($110_{16}$) | Seven word locations for storing the subchannel control word for multiplexer shared subchannels 1 through 7. |
| Status table SCW ($180_{16}$) | One word location for storing the status table subchannel control word pointing to an area in main storage that holds status information for the multiplexer nonshared subchannels. |
| TCW ($190_{16}$) | One word location for storing the timer control word. |
| CSW 1 ($1A0_{16}$) | One double-word location for storing the channel status word for selector channel 1. |
| CSW 2 ($1A8_{16}$) | One double-word location for storing the channel status word for selector channel 2. |
| Supervisor storage ($118_{16}$) and ($1B0_{16}$) | Storage area that may serve any purpose of the programmer, providing the processor is in the supervisor state. |
| Multiplexer channel nonshared SCWs ($200_{16}$) | 128-word locations for storing the subchannel control words for each of the multiplexer nonshared subchannels. This area is not protected. |

The first 512 byte locations are protected from reading or writing when the processor is in the problem state. If the program attempts to gain access to a given location by way of the normal operand addressing of main storage, an addressing exception program interrupt is generated.

## 4.4. WRITE PROTECTION

Storage write protection is achieved through the use of the 16-bit limits register, the format of which is:




This register defines the storage area that the program is permitted to write by specifying the lower and upper bounds of an area that is beyond the first (a) 512 bytes for systems with less than 128K bytes of storage or (b) 1024 bytes for systems with more than 128K bytes of storage. If the number contained in the upper eight bits of the write address is greater than the contents in the lower bound register, the write operation is valid and a write protection fault will not occur.

$$\left[\text{Upper Bound Register}\right] \geq \left[\text{Upper 8 Bits of Write Address}\right] > \left[\text{Lower Bound Register}\right]$$

For example, if the content of the limits register is 0000 0001 | 0000 0000 and the system is less than 128K bytes of storage, writing is permitted in an area whose initial address is 512 and whose final address is 1023.

If the system has more than 128K bytes of storage, the same content of the limits register permits writing in an area whose initial address is 1024 and whose final address is 2047. The minimum address that can be included in a bounded area is either 512 ($200_{16}$) or 1024 ($400_{16}$), depending on size of storage. When the bound registers are equal, the write protection fault occurs unconditionally.

The limits register and the associated logic circuitry required to perform the check operation must be installed as a factory or field option. The instruction used to load the limits register, opcode 81, is a privileged instruction of the RS type. The hardware required to execute this instruction is included in the basic processor. Should the instruction be executed without the storage protection feature (F1091–00) being installed, the instruction functions as a no-operation instruction.

### 4.4.1. Parity Verification

Parity is constantly being checked by the processor and between the I/O channels and the I/O control units. This type of parity checking is called eighth-level and is not to be confused with character parity checks known as seventh level. Seventh-level parity checks occur between I/O channel and I/O control units. Eighth-level parity indicates eight bits plus an odd parity bit; seventh-level parity indicates a 7-bit character plus an odd or even parity bit as specified in the SCW. Odd parity is generated and checked on each byte of data transferred between the processor, main storage, and I/O channels. This parity is generated and checked in the storage logic for a processor transfer and in the I/O channel interface logic for an I/O channel transfer. The storage logic generates parity during the write cycle and checks parity during the read cycle. The I/O channel interface transfers parity during an output transfer sequence (data and command) and checks parity during an input sequence (data, status, sense, or address information). The I/O channel generates the parity for the device number transmitted during the selection sequence.

If a parity error is detected at the storage or I/O channel parity check logic, an immediate halt response is initiated. The halt logic stops the processor and I/O channel operation immediately upon detection of the error. Immediate instruction results and I/O channel conditions are displayed to facilitate maintenance procedures. A restart from the point of halt is provided. An override switch (maintenance aid) is provided to allow the central processor and I/O channel to run even though parity errors are detected.

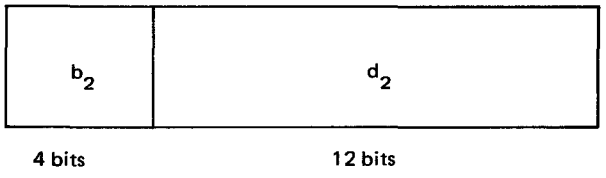
## 4.5. OPERAND ADDRESSING

Operands may appear in one of three places:

- in the instruction;
- in the operating registers; or
- in main storage.

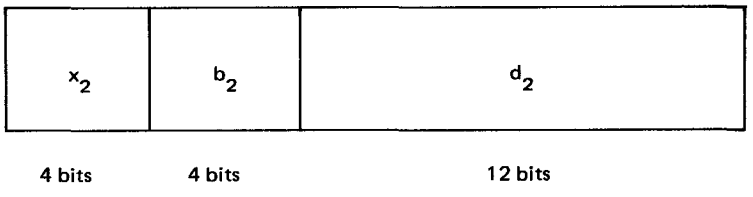
No address is needed for an operand that is part of the instruction (immediate operand). When operands are located in the register area (problem or supervisor), the register is addressed with a 1-digit hexadecimal code. Operands in main storage must be addressed through the rightmost 18 bits of a 24-bit logical address.

Instructions that access operands from main storage contain two operand specification fields:



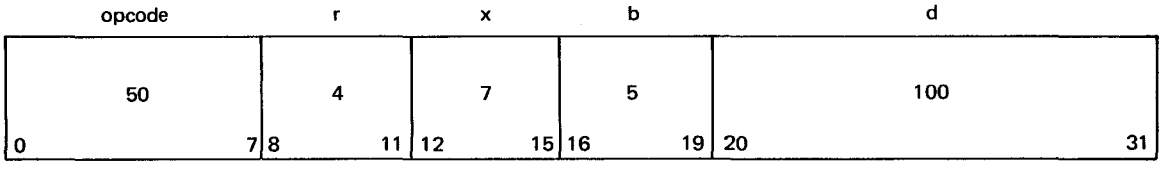

The b field designates a general purpose register. The rightmost 18 bits of this register represent the base address and are treated as a positive binary number. The 12 bits of the d field contain the displacement and also are treated as a positive binary number. The base address and the displacement are added together to obtain the effective address of the operand.

The RX type of instruction contains three fields to specify the operand address:




The $x_2$ field of the instruction designates one of the general purpose registers containing an index, which is treated as an 18-bit positive binary number and is added to the base address plus the displacement to obtain the effective address. When the $b_2$ and/or $x_2$ fields of an instruction contain 0's, no register is accessed and the field is not considered in calculating the effective address.

An example of this addressing:



Store the contents of register 4 (r) in the effective address obtained by adding the contents of base register 5 (b) and the contents of index register 7 (x) to the displacement value 100 (d).

This is a store instruction: operand 1 is stored in the effective address of operand 2.

Before execution:

contents of register 4 is 32;
contents of register 5 is 48;
contents of register 7 is 52;
contents of storage location 200 is 21.

After execution:

the contents of the registers remain the same;
effective address is 48 + 52 + 100 = 200;
contents of storage location 200 is 32.

# 5. System Console

## 5.1. GENERAL

The UNIVAC 9400 System and UNIVAC 9480 System console (system console) is a desk-like cabinet attached to one end of the processor cabinet (Figure 1–1). The system console consists of a printer, a keyboard with its associated control, and I/O interface logic designed to operate on the multiplexer I/O channel. Even though the system console is external to the processor, the control is housed in the processor cabinet.

## 5.2. COMPONENTS

The system console consists of two functional units operating through a common control.

- Keyboard for inserting information into the processor.
- Printer that provides printouts of information received from the processor.

For UNIVAC 9400 Systems with serial numbers from 100 to 204, the system console includes the keyboard shown in Figure 5–1.

For UNIVAC 9400 Systems with serial numbers from 205 and above, and the UNIVAC 9480 System, the system console includes the keyboard shown in Figure 5–2. Brief functional descriptions of the controls and indicators and keyboards for the system consoles are given in Table 5–1. Refer to the *UNIVAC 9400 System Operations Handbook Operator Reference, UP-7871* (current version) for more detailed descriptions of the UNIVAC 9400 System console controls and indicators.

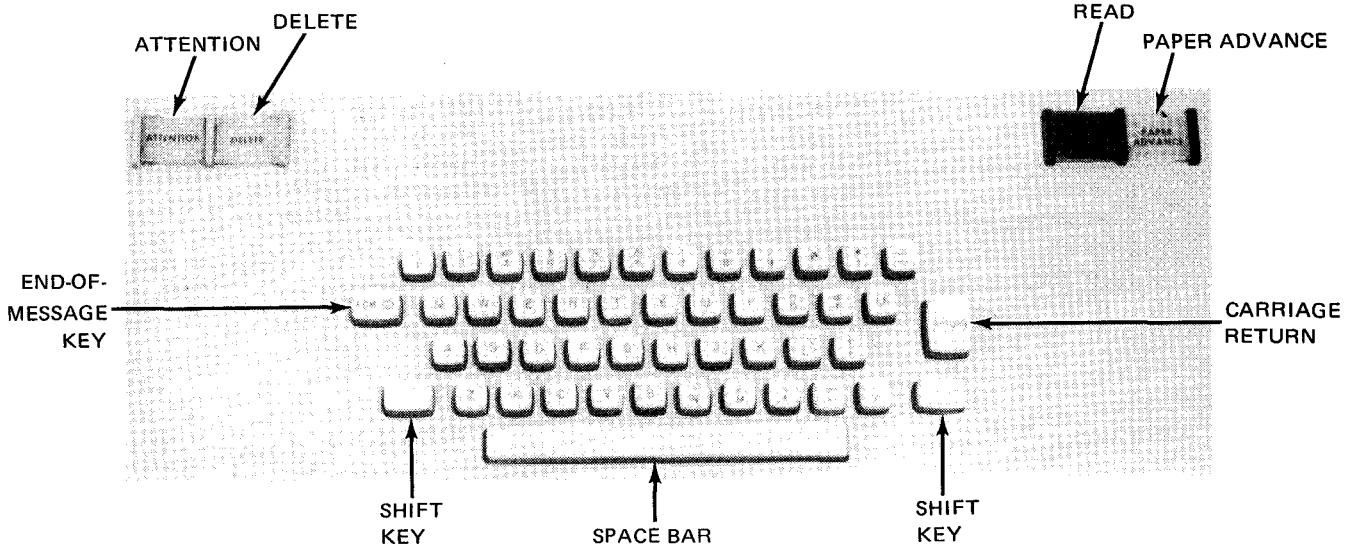



*Figure 5–1. System Console Keyboard (Serial Numbers 100 to 204)*

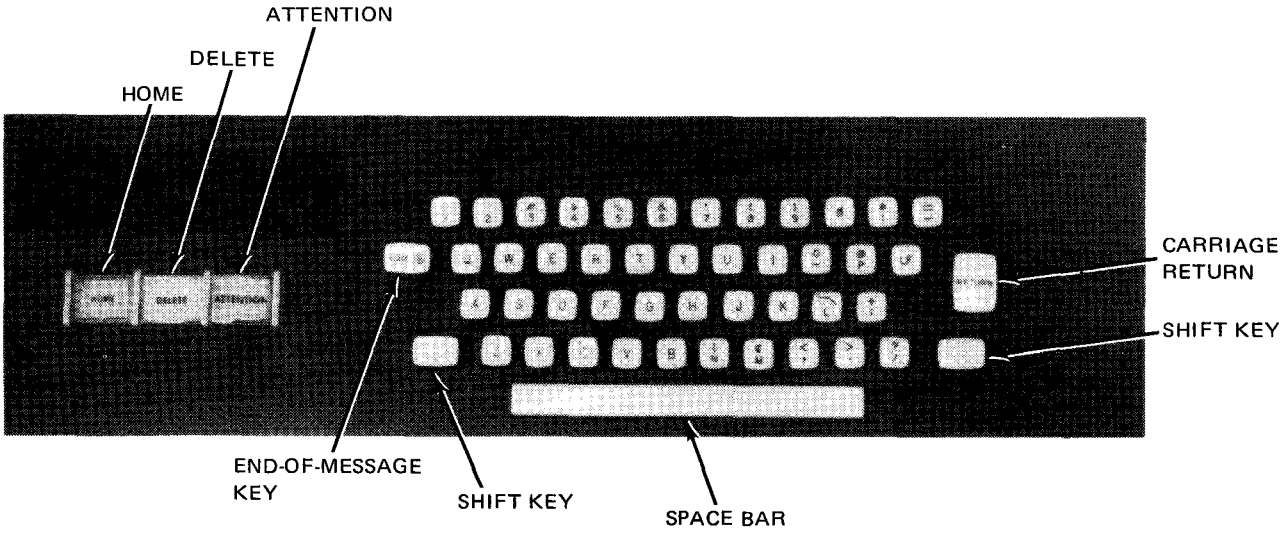



*Figure 5–2. System Console Keyboard (Serial Numbers 205 and Above and UNIVAC 9480 System)*

*Table 5–1. System Console Printer Controls, Indicators, and Keys (Part 1 of 2)*

| Control/Indicator | Function |
|---|---|
| Typewrite keys | These keys are arranged in standard typewriter fashion and include alphanumeric and symbolic functions. These are the keys for message composition. As a key is pressed, the character (uppercase or lowercase depending on shift key) is printed. |
| EOM $E_T$, Ⓢ key | For serial numbers 100 through 204: when pressed indicates end of message with $E_T$ printout. For serial numbers 205 and above: when pressed indicates end of message with Ⓢ printout. |
| ATTENTION momentary contact switch/indicator (white) | When pressed, indicator lights and sets the attention status bit when a subsystem control becomes available. Indicator is extinguished (cleared) when status bit is accepted by the channel. |
| DELETE momentary contact switch (white) | When pressed, sets unit exception status code and terminates the read operation. Unit exception status indication is the result of an incorrect keyin. |
| READ indicator (green) | Serial numbers 100 through 204 only: indicator lights when the subsystem control holds a read command. |
| HOME momentary contact switch (white) | Serial numbers 205 and above only: when pressed, activates the paper feed mechanism; paper advances to next home position. |

*Table 5–1. System Console Printer Controls, Indicators, and Keys (Part 2 of 2)*

| Control/Indicator | Function |
|---|---|
| INTERLOCK indicator (red) | Serial numbers 205 and above: a backlit indicator that lights when printer casework is opened.<br><br>If a read or write operation is in progress at the time the printer casework is opened, the operation will be terminated when the multiplexer channel acknowledges a subsequent data request. The UNIT CHECK signal is returned with the DEVICE END and CHANNEL END signals. Newly addressed read or write commands are not accepted; the UNIT CHECK signal is returned as initial selection status information. Sense bit 1, intervention required, is set to 1 for this condition.<br><br>The attention status bit is set when the casework is closed and the subsystem control is, or becomes, available. This status bit is sent to the channel regardless of any preceding status sequence. |
| PRINT CHECK indicator (red) | Serial numbers 205 and above only: a backlit indicator that lights when the printer activator fuse is blown and the activator does not operate.<br><br>If a read or write operation is in progress during the time this fuse blows, the operation will be terminated when the channel acknowledges a subsequent data request. The UNIT CHECK signal is returned with the DEVICE END and CHANNEL END signals. Newly addressed read or write commands are not accepted; the UNIT CHECK signal is returned as initial selection status information. Sense bit 3 (equipment check) is set to 1 for this condition. |
| FORMS END indicator (red) | Serial numbers 205 and above only: a backlit indicator that lights when two inches of paper remain in the printer.<br><br>The UNIT CHECK signal is returned as initial selection status information for newly addressed read or write commands.<br><br>Sense bit 1 (intervention required) is set to 1 for this condition. This operation does not terminate a read or write operation that is in progress; the UNIT CHECK signal is not returned with the DEVICE END or CHANNEL END signals. |

## 5.2.1. Keyboard

The keyboards generate 8-bit EBCDIC character codes. Each code is transferred to the control with an uppercase or lowercase STROBE signal. The characters and their corresponding codes are listed in Table 5–2.

When the control receives a read command, the keyboard transmits characters to the printer and the channel in an online mode, or to the printer only in an offline mode. Once the control has received a character, another character is not accepted until the first is printed.

When the end-of-message (EOM Ⓢ) key is pressed, the EOM character $E_T$ (serial numbers 100 through 204) is printed, or the graphic symbol Ⓢ (serial numbers 205 and above) is printed. Even though an upper/lowercase shift key is provided, all lowercase characters of single-character-marked keys are printed in uppercase characters.

*Table 5–2. Keyboard Character Coding (Serial Numbers 205 and Above)*

| Code | | EBCDIC Character | Special Graphic Symbol Definition |
|---|---|---|---|
| 0123 | 4567 | | |
| 1100 | 0001 | A | |
| 1100 | 0010 | B | |
| 1100 | 0011 | C | |
| 1100 | 0100 | D | |
| 1100 | 0101 | E | |
| 1100 | 0110 | F | |
| 1100 | 0111 | G | |
| 1100 | 1000 | H | |
| 1100 | 1001 | I | |
| 1101 | 0001 | J | |
| 1101 | 0010 | K | |
| 1101 | 0011 | L | |
| 1101 | 0100 | M | |
| 1101 | 0101 | N | |
| 1101 | 0110 | O | |
| 1101 | 0111 | P | |
| 1101 | 1000 | Q | |
| 1101 | 1001 | R | |
| 1110 | 0010 | S | |
| 1110 | 0011 | T | |
| 1110 | 0100 | U | |
| 1110 | 0101 | V | |
| 1110 | 0110 | W | |
| 1110 | 0111 | X | |
| 1110 | 1000 | Y | |
| 1110 | 1001 | Z | |
| 1111 | 0000 | 0 | |
| 1111 | 0001 | 1 | |
| 1111 | 0010 | 2 | |
| 1111 | 0011 | 3 | |
| 1111 | 0100 | 4 | |
| 1111 | 0101 | 5 | |
| 1111 | 0110 | 6 | |
| 1111 | 0111 | 7 | |
| 1111 | 1000 | 8 | |
| 1111 | 1001 | 9 | |
| 0100 | 1010 | [ | Left bracket |
| 0100 | 1011 | . | Period or decimal point |
| 0100 | 1100 | < | Less than |
| 0100 | 1101 | ( | Left parenthesis |
| 0100 | 1110 | + | Plus |
| 0100 | 1111 | \| | Logical OR |
| 0101 | 0000 | & | Ampersand |
| 0101 | 1010 | ] | Right bracket |
| 0101 | 1011 | $ | Dollar |
| 0101 | 1100 | * | Asterisk |
| 0101 | 1101 | ) | Right parenthesis |
| 0101 | 1110 | ; | Semicolon |
| 0101 | 1111 | ¬ | Logical NOT |
| 0110 | 0000 | - | Minus or hyphen |
| 0110 | 0001 | / | Slash |
| 0110 | 1011 | , | Comma |
| 0110 | 1100 | % | Percent |
| 0110 | 110 | _ | Underscore |
| 0110 | 1110 | > | Greater than |
| 0110 | 1111 | ? | Question |
| 0111 | 10'0 | : | Colon |
| 0111 | 1011 | # | Number |
| 0111 | 1100 | @ | At |
| 0111 | 1101 | ' | Prime or apostrophe |
| 0111 | 1110 | = | Equal |
| 0111 | 1111 | " | Quotation |
| 0100 | 0000 | (Sp) | Space |
| 0010 | 0101 | (LF) | Line feed |
| 0000 | 1101 | (CR) | Carriage return |
| 0011 | 0111 | Ⓢ | End-of-message (EOM) |

## 5.2.2. Printer

The printer used with serial numbers 100 through 204 processors normally prints 10 characters per second on 8 1/2-inch friction-feed roll paper. EBCDIC-coded information from the processor and keyboard is converted to an altered ASCII by the control and is transmitted to the printer in a bit-serial sequence. The printer, designed to accept standard ASCII character, is modified to accept the altered code. This modification requires the rearrangement of characters on the printer as shown in Table 5–3.

For processors with serial numbers 205 and above, the code wheel is the EBCDIC font (Table 5–2) and prints 25 characters per second.

*Table 5–3. Printer Character Coding (Serial Numbers 100 to 204 (Part 1 of 2)*

| Bits | Code | | ASCII Character | Special Graphic Symbol Definition | Modified ASCII Character |
|---|---|---|---|---|---|
| | 0123 | 4567 | | | |
| | 1100 | 0001 | A | | A |
| | 1100 | 0010 | B | | B |
| | 1100 | 0011 | C | | C |
| | 1100 | 0100 | D | | D |
| | 1100 | 0101 | E | | E |
| | 1100 | 0110 | F | | F |
| | 1100 | 0111 | G | | G |
| | 1100 | 1000 | H | | H |
| | 1100 | 1001 | I | | I |
| | 1101 | 0001 | Q | | J |
| | 1101 | 0010 | R | | K |
| | 1101 | 0011 | S | | L |
| | 1101 | 0100 | T | | M |
| | 1101 | 0101 | U | | N |
| | 1101 | 0110 | V | | O |
| | 1101 | 0111 | W | | P |
| | 1101 | 1000 | X | | Q |
| | 1101 | 1001 | Y | | R |
| | 1010 | 0010 | " | Quotation | S |
| | 1010 | 0011 | # | Number | T |
| | 1010 | 0100 | $ | Dollar | U |
| | 1010 | 0101 | % | Percent | V |
| | 1010 | 0110 | & | Ampersand | W |
| | 1010 | 0111 | ' | Prime or apostrophe | X |
| | 1010 | 1000 | ( | Left parenthesis | Y |
| | 1010 | 1001 | ) | Right parenthesis | Z |
| | 1011 | 0000 | 0 | | 0 |
| | 1011 | 0001 | 1 | | 1 |
| | 1011 | 0010 | 2 | | 2 |
| | 1011 | 0011 | 3 | | 3 |
| | 1011 | 0100 | 4 | | 4 |
| | 1011 | 0101 | 5 | | 5 |
| | 1011 | 0110 | 6 | | 6 |
| | 1011 | 0111 | 7 | | 7 |
| | 1011 | 1000 | 8 | | 8 |
| | 1011 | 1001 | 9 | | 9 |
| | 1100 | 1010 | J | | [ |
| | 1100 | 1011 | K | | . |
| | 1100 | 1100 | L | | < |
| | 1100 | 1101 | M | | ( |
| | 1100 | 1110 | N | | + |
| | 1100 | 1111 | O | | \| |
| | 1101 | 0000 | P | | & |

*Table 5–3. Printer Character Coding (Serial Numbers 100 to 204 (Part 2 of 2)*

| Bits | Code | | ASCII Character | Special Graphic Symbol Definition | Modified ASCII Character |
|---|---|---|---|---|---|
| | 0123 | 4567 | | | |
| | 1101 | 1010 | Z | | ] |
| | 1101 | 1011 | $E_T$ | End-of-message (EOM) | $ |
| | 1101 | 1100 | ¬ | Logical NOT | * |
| | 1101 | 1101 | [ | Left bracket | ) |
| | 1101 | 1110 | \| | Logical OR | ; |
| | 1101 | 1111 | – | Underscore | ¬ |
| | 1010 | 0000 | (Sp) | Space | - |
| | 1010 | 0001 | ] | Right bracket | / |
| | 1010 | 1011 | + | Plus | , |
| | 1010 | 1100 | , | Comma | % |
| | 1010 | 1101 | - | Minus or hyphen | – |
| | 1010 | 1110 | . | Period or decimal point | > |
| | 1010 | 1111 | / | Slash | ? |
| | 1011 | 1010 | : | Colon | : |
| | 1011 | 1011 | ; | Semi colon | # |
| | 1011 | 1100 | < | Less than | @ |
| | 1011 | 1101 | = | Equal | ' |
| | 1011 | 1110 | > | Greater than | = |
| | 1011 | 1111 | ? | Question | " |
| | 1100 | 0000 | @ | At | SP |
| | 1000 | 1010 | (LF) | Line feed | LF |
| | 1000 | 1101 | (CR) | Carriage return | CR |
| | 1010 | 1010 | * | Asterisk | $E_T$ |

NOTE:

The characters used with ASCII are replaced with the corresponding characters for using modified ASCII.

Figure 5–3 shows the transfer sequence for each character transmitted to the printer. Each unit of time is 9.09 milliseconds (205 and above), 22.63 milliseconds (100 through 204). Each character is preceded by a start bit. Two stop bits follow to ensure sychronization. The eighth bit of the character is always 1 in ASCII. The printer interprets three nonprinting functions: carriage return (CR), line feed (LF), and space (SP). The CR and LF functions must be inserted by the program to prevent overprinting. To ensure that a character is not printed while the carriage is returning, the LF function must follow the CR function. Characters are sent to the printer at a maximum rate of one per 100 milliseconds. The time required for the printer to execute the LF and CF functions and to transfer each character:

- Line feed — 33 milliseconds
- Carriage return — 135 milliseconds (from end of a 75-character line)
  100 milliseconds (from end of a 50-character line)
- Character (or space function) — 100 milliseconds

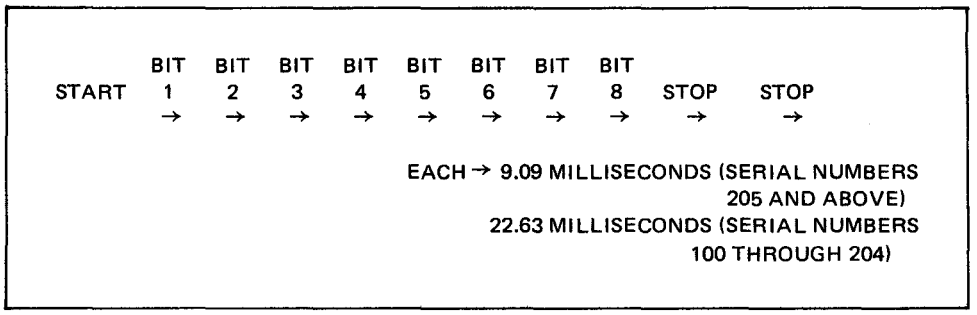



*Figure 5–3. Character Transfer Sequence*

The printer used with serial numbers 205 and above processors provides a control with the capability of translating EBCDIC-coded data bytes to interface signals required by the printer. The arrangement of the font on the print wheel is shown in Table 5–4. This printer interprets two nonprinting functions: carriage return (CR) and line feed (LF). No special order in the interpretation of these functions is necessary to ensure proper printer operation.

*Table 5–4. Serial Arrangement of Font on Printer*

| ASCII Character | Special Graphic Symbol Definition | ASCII Character | Special Graphic Symbol Definition |
|---|---|---|---|
| A | Space | / | Slash |
| B | | T | |
| C | | U | |
| D | | V | |
| E | | W | |
| F | | X | |
| G | | Y | |
| H | | Z | |
| I | | $E_T$ or Ⓢ | End-of-message (EOM) |
| ¢ | Cent | , | Comma |
| . | Period or decimal point | % | Percent |
| < | Less than | _ | Underscore |
| ( | Left parenthesis | > | Greater than |
| + | Plus | ? | Question |
| ¦ | Vertical bar | 0 | Zero |
| & | Ampersand | 1 | |
| J | | 2 | |
| K | | 3 | |
| L | | 4 | |
| M | | 5 | |
| N | | 6 | |
| O | | 7 | |
| P | | 8 | |
| Q | | 9 | |
| R | | : | Colon |
| ! | Exclamation | # | Number |
| $ | Dollar | @ | At |
| * | Asterisk | ' | Prime or apostrophe |
| ) | Right parenthesis | = | Equal |
| ; | Semicolon | " | Quotation |
| ¬ | OR | | |
| | Minus or hyphen | | |

The carriage returns automatically and the paper advances one line when the carriage reaches the right margin. The right margin switch is enabled when the carriage advances to the right of the nth + 2 column, where n is the maximum number of printed columns on the selected paper stock. Requests to transfer data are held until the carriage returns to the left margin stop. Automatic carriage return is not indicated to the program.

### 5.2.3. Command Codes

The console control responds to the commands listed in Table 5–5. When a subsystem control is available and is online, it accepts a command byte with odd parity.

*Table 5–5. Command Codes*

| | Bit Position | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Command** | **P** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Test-I/O | P | D | D | D | D | 0 | 0 | 0 | 0 |
| Sense | P | D | D | D | D | 0 | 1 | 0 | 0 |
| Write | P | D | D | D | D | D | D | 0 | 1 |
| Read | P | D | D | D | D | D | D | 1 | 0 |
| Data-hold | P | D | D | D | D | 1 | 1 | 1 | 1 |
| No-operation | P | D | D | D | D | 0 | 0 | 1 | 1 |
| Select-shared-address | P | D | D | D | D | 1 | 0 | 1 | 1 |
| Select-nonshared-address | P | D | D | D | D | 0 | 1 | 1 | 1 |
| Illegal-command | P | D | D | D | D | 1 | D | 0 | 0 |

NOTES:

1. D denotes noninterpreted bit.
2. P denotes odd parity bit.

- Test-I/O (DDDD 0000)

  The control presents all pending status or zero status indication to the channel. The status register in the control is cleared when the status byte is accepted by the channel.

- Sense (DDDD 0100)

  The control presents one byte of sense information to the channel. (This information is not cleared by the sense command.) This command is used in conjunction with the data hold command in performing a data turnaround operation. The first sense command after the data hold command causes the control to return the data turnaround data byte and reset the control. Further sense commands return the sense information.

- Write (DDDD DD01)

  The write command conditions the control to request data from the channel, modify it, and send it to the printer.

- Read (DDDD DD10)

  The read command conditions the control to accept data from the keyboard. The data byte presented to the channel is printed after SERVICE OUT is received for SERVICE IN.

- Data-hold (DDDD 1111)

  This is a control command that conditions the control to request one byte of data and store it without printing. This byte is returned in response to the first sense command. Read and write commands sent to the control before the turnaround data has been returned resets the control to execute a sense command in the normal manner. The data hold command has to be reinitiated to set up the data turnaround operation.

- No-Operation (DDDD 0011)

  This is an immediate control command. The control returns CHANNEL END and DEVICE END for the initial selection sequence (ISS) without taking any action at the device.

- Select-shared-address (DDDD 1011)

  This is an immediate control command that conditions the control to recognize and return 1000 DDDD for an address.

- Select-nonshared-address (DDDD 0111)

  This is an immediate control command that conditions the control to recognize and return 0000 0000 for an address. The control continues to recognize address 1000 DDDD in the nonshared mode, where D in the address denotes noninterpreted bits.

- Illegal-command (DDDD 1D00)

  This command causes the control to reject the command by setting unit check status and command reject sense.

## 5.2.4. Status Codes

The control generates status information when unusual or ending conditions are detected in an operation, or when other conditions necessitate attention from the program. The status codes are listed in Table 5–6.

*Table 5–6. Status Codes*

| | Bit Position | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Status** | **P** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Attention | P | 1 | 0 | 0 | D | D | D | D | D |
| Busy | P | D | 0 | 0 | 1 | D | D | D | D |
| Channel end | P | D | 0 | 0 | D | 1 | D | D | D |
| Device end | P | D | 0 | 0 | D | D | 1 | D | D |
| Unit check | P | D | 0 | 0 | D | D | D | 1 | D |
| Unit exception | P | D | 0 | 0 | D | D | D | D | 1 |

NOTES:

1. D denotes a status bit that also may be set in the status byte.
2. P denotes odd parity bit.

- Attention (100D DDDD)

  This is an unsolicited status code that is set by pressing the console ATTENTION switch. It can be set and held any time the control is online, but is presented to the channel only when the control is available. Attention status indicates to the program that the operator has requested a read command. It normally is presented by itself.

- Busy (D001 DDDD)

  This code is set and returned for ISS status for all commands, except test-I/O, if the control is executing a previous command or has pending status.

- Channel end (D00D 1DDD)

  This code always is generated with device end status.

- Device end (D00D D1DD)

  This code is generated when the control detects ending conditions. The conditions occur when:

  – data transfers are terminated, except if they are terminated by an ISS;

  – unit check status conditions are detected, except during ISS; or if unit exception is set; or

  – the command is immediate; device end is returned for ISS status.

- Unit check (D00D DD1D)

  This code indicates that the control has detected an unusual condition that is detailed by the information available to a sense command. It is set when:

  – the control is available and the presented command has even parity, or when it is an illegal command; or

  – the data byte sent to the control for data hold or write has even parity.

  Unit check immediately terminates the data transfer operation; it causes the command to be rejected during ISS.

- Unit exception (D00D DDD1)

  This code is generated when the console DELETE switch is pressed. This status condition can be set only when the control holds a read command. Unit exception immediately terminates the read operation and prevents further characters from being entered through the keyboard until a new read command is issued.

All status bits, except device end and channel end, are suppressible. Device end and channel end are suppressible only if they have been stacked.

### 5.2.5. Sense Codes

The control detects the sense conditions defined in Table 5–7.

*Table 5–7. Sense Byte Codes*

| | Bit Position | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Sense** | **P** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| Command reject | P | 1 | 0 | D | 0 | 0 | 0 | 0 | 0 |
| Bus check | P | D | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Intervention* required | P | D | 1 | D | D | 0 | 0 | 0 | 0 |
| Equipment* check | P | D | D | D | 1 | 0 | 0 | 0 | 0 |

*For printers with serial numbers 205 and above only.

NOTES:

1. D denotes a sense bit that also may be set in the sense byte.
2. P denotes odd parity bit.

- Command reject (10D0 0000)

  This code is set if the command code is illegal; the command code byte parity must be odd, and the control must be in the available state before command reject is sensed.

- Bus check (D010 0000)

  This code is set if the bus out data byte (for write or data hold) has even parity, or if the command code byte has even parity.

- Intervention required (D1DD 0000)

  This code is set to indicate that only a limited amount of paper left in the printer or that its casework (access panel) is open.

- Equipment check (DDD1 0000)

  This code is set to indicate that the printer actuator fuse is defective.

The sense byte is not reset or changed by the sense, test-I/O, no-operation, or the other immediate control commands. It is cleared by the write, read, and data-hold commands.

## 5.2.6. Operating Characteristics

### 5.2.6.1. Interface

The control unit is connected to the multiplexer-I/O channel interface and operates in byte-multiplex mode; that is, it receives or transmits one 8-bit data byte then releases the channel. This mode of operation continues until the data transfer is completed.

The control unit is defined to be the lowest priority control on the multiplexer-I/O interface.

### 5.2.6.2. Initial Selection Sequence

The action taken by the control unit to an ISS depends upon its current state:

- Offline

  The control unit generates the SELECT OUT signal and takes no further action.

- Online

  The control unit decodes the address on the bus out lines as soon as the ADDRESS OUT signal is raised. It blocks the propagation of the SELECT OUT signal when both of the following conditions exist:

  – Parity on the bus out lines is odd.

  – The four high-order bits of the bus out address agree with the control address.

  The control unit response to the command depends on whether it is available, executing a previous command, or holding pending status.

- Available

  The control unit accepts the commands if the command byte parity is odd and if the command code is valid. Illegal command codes and/or even parity cause the command to be rejected with the appropriate sense and status codes set.

  Immediate commands (no-operation, select-shared-address, select-nonshared-address) condition the control unit to return device end and channel end for ISS status. For immediate commands, test-I/O command, commands with invalid codes, or even parity, the control unit becomes available again after the status condition is accepted by the channel. Data transfer commands (read, write, data-hold, sense) initiate a control-unit-initiated sequence after the immediate status byte is accepted by the channel.

- Executing a previous command

  The control unit holds and decodes but does not execute the new command if the command byte parity is odd. The command that the control unit had been executing is terminated. The status byte presented in reponse to the command depends on the new command as follows:

  – For a test-I/O command, the control unit returns a status byte of zeros.

  – For commands other than the test-I/O commands, and for illegal commands, the control unit returns a status byte with only the busy bit set.

  If the command byte parity is even, the command is not decoded. The control unit terminates the previous command and returns a status byte with only the busy bit set.

- Pending status

  The control unit proceeds with initial selection sequence except that the pending status condition (regardless of whether it had been stacked or suppressed) is returned with the status byte.

  When the status indication is accepted by the channel for commands other than test-I/O or illegal commands, the control unit clears the status byte, disconnects, and becomes available.

  In all cases, the COMMAND OUT signal in response to STATUS IN signal causes the status byte to be stacked.

### 5.2.6.3. Control-Unit-Initiated Sequence

The transfer of data and status, other than initial selection status, is requested by the control unit in the control-unit-initiated sequence.

- Data transfers

  The control unit holding a valid command requiring data transfers attempts to initiate a control-unit-initiated sequence, as soon as possible after disconnecting from ISS, for write, data hold, and sense commands, and after data has been transferred into it from the keyboard for a read command. The control unit disconnects and initiates a new control-initiated sequence for each data byte. Data transfers continue until:

  - the channel responds to SERVICE IN with COMMAND OUT;
  - one byte of data has been transferred for the data-hold command;
  - one byte of sense or turnaround information has been transferred for the sense command;
  - the end-of-message (EOM) character has been received either from the channel or from the keyboard (character is before the control unit terminates the data transfer);
  - the DELETE switch is set if the control unit holds a read command;
  - even parity is detected by the control unit for an output data byte from the channel; or
  - the channel terminates the current command with an ISS.

  Input data to the channel (read command) is printed only after SERVICE OUT is received for SERVICE IN, indicating that the channel has received the data byte.

- Status transfers

  The control unit attempts to initiate a control-unit-initiated sequence as soon as possible after detecting the conditions requiring a status transfer. Initiation occurs:

  - after the control unit has disconnected from termination of a data sequence, except when the data sequence was terminated by an ISS; or
  - any time the status byte has been stacked, provided it is not being suppressed. The control unit attempts a control-unit-initiated sequence for suppressed status as soon as the SUPPRESS OUT signal drops.

### 5.2.6.4. Control Unit Resets

Reset conditions occur when:

- System reset

  The system reset master clears the control unit, causing it to disconnect immediately from the interface, and clears the status, sense, and command registers. The control unit is set to the available state and sends its shared device number for its address.

- Interface disconnect

  The control unit is not required to respond to an interface disconnect sequence.

- Selective reset

  The control unit is not required to recognize the selective reset sequence when logically connected to the interface.

### 5.2.6.5. Chaining

The control unit takes no special action for chaining.

# Appendix A. Instructions

The UNIVAC 9400 and UNIVAC 9480 Systems are provided with 70 instructions each (including 7 privileged) in normal operational modes, identified by unique codes (opcodes).

Table A–1 is an alphabetical listing of the instructions, including instruction formats, hexadecimal codes, instruction mnemonics, and execution times.

*Table A–1. Alphabetical List of Instructions (Part 1 of 2)*

| Instruction | Format | Opcode | Mnemonic | Time in Microseconds |
|---|---|---|---|---|
| Add | RR | 1A | AR | 6.0 |
| Add | RX | 5A | A | 6.0 |
| Add-decimal | SS | FA | AP | $15.0 + 2.4n_1$; if recomplement, add $4.8 + 2.4n_1$ |
| Add-half-word | RX | 4A | AH | 6.0 |
| Add-immediate | SI | 93 | AI | 4.2 |
| AND | RR | 14 | NR | 6.0 |
| AND | RX | 54 | N | 6.0 |
| AND | SS | D4 | NC | $15.0 + 2.4n$ |
| AND-immediate | SI | 94 | NI | 4.2 |
| Branch-and-link | RR | 05 | BALR | 6.0 |
| Branch-and-link | RX | 45 | BAL | 4.8 |
| Branch-on-condition | RR | 07 | BCR | 4.2 |
| Branch-on-condition | RX | 47 | BC | 3.0 |
| Branch-on-count | RR | 06 | BCTR | 7.2 |
| Branch-on-count | RX | 46 | BCT | 6.0 |
| Compare | RR | 19 | CR | 6.0 |
| Compare | RX | 59 | C | 6.0 |
| Compare-decimal | SS | F9 | CP | $15.0 + 2.4n_1$; if recomplement, add $4.8 + 2.4n_1$ |
| Compare-half-word | RX | 49 | CH | 6.0 |
| Compare-logical | RR | 15 | CLR | 6.0 |
| Compare-logical | RX | 55 | CL | 6.0 |
| Compare-logical | SI | 95 | CLI | 4.2 |
| Compare-logical | SS | D5 | CLC | $15.0 + 2.4n$ |
| Divide-decimal | SS | FD | DP | $26.4 (n_1 - n_2) (n_2 + 2.99) - 10.8n_2 - 23.5$ |
| Edit | SS | DE | ED | $13.8 + 3.6n + 1.3n_s$ |
| Exclusive-OR | RR | 17 | XR | 6.0 |
| Exclusive-OR | RX | 57 | X | 6.0 |
| Exclusive-OR | SI | 97 | XI | 4.2 |
| Exclusive-OR | SS | D7 | XC | $15.0 + 2.4n$ |
| Halt-and-proceed | SI | 99 | HPR | 6.0 |
| Insert-character | RX | 43 | IC | 4.2 |
| Load | RR | 18 | LR | 4.8 |
| Load | RX | 58 | L | 4.8 |
| Load-address | RX | 41 | LA | 4.8 |
| Load-and-test | RR | 12 | LTR | 4.8 |

*Table A–1. Alphabetical List of Instructions (Part 2 of 2)*

| Instruction | Format | Opcode | Mnemonic | Time in Microseconds |
|---|---|---|---|---|
| Load-half-word | RX | 48 | LH | 6.0 |
| Load-limits-register | RS | 81 | LLR | 6.0 |
| Load-multiple | RS | 98 | LM | 2.4 + 2.4f |
| Load-program-status-word | SI | 82 | LPSW | 7.2 |
| Move-characters | SS | D2 | MVC | 15.0 + 2.4n |
| Move-immediate | SI | 92 | MVI | 4.2 |
| Move-numerics | SS | D1 | MVN | 15.0 +2.4n |
| Move-with-offset | SS | F1 | MVO | $15.0 + 2.4n_1$ |
| Move-zones | SS | D3 | MVZ | 15.0 + 2.4n |
| Multiply-decimal | SS | FC | MP | $21.6\ (n_1 - n_2)\ (n_2 + 2.68) - 10.8n_2 - 15.3$ |
| OR | RR | 16 | OR | 6.0 |
| OR | RX | 56 | O | 6.0 |
| OR | SS | D6 | OC | 15.0 + 2.4n |
| OR-immediate | SI | 96 | OI | 4.2 |
| Pack | SS | F2 | PACK | $12.6 + 4.8n_1$ |
| Set-program-mask | RR | 04 | SPM | 6.0 |
| Set-system-mask | SI | 80 | SSM | 6.0 |
| Shift-left-single-logical | RS | 89 | SLL | $11.4 + 1.2c_1$ |
| Shift-right-single-logical | RS | 88 | SRL | 11.4 if $c_2 = 0$; 15.6 if $c_2 = 16$; 16.2 if $c_2 =$ 32 or 48; $54.0 - 2.4c_1$ if $0 < c_2 < 16$; $54.6 - 2.4\ c_1$ if $c_2 > 16$ |
| Start-I/O | SI | 9C | SIO | 6.0 + CV Time |
| Store | RX | 50 | ST | 6.0 |
| Store-character | RX | 42 | STC | 4.2 |
| Store-half-word | RX | 40 | STH | 4.2 |
| Store-multiple | RS | 90 | STM | 2.4 + 2.4f |
| Subtract | RR | 1B | SR | 6.0 |
| Subtract | RX | 5B | S | 6.0 |
| Subtract-decimal | SS | FB | SP | $15.0 + 2.4n_1$; if recomplement, add $4.8 + 2.4n_1$ |
| Subtract-half-word | RX | 4B | SH | 6.0 |
| Supervisor-call | RR | 0A | SVC | 7.8 (includes IPT) |
| Supervisor-load-multiple | RS | B8 | SLM | 2.4 + 2.4f |
| Supervisor-store-multiple | RS | B0 | SSTM | 2.4 + 2.4f |
| Test-under-mask | SI | 91 | TM | 6.0 |
| Translate | SS | DC | TR | $13.8 + 4.8n_1$ |
| Unpack | SS | F3 | UNPK | $15.0 + 2.4n_1$ |
| Zero-and-add-decimal | SS | F8 | ZAP | $15.0 + 24n_1$; if recomplement, add $4.8 + 24n_1$ |

NOTES:

1. In the RX instructions the timing is increased by 1.2 microseconds when the index (x) field is not equal to 0.

2. Interrupt control timing is 6.0 microseconds; if it happens:

f is the number of full words;
n is the number of result bytes;
$n_1$ is the number of bytes in operand 1;
$n_2$ is the number of bytes in operand 2;
$n_s$ is the number of sign digits in right digit of operand 2 at digit select or significant start character cycle;
$c_1$ is the LSB 4 bits of (b) + d, which is specified by instruction;
$c_2$ is the LSB 6 bits of (b) + d, which is specified by instruction.

# Appendix B. Glossary

## A

**ASCII**
American Standard Code for Information Interchange

## B

**BCI**
Buffer control interrupt

**BCW**
Buffer control words

**BCWI**
Buffer control word index

## C

**CA**
Communications adapter

**CAW**
Channel address word

**CBI**
Control byte interrupt

**CCW**
Channel command word

**CR**
Carriage return

**CSW**
Channel status word

## D

**DCS**
Data communications subsystem

## E

**EBCDIC**
Extended binary coded decimal interchange code

**EOM**
End of message

## I

**IC**
Interrupt count

**ILC**
Instruction length code

**IPT**
Internal processing time

**ISS**
Initial selection sequence

## K

**K**
1024 ($2^{10}$) used to specify main storage capacity only

**k**
1000

## L

**LF**
Line feed

## P

**PSW**
Program status word

**PSWR**
PSW register

## R

**RTC**
Running time counter

## S

**SCW**
Subchannel control word

**SP**
Space

**SVC**
Supervisor-call instruction

**SVI**
Supervisor call interrupt

## T

**TCW**
Timer control word

**TIO**
Test I/O

# Index

Comments concerning this manual may be made in the space provided below. Please fill in the requested information.

System: _____

Manual Title: _____

UP No: _____ Revision No: _____ Update: _____

Name of User: _____

Address of User: _____

Comments:

CUT

FOLD

FIRST CLASS
PERMIT NO. 21
BLUE BELL, PA.

**BUSINESS REPLY MAIL** NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

UNIVAC

P.O. BOX 500

BLUE BELL, PA. 19422

ATTN: SYSTEMS PUBLICATIONS DEPT.

FOLD

CUT