60364500

CONTROL DATA CORPORATION

CONTROL DATA®
FA710-A/B/C/D, FA719-A/B/C/D, FA720-A/B/C/D, FA722-A/B/C, FA723-A/B
DISK CONTROLLERS
DT716-A
DOUBLE DENSITY OPTION

GENERAL DESCRIPTION
SUBSYSTEM PROCESSOR INSTRUCTIONS
FUNCTION CODES
STATUS REPORTING
CONTROL LOGIC INSTRUCTIONS
DIRECTOR SEQUENCING AND SELECTION
SPECIAL HARDWARE CONSIDERATIONS
ERROR DETECTION AND CORRECTION
6000 COUPLER PROGRAMMING
7000 COUPLER PROGRAMMING

HARDWARE REFERENCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
01	Review edition.
02	Preliminary edition.
03	Various technical and editorial corrections.
A	Final edition.
B	Manual revised; includes Engineering Change Order 32983 which incorporates Engineering
(10-13-72)	Change Order 32086 into this manual. Pages revised: 5-14 and 5-39.
C	Manual revised; includes Engineering Change Order 32984 which incorporates Engineering
(10-23-72)	Change Order 32449 into this manual. Pages revised: 5-18 and 5-29.
D	Manual revised; incorporates Field Change Order 32900 which revises page 3-1. Remaining
(1-24-73)	revised pages add FA710-C/D models and correct previous documentation errors. Pages
	revised: Cover, iii, 5-15, 5-30 through 5-34, 8-2, 9-3, 9-4, 9-7, 9-9.
E	Manual revised; includes Engineering Change Order 33910, publications change only. Cover and
(6-25-73)	pages iii, v, 1-1, 1-3 are revised. Pages iv, 1-4, 1-5 and Section 10 Divider are added. Any
	reference to FA710-E/F has been removed from this manual.
F	Manual revised; includes Engineering Change Order 35406. Cover, Title Page, iii, iv, v, vi, vii,
(8-12-74)	1-1, 1-4, 3-1, 5-18, 5-35, 6-1, 6-2, 6-4, 9-1, 9-3, 9-4, 9-5, 9-6, 9-7, 9-8, 9-9, 9-10, 9-11,
	9-12, 9-13 and 9-17 are revised. Pages viii, 9-13.0, 9-13.1 and Section 10 are added.
G	Manual revised; includes Engineering Change Order 36247 (publications change only).
(3-31-75)	Cover, Title Page, vi, vii, 4-3, 5-13, 5-16, 5-22, 5-30 through 5-34, 5-39, 6-1, 6-2, and
	7-6 are revised. Pages 5-34.1 and 6-4.1 are added.
H	Manual revised; includes Engineering Change Order 36259, publications change only. Cover, Title
(7-8-75)	Page, iii, iv, v, 1-1, and 1-4 are revised.
Publication No. 60364500	

Printed in the United States of America

Address comments concerning this manual to:
Control Data Corporation
Publications and Graphics Division
4201 North Lexington Avenue
Arden Hills, Minnesota 55112

or use Comment Sheet in the back of this manual.

PREFACE

This manual contains internal programming information for the CONTROL DATA® FA710-A/B/C/D Disk Controller, FA719-A/B/C/D Disk Controller, FA720-A/B/C/D Disk Controller, FA722-A/B/C Disk Controller, and FA723-A/B Disk Controller. The sections contained in this manual and their applicability to the various equipment models are described as follows:

1. General Description - This section describes the physical makeup of each of the equipment configurations.
2. Subsystem Processor Instructions - This section provides programming information which is applicable to the subsystem processor used in all of the equipment configurations.
3. Function Codes - This section describes the function codes used for communication between the subsystem processor and the control logic. Section 3 is applicable to all equipment configurations.
4. Status Reporting - This section describes the status codes used for communication between the subsystem processor and the control logic. Section 4 is applicable to all equipment configurations.
5. Control Logic Instructions - This section describes the instruction set used in the control logic portion of the disk controller. Section 5 is applicable to all equipment configurations.
6. Director Sequencing and Selection - This section provides programming information for use of the instruction set used in the control logic. Section 6 is applicable to all configurations of equipment.
7. Special Hardware Considerations - This section contains information on hardware features which may be of interest to the programmer. Section 7 is applicable to all configurations of equipment.
8. Error Detection and Correction - This section contains information regarding the error detecting and correcting capabilities of the hardware. Section 8 is applicable to all equipment configurations.
9. 6000 Coupler Programming - This section contains programming information for the 6000 System Coupler used in the FA719-A/B/C/D, FA722-A/B/C, and for the DT209/DT220 6000 System Coupler which can be used with the FA710.
10. 7000 Coupler Programming - This section contains programming information for the 7000 system coupler used in the FA720-A/B/C/D, FA723-A/B, and for the DT210 7000 System Coupler which can be used with the FA710.

Refer to the following manuals for additional information on the subsystem.

CONTROL DATA PUBLICATION	PUBLICATION NO.
7054/7654 Disk Storage Controller General Information Manual	60364400
7054/7654 Disk Storage Controller Operation and Programming Manual	60363900
FA710-A/B/C/D Disk Controller Customer Engineering Manual	60364000
FA719-A/B 6000 Disk Storage Controller and FA720-A/B 7000 Disk Storage Controller Customer Engineering Manual	60405500
FA719-A/B 6000 Disk Controller Subsystem Coupler Customer Engineering Manual	60416400
FA720-A/B 7000 Disk Controller System Coupler Customer Engineering Manual	60410000
DT209/DT220 6000 System Coupler Reference Customer Engineering Manual	60364100
DT210 7000 System Coupler Reference/ Customer Engineering Manual	60364200
FA722-A/B/C Disk Storage Controller DT220-C Dual Access Option DT716-A Double Density Option	60428500

CONTENTS

9. 6000 COUPLER PROGRAMMING

10. 7000 COUPLER PROGRAMMING

FIGURES

TABLES

SECTION 1

GENERAL DESCRIPTION

GENERAL DESCRIPTION

INTRODUCTION

This manual provides reference information for the FA710-A/B/C/D Disk Controller, FA719-A/B/C/D 6000 Disk Controller, FA720-A/B/C/D 7000 Disk Controller and FA 722-A/B Disk Controller. The following paragraphs describe the physical configurations of these equipments.

FA710-A/B/C/D DISK CONTROLLER

The FA710 Disk Controllers consist of two programmable asynchronous processors and a 4K memory which they share. The two processors are the subsystem processor and the control logic. Each processor has a unique instruction set and performs a unique function within the system. The FA710 does not include a system coupler. A DT209 6000 System Coupler must be added to configure the disk controller for single channel 6000 applications. For dual channel 6000 applications, the DT220-A Dual Access Option must also be used. A DT210 7000 System Coupler must be added to the FA710 to configure the disk controller for 7000 applications.

FA719-A/B/C/D DISK CONTROLLER

The FA719 6000 Disk Controller is identical to the FA710 except that it includes a 6000 system coupler and is specifically configured for single channel 6000 applications. A DT220-B Dual Access Option may be added for dual channel operations.

FA722-A/B/C 6000 DISK CONTROLLER

The FA722 Disk Controller has one functional difference between it and the FA719, the FA722 can check parity on the CYBER 170 PPU interface. Physical differences between the two are in the cabinet dimensions and in the chassis arrangement. Thus, they have different card placements. The DT220-C Second Channel Feature can be added for dual channel operations.

FA720-A/B /C/D DISK CONTROLLER

The FA720 7000 Disk Controller is identical to the FA710 except that it includes a 7000 system coupler and is specifically configured for 7000 applications.

FA723-A/B 7000 DISK CONTROLLER

The FA723 Disk Controller is functionally identical to the FA720. The difference between the FA720 and FA723 are entirely in the cabinet size, color, and weight.

Figure 1-1 is a block diagram showing the relationships between the major functional blocks of the disk controller. The diagram also provides a guide to the contents of this manual.

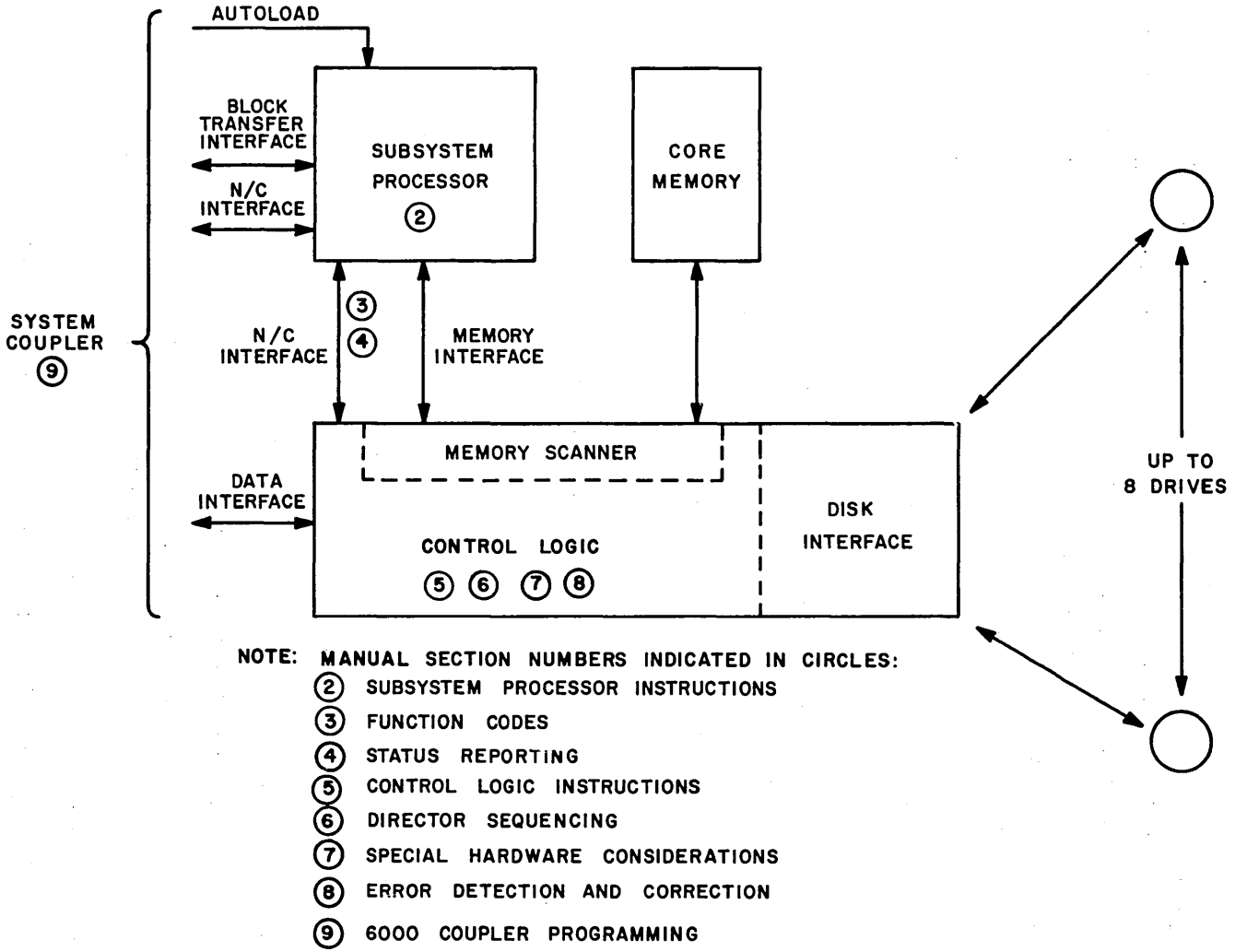


Figure 1-1. Controller Block Diagram

SUBSYSTEM PROCESSOR

The subsystem processor is a general purpose processor which uses a 16-bit instruction set and controls overall system activity. The subsystem processor communicates with the higher level processor via the system coupler and controls the activities of the control logic. Section 2 of this manual describes subsystem processor instructions and their formats. Function codes and status signals are exchanged between the subsystem processor and the control logic via the normal channel interface. Function codes are described in section 3 and status signals in section 4 of this manual.

CONTROL LOGIC

The control logic is a special purpose processor designed to control disk storage units (disk drives) and to perform other special purpose functions related to disk storage. The instruction set for the control logic varies in length from 16 to 64 bits. The instructions are called directors to distinguish them from subsystem processor instructions. Section 5 of this manual contains director formats and descriptions. Director sequences are selected by the subsystem processor dependent on the operation which is to be performed.

They are then loaded into the control logic where they control disk drive operations and perform data handling and processing. All data is streamed directly between the control logic and the higher level processor without being processed by the subsystem processor. This method provides for a higher transfer rate. Section 6 contains director sequencing information. Some of the major functional components of the control logic and their effects on internal programming are discussed in section 7. Section 8 contains information on error detection and correction which is performed by the control logic as specified by the director software.

CORE MEMORY

The core memory contains 4096 16-bit words and is shared by the subsystem processor and the control logic. Although the core memory has a potential cycle time of 750 nanoseconds, the effective cycle time is dependent upon the activities being performed by the subsystem processor and the control logic (see section 7). The subsystem processor uses the memory to control overall operation of the disk controller and to select director sequences for the control logic. The control logic references memory to obtain the director sequences which then control the operation

of the disk drives and all data handling operations. All memory references are routed through the memory scanner in the control logic so that the highest priority references are handled first.

6000 SYSTEM COUPLER

The 6000 system coupler provides an interface with one standard I/O channel from a CDC 6000 Series, CDC CYBER 70, Model 72, 73, 74, or CDC CYBER 170 family computer system. This coupler is included in the equipment configuration of the FA719-A/B/C/D 6000 Disk Controller, and FA722-A/B/C Disk Controller. The FA710 Disk Controller does not includes a system coupler. The DT209 6000 System Coupler must be added to the FA710 for 6000 applications. Although the equipments are configured in a different manner, the 6000 system coupler used in the FA719 and the DT209 6000 System Coupler are functionally identical. The FA722 coupler has one functional difference between it and the other couplers: it checks and generates parity on transfers between the PPU and coupler. Parity is used only in a CYBER 170 configuration. A switch on a coupler logic module disables the parity circuits in a 6000 or CYBER 70 configuration.

7000 SYSTEM COUPLER

The 7000 system coupler provides an interface with two pairs of I/O channels from a CDC 7000 Series or CDC CYBER 70, Model 76 computer system. This coupler is included in the equipment configuration of the FA720-A/B 7000 Disk Controller. The FA710 does not include a system coupler. The DT210 7000 System Coupler must be added to the FA710 for 7000 applications. Although the equipments are configured in a different manner, the 7000 system coupler used in the FA720, FA723, and the DT210 7000 System Coupler are functionally identical. Section 10 provides programming information which is applicable to these couplers.

FUNCTIONAL DESCRIPTION

The subsystem processor is the major control element in the disk controller. It receives function codes from the higher level processor through the system coupler and then selects a list of directors for execution by the control logic. The function codes, their format, and their method of transmission are dependent on the subsystem software and the type of higher level processor. When the subsystem processor has accepted a valid function, it selects a director sequence and informs the control logic. The control logic accesses memory directly and begins loading the director sequence into its director buffer. Execution of the director sequence is initiated by the subsystem processor. Execution of the directors results in the operations necessary to transfer data to or from the disk drive. Communication between the subsystem processor and the control logic is via the normal channel interface of the subsystem processor.

SECTION 2

SUBSYSTEM PROCESSOR INSTRUCTIONS

SUBSYSTEM PROCESSOR INSTRUCTIONS

INTRODUCTION

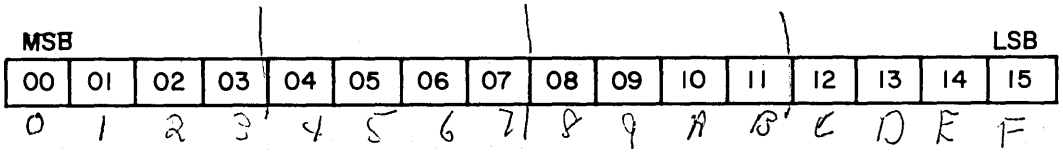
The subsystem processor provides central control of the disk storage subsystem. The subsystem processor is a small scale, internally programmed, parallel mode digital computer which contains a control section, an arithmetic section, and interface units. Since both the address and data words consist of 16 bits, the subsystem processor uses hexadecimal coding.

DATA FORMATS

Data within the arithmetic section of the subsystem processor or within memory may be treated as 16-bit words or 8-bit bytes.

WORD FORMAT

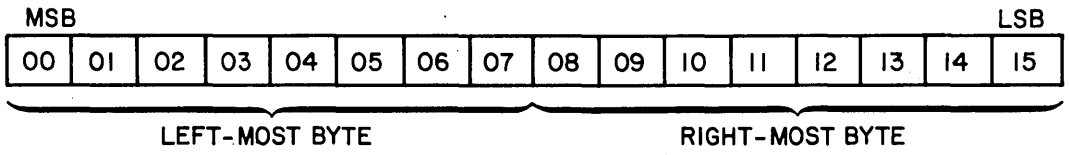
Registers and storage locations contain 16-bit words with the following bit designations.



Bit 00 is the leftmost bit and bit 15 is the rightmost bit. For signed quantities, the 1 state of bit 00 denotes a negative two's complement quantity and the 0 state denotes a positive two's complement quantity.

BYTE FORMAT

Registers and storage locations contain two 8-bit bytes within each 16-bit word with the following designations.



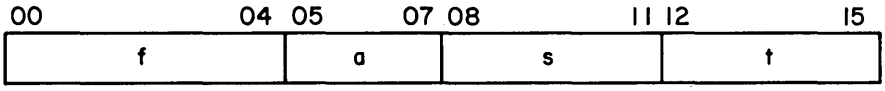
The leftmost byte occupies the leftmost bit positions, 00 through 07, and the rightmost byte occupies the rightmost bit positions, 08 through 15.

INSTRUCTION FORMATS

Two instruction code formats are used to program the subsystem processor, a no-address format and a single-address format. The no-address format is used for codes which control operations, contain an immediate operand, or perform operations with the directly addressable registers (A, B1, and B2). The single-address format is used for codes which carry an address.

FORMAT 1 - NO-ADDRESS

The format for the no-address instruction codes is as follows:



The instruction fields are defined as follows:

f = function code

a = sub-function code

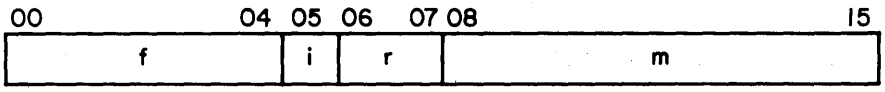
s = channel designator, instruction condition designator, or the leftmost four bits of an 8-bit immediate operand

t = bit designator, right-shift count, or the rightmost 4 bits of an 8-bit immediate operand

Format 1 instructions contain no execution address since they perform control operations, contain an immediate operand, or perform operations involving only directly addressable registers. The single exception is the LOAD FROM (A) instruction (17XX) in which the A register initially contains the base address.

FORMAT 2 - SINGLE-ADDRESS

The format for the single-address instruction codes is as follows:



The instruction fields are defined as follows:

f = function code

i, r = addressing mode designators

m = base address (before modification)

ADDRESSING MODES

Addressing modes for format 2 instructions are described with respect to the formation of the execution address, M. Parentheses are used to indicate the contents of a register or storage location. Where the rightmost 8 bits (m bits) of an instruction are used directly or arithmetically to form a 16-bit address, zeros are appended to m in the leftmost bit positions, 00 through 07. All address arithmetic is performed in two's complement mode and does not alter the state of the adder generate bit.

The addressing modes are as follows:

1. Direct Address ($i = 0$, $r = 00_2$)

   A direct address is formed by using only the rightmost 8 bits of the instruction word. This mode provides direct access to the first 256_{10} storage locations.

   M = m

2. Index B1 ($i = 0$, $r = 01_2$)

   An index B1 modified address is formed by adding the contents of the B1 register to the m bits of the instruction word.

   M = (B1) + m

3. Index B2 ($i = 0$, $r = 10_2$)

   An index B2 modified address is formed by adding the contents of the B2 register to the m bits of the instruction.

   M = (B2) + m

4. Relative Forward ($i = 0$, $r = 11_2$)

   A relative forward address is formed by adding the contents of the P register to the m bits of the instruction word.

   M = (P) + m

5. Indirect Address ($i = 1$, $r = 00_2$)

   An indirect address is formed by reading the contents of the storage location designated by the m bits of the instruction word.

   M = (m)

6. Indirect/Index B1 ($i = 1$, $r = 01_2$)

   An indirect/index B1 modified address is formed by reading the contents of the storage location designated by the m bits of the instruction word and adding the contents of the B1 register.

   M = (m) + (B1)

7. Indirect/Index B2 ($i = 1$, $r = 10_2$)

   An indirect/index B2 modified address is formed by reading the contents of the storage location designated by the m bits of the instruction word and adding the contents of the B2 register.

   M = (m) + (B2)

8. Relative Backward ($i = 1$, $r = 11_2$)

   A relative backward address is formed by subtracting the m bits of the instruction word from the contents of the P register.

   M = (P) - m

INTERNAL FLAGS

CONDITION BIT

The condition bit is an internal flag that may be altered and sensed by program means. In the 1 state this flag is referred to as condition true and in the 0 state it is referred to as condition false. The following instructions are capable of altering the state of this flag.

- Set condition equal: internal tests (08XX)
- Set condition equal: bit t of channel s (09XX)
- Test index B1, no address (14XX)
- Test index B2, no address (15XX)
- Test index B1 (30XX through 37XX)
- Test index B2 (38XX through 3FXX)
- Input block transfer (F0XX through F7XX)
- Output block transfer (F8XX through FFXX)

ADDER GENERATE BIT

The adder generate bit constitutes the 17th output bit of the adder and reflects the generation of an end-off carry following operand arithmetic for the following instructions.

- Add no address (10XX)
- Subtract no address (11XX)
- Add (60XX through 67XX)
- Subtract (68XX through 6FXX)
- Replace add (80XX through 87XX)
- Replace add one (88XX through 8FXX)

The adder generate bit does not convey any information concerning a conventional arithmetic overflow. During multiple precision arithmetic operations, this flag provides an indication of register overflow when adding like-signed operands. When the flag is absent, it indicates register overflow when subtracting unlike-signed operands.

INSTRUCTIONS

The following paragraphs describe the format 1 and format 2 instructions used in programming the subsystem processor. The abbreviations used in the descriptions have been previously defined. In addition, the subscript i is used to denote initial and f to denote final, and both refer to the contents of registers or storage locations.

Instructions, which serve no purpose except to increment the contents of the P register by one, are referred to as resulting in no-operation.

Instruction words in which bit positions are unused must have the unused bits in the 0 state. When these bit positions are in the 1 state, the results of the instructions are undefined.

In the case of the SELECTIVE STOP instruction (00XX), the s and t fields are unused to the extent that they do not participate in the execution of the instruction. These fields may be nonzero and are defined as having no effect. In this instruction, these fields may be used for identification, temporary storage, or other program purposes.

FORMAT 1 INSTRUCTIONS

Table 2-1 lists the format 1 instructions used to program the subsystem processor. The hexadecimal codes for the f and a fields are given for each instruction. The s and t fields are designated by an X when the bits are used and by a 0 when they are not used. The individual instruction descriptions explain the coding of the s and t fields. In addition, the table contains the mnemonics and execution times for each instruction.

TABLE 2-1. FORMAT 1 HEXADECIMAL INSTRUCTION CODES

Instruction	Hex Code		Execution Time*
Selective Stop	00	XX	1
Selective Set Bit t of A	01	0X	1
Selective Clear Bit t of A	02	0X	1
Selective Complement Bit t of A	03	0X	1
Count of Leading Zeros in A_i to A_f	04	00	1
Shift A Right, t Places	05	XX	1
Transfer A to B1;	06	XX	1
Transfer A to B2;	07	XX	1
Set Condition Equal : Internal Tests	08	XX	1
Set Condition Equal : Bit t of Channel s	09	XX	1
Input to A From Channel s	0C	X0	1
Set Channel s from A	0D	XX	1
Clear Channel s from A	0E	XX	1
Transfer A to Channel s	0F	XX	1
Add No Address	10	XX	1
Subtract No Address	11	XX	1
Exclusive OR No Address	12	XX	1
Logical Product No Address	13	XX	1
Test Index B1 No Address	14	XX	1
Test Index B2 No Address	15	XX	1
Load A Complement No Address	16	XX	1
Load From (A)	17	XX	2

* Instruction times in number of storage reference cycles.

** See instruction description for mnemonic.

Selective Stop	00 XX

The execution of this instruction is dependent on the state of the NOP-00 signal which appears on the input of the station control interface. When the NOP-00 signal is in the 1 state, this instruction results in a no-operation. When the NOP-00 signal is in the 0 state, this instruction results in an immediate halt of program execution.

Once halted, the program execution may be resumed with the reading of the next instruction at P+1 through the use of additional control inputs on the station control interface.

Selective Set Bit t of A	01 0X

This instruction unconditionally sets bit t of the A register, where the 4-bit designator t specifies one of the 16 bit positions in A. The remaining 15 bits of A are left unchanged.

If bit t of A_i is already in the set state (1), this instruction effectively results in a no-operation.

Selective Clear Bit t of A	02 0X

This instruction unconditionally clears bit t of the A register, where the 4-bit designator t specifies one of the 16 bit positions in A. The remaining 15 bit positions of A are left unchanged.

If bit t of A_i is already in the clear state (0), this instruction effectively results in a no-operation.

Selective Complement Bit t of A	03 0X

This instruction complements bit t of the A register, where the 4-bit designator t specifies one of the 16 bit positions in A. The remaining 15 bit positions of A are left unchanged.

The complement operation is performed so that when bit t of A_i is set, bit t of A_f is clear and when bit t of A_i is clear, bit t of A_f is set.

Count of Leading Zeros in A_i to A_f	04 00

This instruction scans the contents of A_i from left to right, and transfers the count of leading zeros into A_f as a 5-bit right-justified quantity. The leftmost 11 bits of A_f are cleared.

Shift A Right, t Places	05 0X and 05 8X

This instruction shifts the contents of A to the right t bit positions, where t is a 4-bit shift count.

The shift may be circular or end-off. When bit 08 of the instruction is in the 1 state, the shift is circular with the rightmost bits shifted end-around into the leftmost bit positions. When bit 08 is in the 0 state, the shift is end-off with zeros inserted into the leftmost bit positions of A.

When t = 0, this instruction effectively results in a no-operation.

Hexadecimal Code	Operation
050X	Shift Right Circular
058X	Shift Right Open

Transfer A to B1; (A) + s and t → B1	06 XX

This instruction adds the rightmost 8 bits of the instruction word, with zeros extended, to the contents of the A register and transfers the sum to the B1 register.

The state of the adder generate bit is unaltered by the execution of this instruction.

Transfer A to B2; (A) + s and t → B2	07 XX

This instruction adds the rightmost 8 bits of the instruction word, with zeros extended, to the contents of the A register and transfers the sum to the B2 register.

The state of the adder generate bit is unaltered by the execution of this instruction.

Set Condition Equal: Internal Tests	08 0X through 08 7X

This instruction forces the state of the condition bit to reflect the state of selected internal tests. The s field designates the internal test(s) to be performed. The t field specifies which bit of the A register is to be tested when the s field designates an A register bit test. A 080X code forces the condition bit false.

For the following code descriptions, X and Z are defined as follows:

X = any hexadecimal digit
Z = 1 or 3 or 5 or 7

421

The following codes force the condition bit true (to the 1 state) for the following listed reasons.

Shift Count

081X	The A register bit specified by the t field is a 1.
082X	The adder generate bit is a 1.
083X	The A register bit specified by the t field is a 1 or the adder generate bit is a 1.
084X	The A register contains an odd number of 1 bits.
085X	The A register bit specified by the t field is a 1 or the A register contains an odd number of 1 bits.
086X	The adder generate bit is a 1 or the A register contains an odd number of 1 bits.
087X	The A register bit specified by the t field is a 1, the adder generate bit is a 1, or the A register contains an odd number of 1 bits.

For 08ZX codes, the t field specifies the A register bit to be tested as follows:

08Z0	A register bit 15 is to be tested
08Z1	A register bit 14 is to be tested
08Z2	A register bit 13 is to be tested
08Z3	A register bit 12 is to be tested
08Z4	A register bit 11 is to be tested
08Z5	A register bit 10 is to be tested
08Z6	A register bit 09 is to be tested
08Z7	A register bit 08 is to be tested
08Z8	A register bit 07 is to be tested
08Z9	A register bit 06 is to be tested
08ZA	A register bit 05 is to be tested
08ZB	A register bit 04 is to be tested
08ZC	A register bit 03 is to be tested
08ZD	A register bit 02 is to be tested
08ZE	A register bit 01 is to be tested
08ZF	A register bit 00 is to be tested

Set Condition Equal: Bit t, Channel s	09 XX

This instruction forces the state of the condition bit to reflect the state of the selected bit on the selected normal input channel.

The condition bit is forced true (1) if bit t of normal input channel s is true (1). The condition bit is forced false (0) if bit t of normal input channel s is false (0).

The 4-bit designator s specifies one of 16 possible normal input channels and the 4-bit designator t specifies one of 16 bit positions to be tested within the selected channel. If the selected input channel is not physically present, the condition bit is forced true.

Input to A from Channel s	0C X0

This instruction transfers a word from normal input channel s to A.

The 4-bit designator s specifies 1 of 16 possible normal input channels. If the selected channel is not physically present, A_f will contain all ones ($FFFF_{16}$).

Set Channel s from A	0D X0 or 0D X8

This instruction sets bits on normal output channel s according to the contents of the A register.

If bit 12 of the instruction is a 0, this instruction sets bits on normal output channel s where corresponding bits are present in the A register. Where zeros are present in the A register, corresponding bits on the selected channel are left unchanged.

If bit 12 of the instruction is a 1, this instruction sets bits on normal output channel s where corresponding zeros are present in the A register. Where ones are present in the A register, corresponding bits on the selected channel are left unchanged.

The 4-bit designator s specifies 1 of 16 possible normal output channels. If the selected channel is not physically present, this instruction effectively results in a no-operation.

Clear Channel s from A	0E X0 or 0E X8

This instruction clears bits on normal output channel s according to the contents of the A register.

If bit 12 of the instruction is a 0, this instruction clears bits on normal output channel s where corresponding zeros are present in the A register. Where ones are present in the A register, corresponding bits on the selected channel are left unchanged.

If bit 12 of the instruction is a 1, this instruction clears bits on normal output channel s where corresponding ones are present in the A register. Where zeros are present in the A register, corresponding bits on the selected channel are left unchanged.

The 4-bit designator s specifies 1 of 16 possible normal output channels. If the selected channel is not physically present, this instruction effectively results in a no-operation.

Transfer A to Channel s	0F X0 or 0F X8

This instruction transfers the contents of the A register either directly or in one's complement mode to normal output channel s.

If bit 12 of the instruction is a 0, this instruction force transfers the contents of the A register directly to normal output channel s.

If bit 12 of the instruction is a 1, this instruction force transfers the one's complement of the contents of the A register to normal output channel s.

The 4-bit designator s specifies 1 of 16 possible normal output channels. If the selected channel is not physically present, this instruction results in a no-operation.

Hexadecimal Code	Operation
0FX0	Transfer (A) to channel s
0FX8	Transfer (A) complement to channel s

Add No Address	10 XX

This instruction adds the rightmost 8 bits of the instruction word, with zeros extended, to the contents of A_i and transfers the sum to A_f.

The adder generate bit receives the end-off carry from the adder at the time the result is transferred to A_f.

Subtract No Address	11 XX

This instruction subtracts the rightmost bits of the instruction word, with zeros extended, from the contents of A_i and transfers the difference to A_f.

The adder generate bit receives the end-off carry from the adder at the time the result is transferred to A_f.

Exclusive OR No Address	12 XX

This instruction performs an exclusive OR with the rightmost 8 bits of the instruction. word and the rightmost 8 bits of A_i, and places the results in A_f. The leftmost 8 bits of A_i remain unchanged.

The exclusive OR operation is performed according to the following truth table.

Bit n of A_i	Bit n Immediate Operand	Bit n of A_f
0	0	0
0	1	1
1	0	1
1	1	0

Logical Product No Address	13 XX

This instruction performs the logical product with the rightmost 8 bits of the instruction word and the rightmost 8 bits of A_i, and places the results in A_f. The leftmost 8 bits of A_f are cleared.

The logical product operation is performed according to the following truth table.

Bit n of A_i	Bit n Immediate Operand	Bit n of A_f
0	0	0
0	1	0
1	0	0
1	1	1

Test Index B1 No Address	14 XX

This instruction compares the rightmost 8 bits of the instruction word with the rightmost 8 bits initially contained in the B1 index register.

If the quantities are equal, the condition bit is forced true (1) and the contents of $B1_i$ are left unchanged.

If the quantities are not equal, the condition bit is forced false (0) and the contents of $B1_i$ are increased by one and transferred to $B1_f$.

Test Index B2 No Address	15 XX

This instruction compares the rightmost 8 bits of the instruction word with the rightmost 8 bits initially contained in the B2 index register.

If the quantities are equal, the condition bit is forced true (1) and the contents of $B2_i$ are left unchanged.

If the quantities are not equal, the condition bit is forced false (0) and the contents of $B2_i$ are increased by one and transferred to $B2_f$.

Load A Complement No Address	16 XX

This instruction performs the two's complement on the rightmost 8 bits of the instruction word and transfers the result, with ones extended, into the A register.

Load from (A); ((A_i) + s and t) → A_f	17 XX

This instruction transfers the contents of the execution address M to the A register.

The execution address M is formed by adding the rightmost 8 bits of the instruction word to the contents of A_i.

FORMAT 2 INSTRUCTIONS

Table 2-2 lists the format 2 instructions used to program the subsystem processor. The hexadecimal codes for the f, i, and r fields for each instruction are given. A description of the addressing modes for the m field of format 2 instructions can be found at the beginning of this section. Table 2-3 lists the execution times for format 2 instructions.

Enter A With Address	18 XX through 1F XX

This instruction transfers M (address after modification) to the A register. The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
18 XX	m → A
19 XX	m + (B1) → A
1A XX	m + (B2) → A
1B XX	m + (P) → A
1C XX	(m) → (A)
1D XX	(m) + (B1) → A
1E XX	(m) + (B2) → A
1F XX	-m + (P) → A

Enter B1 With Address	20 XX through 27 XX

This instruction transfers M (address after modification) to the B1 index register. The following table lists the codes for the eight addressing modes used with this instruction.

TABLE 2-2. FORMAT 2 HEXADECIMAL INSTRUCTION CODES

Instruction	Addressing Mode							
	Direct	Index B1	Index B2	Relative Forward	Indirect	Indirect Index B1	Indirect Index B2	Relative Backward
Enter A with Address	18	19	1A	1B	1C	1D	1E	1F
Enter B1 with Address	20	21	22	23	24	25	26	27
Enter B2 with Address	28	29	2A	2B	2C	2D	2E	2F
Test Index B1	30	31	32	33	34	35	36	37
Test Index B2	38	39	3A	3B	3C	3D	3E	3F
Load A	40	41	42	43	44	45	46	47
Load A Complement	48	49	4A	4B	4C	4D	4E	4F
Load Left-Most Byte	50	51	52	53	54	55	56	57
Load Right-Most Byte	58	59	5A	5B	5C	5D	5E	5F
Add	60	61	62	63	64	65	66	67
Subtract	68	69	6A	6B	6C	6D	6E	6F
Exclusive OR	70	71	72	73	74	75	76	77
Logical Product	78	79	7A	7B	7C	7D	7E	7F
Replace Add	80	81	82	83	84	85	86	87
Replace Add One	88	89	8A	8B	8C	8D	8E	8F
Replace Left-Most Byte	90	91	92	93	94	95	96	97
Replace Right-Most Byte	98	99	9A	9B	9C	9D	9E	9F
Store	A0	A1	A2	A3	A4	A5	A6	A7
Store Zeros	A8	A9	AA	AB	AC	AD	AE	AF
Destructive Load	B0	B1	B2	B3	B4	B5	B6	B7
Unconditional Jump	B8	B9	BA	BB	BC	BD	BE	BF
A Zero Jump	C0	C1	C2	C3	C4	C5	C6	C7
A Nonzero Jump	C8	C9	CA	CB	CC	CD	CE	CF
A Positive Jump	D0	D1	D2	D3	D4	D5	D6	D7
A Negative Jump	D8	D9	DA	DB	DC	DD	DE	DF
Condition True Jump	E0	E1	E2	E3	E4	E5	E6	E7
Condition False Jump	E8	E9	EA	EB	EC	ED	EE	EF
Input Block Transfer	F0	F1	F2	F3	F4	F5	F6	F7
Output Block Transfer	F8	F9	FA	FB	FC	FD	FE	FF

TABLE 2-3. FORMAT 2 INSTRUCTIONS EXECUTION TIME

Instruction	Execution Time*							
	Direct	Index B1	Index B2	Relative Forward	Indirect	Indirect Index B1	Indirect Index B2	Relative Backward
Enter A with Address	1	1	1	1	2	2	2	1
Enter B1 with Address	1	1	1	1	2	2	2	1
Enter B2 with Address	1	1	1	1	2	2	2	1
Test Index B1	2	2	2	2	3	3	3	2
Test Index B2	2	2	2	2	3	3	3	2
Load A	2	2	2	2	3	3	3	2
Load A Complement	2	2	2	2	3	3	3	2
Load Left-Most Byte	2	2	2	2	3	3	3	2
Load Right-Most Byte	2	2	2	2	3	3	3	2
Add	2	2	2	2	3	3	3	2
Subtract	2	2	2	2	3	3	3	2
Exclusive OR	2	2	2	2	3	3	3	2
Logical Product	2	2	2	2	3	3	3	2
Replace Add	3	3	3	3	4	4	4	3
Replace Add One	3	3	3	3	4	4	4	3
Replace Left-Most Byte	3	3	3	3	4	4	4	3
Replace Right-Most Byte	3	3	3	3	4	4	4	3
Store	2	2	2	2	3	3	3	2
Store Zeros	2	2	2	2	3	3	3	2
Destructive Load	3	3	3	3	4	4	4	3
Unconditional Jump	1	1	1	1	2	2	2	1
A Zero Jump	1	1	1	1	2**	2**	2**	1
A Nonzero Jump	1	1	1	1	2**	2**	2**	1
A Positive Jump	1	1	1	1	2**	2**	2**	1
A Negative Jump	1	1	1	1	2**	2**	2**	1
Condition True Jump	1	1	1	1	2**	2**	2**	1
Condition False Jump	1	1	1	1	2**	2**	2**	1
Input Block Transfer	1+k	1+k	1+k	1+k	2+k	2+k	2+k	1+k
Output Block Transfer	1+k	1+k	1+k	1+k	2+k	2+k	2+k	1+k

* Instruction times in number of storage reference cycles.

** The execution time for block transfer instructions is dependent upon the number of transfers to be performed as well as the time required for each transfer. K in the above table is defined as follows:

$k = n(1+c)$ where:

n = number of transfers
1 = one storage reference cycle
c = a constant determined by the characteristics of the coupler attached to the coupler interface of the subsystem processor.

Hexadecimal Code	Operation
20 XX	m → B1
21 XX	m + (B1) → B1
22 XX	m + (B2) → B1
23 XX	m + (P) → B1
24 XX	(m) → B1
25 XX	(m) + (B1) → B1
26 XX	(m) + (B2) → B1
27 XX	-m + (P) → B1

Enter B2 With Address	28 XX through 2F XX

This instruction transfers M (address after modification) to the B2 index register. The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
28 XX	m → B2
29 XX	m + (B1) → B2
2A XX	m + (B2) → B2
2B XX	m + (P) → B2
2C XX	(m) → B2
2D XX	(m + (B1)) → B2
2E XX	(m + (B2)) → B2
2F XX	-m + (P) → B2

Test Index B1	30 XX through 37 XX

This instruction compares the contents of M (address after modification) with the contents of $B1_i$. If equal, the condition bit is forced true (1 state) and the contents of $B1_i$ are not changed. If unequal, the condition bit is forced false (0 state) and the contents of $B1_i$ are incremented by one and transferred to $B1_f$.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Execution Address
30 XX	m
31 XX	m + (B1)
32 XX	m + (B2)
33 XX	m + P
34 XX	(m)
35 XX	(m) + (B1)
36 XX	(m) + (B2)
37 XX	-m + (P)

Test Index B2	38 XX through 3F XX

This instruction compares the contents of M (address after modification) with the contents of $B2_i$. If equal, the condition bit is forced true (1 state) and the contents of $B2_i$ are not changed. If unequal, the condition bit is forced false (0 state) and the contents of $B2_i$ are incremented by one and transferred to $B2_f$.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Execution Address
38 XX	m
39 XX	m + (B1)
3A XX	m + (B2)
3B XX	m + (P)
3C XX	(m)
3D XX	(m) + (B1)
3E XX	(m) + (B2)
3F XX	-m + (P)

Load A	40 XX through 47 XX

This instruction transfers the contents of M (address after modification) to the A register.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
40 XX	$m \rightarrow A_f$
41 XX	$m + (B1) \rightarrow A_f$
42 XX	$m + (B2) \rightarrow A_f$
43 XX	$m + (P) \rightarrow A_f$
44 XX	$(m) \rightarrow A_f$
45 XX	$(m) + (B1) \rightarrow A_f$
46 XX	$(m) + (B2) \rightarrow A_f$
47 XX	$-m + (P) \rightarrow A_f$

Load A Complement	48 XX through 4F XX

This instruction transfers the two's complement of the contents of M (address after modification) to the A register.

The two's complement of 0000_{16} and 8000_{16} leaves the number unaltered. The hardware does not detect these as exceptions.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
48 XX	$m \rightarrow A_f$
49 XX	$m + (B1) \rightarrow A_f$
4A XX	$m + (B2) \rightarrow A_f$
4B XX	$m + (P) \rightarrow A_f$
4C XX	$(m) \rightarrow A_f$
4D XX	$(m) + (B1) \rightarrow A_f$
4E XX	$(m) + (B2) \rightarrow A_f$
4F XX	$-m + (P) \rightarrow A_f$

Load Left-Most Byte	50 XX through 57 XX

This instruction transfers the leftmost byte from the contents of M (address after modification) to the rightmost byte position in the A register and clears the leftmost byte in the A register.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
50 XX	$m \rightarrow A_f$
51 XX	$m + (B1) \rightarrow A_f$
52 XX	$m + (B2) \rightarrow A_f$
53 XX	$m + (P) \rightarrow A_f$
54 XX	$(m) \rightarrow A_f$
55 XX	$(m) + (B1) \rightarrow A_f$
56 XX	$(m) + (B2) \rightarrow A_f$
57 XX	$-m + (P) \rightarrow A_f$

Load Right-Most Byte	58 XX through 5F XX

This instruction transfers the rightmost byte from the contents of M (address after modification) to the rightmost byte position in the A register and clears the leftmost byte in the A register.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
58 XX	$m \rightarrow A_f$
59 XX	$m + (B1) \rightarrow A_f$
5A XX	$m + (B2) \rightarrow A_f$
5B XX	$m + (P) \rightarrow A_f$
5C XX	$(m) \rightarrow A_f$
5D XX	$(m) + (B1) \rightarrow A_f$
5E XX	$(m) + (B2) \rightarrow A_f$
5F XX	$-m + (P) \rightarrow A_f$

Add	60 XX through 67 XX

This instruction adds the contents of memory location M (address after modification) to the contents of the A register (A_i) and transfers the sum to the A register (A_f). The adder generate bit receives the end-off carry.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
60 XX	$m + (A_i) \rightarrow A_f$
61 XX	$m + (B1) + (A_i) \rightarrow A_f$
62 XX	$m + (B2) + (A_i) \rightarrow A_f$
63 XX	$m + (P) + (A_i) \rightarrow A_f$
64 XX	$(m) + (A_i) \rightarrow A_f$
65 XX	$(m) + (B1) + (A_i) \rightarrow A_f$
66 XX	$(m) + (B2) + (A_i) \rightarrow A_f$
67 XX	$-m + (P) + (A_i) \rightarrow A_f$

Subtract	68 XX through 6F XX

This instruction subtracts the contents of memory location M (address after modification) from the contents of the A register (A_i) and transfers the difference to the A register (A_f). The adder generate bit receives the end-off carry.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
68 XX	$m - (A_i) \rightarrow A_f$
69 XX	$m + (B1) - (A_i) \rightarrow A_f$
6A XX	$m + (B2) - (A_i) \rightarrow A_f$
6B XX	$m + (P) - (A_i) \rightarrow A_f$
6C XX	$(m) - (A_i) \rightarrow A_f$
6D XX	$(m) + (B1) - (A_i) \rightarrow A_f$
6E XX	$(m) + (B2) - (A_i) \rightarrow A_f$
6F XX	$-m + (P) - (A_i) \rightarrow A_f$

Exclusive OR	70 XX through 77 XX

This instruction performs an exclusive-OR with the contents of memory location M (address after modification) and the contents of the A register (A_i); the result is transferred to the A register (A_f).

The exclusive-OR operation is performed according to the following truth table.

Bit n of A_i	Bit n of (M)	Bit n of A_f
0	0	0
0	1	1
1	0	1
1	1	0

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
70 XX	$m + (A_i) \rightarrow A_f$
71 XX	$m + (B1) + A_i \rightarrow A_f$
72 XX	$m + (B2) + A_i \rightarrow A_f$
73 XX	$m + (P) + A_i \rightarrow A_f$
74 XX	$(m) + A_i \rightarrow A_f$
75 XX	$(m) + (B1) + A_i \rightarrow A_f$
76 XX	$(m) + (B2) + A_i \rightarrow A_f$
77 XX	$-m + (P) + A_i \rightarrow A_f$

Logical Product	78 XX through 7F XX

This instruction performs the logical product with the contents of memory location M (address after modification) and the contents of the A register (A_i); the results are transferred to the A register (A_f).

The logical product operation is performed according to the following truth table.

Bit n of A_i	Bit n of (M)	Bit n of A_f
0	0	0
0	1	0
1	0	0
1	1	1

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
78 XX	$m\ (A_i) \rightarrow A_f$
79 XX	$m + (B1)\ (A_i) \rightarrow A_f$
7A XX	$m + (B2)\ (A_i) \rightarrow A_f$
7B XX	$m + (P)\ (A_i) \rightarrow A_f$
7C XX	$(m)\ (A_i) \rightarrow A_f$
7D XX	$(m) + (B1)\ (A_i) \rightarrow A_f$
7E XX	$(m) + (B2)\ (A_i) \rightarrow A_f$
7F XX	$-m + (P)\ (A_i) \rightarrow A_f$

Replace Add	80 XX through 87 XX

This instruction adds the contents of memory location M_i (initial address after modification) to the contents of the A register (A_i) and transfers the sum to the A register (A_f) and to M_f.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
80 XX	$m + (A_i) \rightarrow A_f, M_f$
81 XX	$m + (B1) + (A_i) \rightarrow A_f, M_f$
82 XX	$m + (B2) + (A_i) \rightarrow A_f, M_f$
83 XX	$m + (P) + (A_i) \rightarrow A_f, M_f$
84 XX	$(m) + (A_i) \rightarrow A_f, M_f$
85 XX	$(m) + (B1) + (A_i) \rightarrow A_f, M_f$
86 XX	$(m) + (B1) + (A_i) \rightarrow A_f, M_f$
87 XX	$-m + (P) + (A_i) \rightarrow A_f, M_f$

Replace Add One	88 XX through 8F XX

This instruction adds one to the memory location M_i (address after modification) and transfers the result to M_f and the A register (A_f). The adder generate bit receives the end-off carry.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
88 XX	$m + 1 \rightarrow A_f \rightarrow M_f$
89 XX	$m + (B1) + 1 \rightarrow A_f \rightarrow M_f$
8A XX	$m + (B2) + 1 \rightarrow A_f \rightarrow M_f$
8B XX	$m + (P) + 1 \rightarrow A_f \rightarrow M_f$
8C XX	$(m) + 1 \rightarrow A_f \rightarrow M_f$
8D XX	$(m) + (B1) + 1 \rightarrow A_f \rightarrow M_f$
8E XX	$(m) + (B2) + 1 \rightarrow A_f \rightarrow M_f$
8F XX	$-m + (P) + 1 \rightarrow A_f \rightarrow M_f$

Replace Left-Most Byte	90 XX through 97 XX

This instruction stores the rightmost byte of the contents of the A register (A_i) into the leftmost byte position of storage location M (address after modification). The rightmost byte of M is left unchanged.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
90 XX	$A_{08-15} \rightarrow m$
91 XX	$A_{08-15} \rightarrow m + (B1)$
92 XX	$A_{08-15} \rightarrow m + (B2)$
93 XX	$A_{08-15} \rightarrow m + (P)$
94 XX	$A_{08-15} \rightarrow (m)$
95 XX	$A_{08-15} \rightarrow (m) + (B1)$
96 XX	$A_{08-14} \rightarrow (m) + (B2)$
97 XX	$A_{08-15} \rightarrow -m + (P)$

Replace Right-Most Byte	98 XX through 9F XX

This instruction stores the rightmost byte of the contents of the A register (A_i) into the rightmost byte position of storage location M (address after modification). The leftmost byte of M is unchanged.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
98 XX	$A_{08-15} \rightarrow m$
99 XX	$A_{08-15} \rightarrow m + (B1)$
9A XX	$A_{08-15} \rightarrow m + (B2)$
9B XX	$A_{08-15} \rightarrow m + (P)$
9C XX	$A_{08-15} \rightarrow (m)$
9D XX	$A_{08-15} \rightarrow (m) + (B1)$
9E XX	$A_{08-15} \rightarrow (m) + (B2)$
9F XX	$A_{08-15} \rightarrow -m + (P)$

Store	A0 XX through A7 XX

This instruction stores the contents of the A register at memory location M (address after modification).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
A0 XX	(A) → m
A1 XX	(A) → m + (B1)
A2 XX	(A) → m + (B2)
A3 XX	(A) → m + (P)
A4 XX	(A) → (m)
A5 XX	(A) → (m) + (B1)
A6 XX	(A) → (m) + (B2)
A7 XX	(A) → -m + (P)

Store Zeros	A8 XX through AF XX

This instruction stores all zeros at storage location M (address after modification).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
A8 XX	0 → m
A9 XX	0 → m + (B1)
AA XX	0 → m + (B2)
AB XX	0 → m + (P)
AC XX	0 → (m)
AD XX	0 → (m) + (B1)
AE XX	0 → (m) + (B2)
AF XX	0 → -m + (P)

Destructive Load	B0 XX through B7 XX

This instruction transfers the contents of storage location M (address after modification) to the A register and stores all zeros at storage location M.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
B0 XX	m → A; 0 → M_f
B1 XX	m + (B1) → A; 0 → M_f
B2 XX	m + (B2) → A; 0 → M_f
B3 XX	m + (P) → A; 0 → M_f
B4 XX	(m) → A; 0 → M_f
B5 XX	(m) + (B1) → A; 0 → M_f
B6 XX	(m) + (B2) → A; 0 → M_f
B7 XX	-m + (P) → A; 0 → M_f

Unconditional Jump	B8 XX through BF XX

This instruction unconditionally performs a jump exit to the specified storage location M (address after modification).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
B8 XX	j → m
B9 XX	j → m + (B1)
BA XX	j → m + (B2)
BB XX	j → m + (P)
BC XX	j → (m)
BD XX	j → (m) + (B1)
BE XX	j → (m) + (B2)
BF XX	j → -m + (P)

A Zero Jump	C0 XX through C7 XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the A register contains all zeros (0000_{16}).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
C0 XX	If (A) = 0, j → m
C1 XX	If (A) 0, j → m + (B1)
C2 XX	If (A) 0, j → m + (B2)
C3 XX	If (A) = 0, j → m + (P)
C4 XX	If (A) 0, j → (m)
C5 XX	If (A) = 0, j → (m) + (B1)
C6 XX	If (A) = 0, j → (m) + (B2)
C7 XX	If (A) = 0, j → -m + (P)

A Non Zero Jump	C8 XX through CF XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the A register does not contain all zeros (0000_{16}).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
C8 XX	If (A) ≠ 0, j → m
C9 XX	If (A) ≠ 0, j → m + (B1)
CA XX	If (A) ≠ 0, j → m + (B2)
CB XX	If (A) ≠ 0, j → m + (P)
CC XX	If (A) ≠ 0, j → (m)
CD XX	If (A) ≠ 0, j → (m) + (B1)
CE XX	If (A) ≠ 0, j → (m) + (B2)
CF XX	If (A) ≠ 0, j → -m + (P)

A Positive Jump	D0 XX through D7 XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the contents of the A register are equal to or greater than zero ($\geq 0000_{16}$).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
D0 XX	If (A) $\geq$ 0, j → m
D1 XX	If (A) $\geq$ 0, j → m + (B1)
D2 XX	If (A) $\geq$ 0, j → m + (B2)
D3 XX	If (A) $\geq$ 0, j → m + (P)
D4 XX	If (A) $\geq$ 0, j → (m)
D5 XX	If (A) $\geq$ 0, j → (m) + (B1)
D6 XX	If (A) $\geq$ 0, j → (m) + (B2)
D7 XX	If (A) $\geq$ 0, j → -m + (P)

A Negative Jump	D8 XX through DF XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the contents of the A register is less than zero ($< 0000_{16}$).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
D8 XX	If (A) < 0, j → m
D9 XX	If (A) < 0, j → m + (B1)
DA XX	If (A) < 0, j → m + (B2)
DB XX	If (A) < 0, j → m + (P)
DC XX	If (A) < 0, j → (m)
DD XX	If (A) < 0, j → (m) + (B1)
DE XX	If (A) < 0, j → (m) + (B2)
DF XX	If (A) < 0, j → -m + (P)

Condition True Jump	E0 XX through E7 XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the condition bit (internal flag) is true (1 state).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
E0 XX	If CB = 1, j → m
E1 XX	If CB = 1, j → m + (B1)
E2 XX	If CB = 1, j → m + (B2)
E3 XX	If CB = 1, j → m + (P)
E4 XX	If CB = 1, j → (m)
E5 XX	If CB = 1, j → (m) + (B1)
E6 XX	If CB = 1, j → (m) + (B2)
E7 XX	If CB = 1, j → -m + (P)

Condition False Jump	E8 XX through EF XX

This instruction performs a jump exit to the specified storage location M (address after modification) if the condition bit (internal flag) is false (0 state).

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Operation
E8 XX	If CB = 0, j → m
E9 XX	If CB = 0, j → m + (B1)
EA XX	If CB = 0, j → m + (B2)
EB XX	If CB = 0, j → m + (P)
EC XX	If CB = 0, j → (m)
ED XX	If CB = 0, j → (m) + (B1)
EE XX	If CB = 0, j → (m) + (B2)
EF XX	If CB = 0, j → -m + (P)

Input Block Transfer	F0 XX through F7 XX

This instruction transfers data from the coupler interface into storage beginning at storage location M (address after modification). This instruction requires that the initial contents of the A register (A_i) be set to the two's complement of the total number of words to be transferred.

If the READY signal on the coupler interface is a 0 at the time this instruction is read from storage, the condition bit is forced false (0 state) and an immediate normal exit is performed with the initial contents of the A register (A_i) left unchanged.

If the READY signal is a 1 at the time this instruction is read from storage, the execution address (M) is formed and then transferred to the S register. As each word is supplied to the coupler interface, a storage reference cycle is performed to store the word at the address specified by the contents of the S register. At the same time, the contents of the A register are incremented by one and the S register is incremented by one.

The instruction terminates the block transfer, performs a normal exit, and forces the condition bit true (1 state) when the contents of the A register are equal to zero.

The instruction will also terminate the block transfer, perform a normal exit, and force the condition bit false (0 state) if a TERMINATE signal is received on the coupler interface.

If the initial contents of the A register (A_i) are equal to zero (0000_{16}), the number of words to be transferred is translated as $65,536_{10}$.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Execution Address
F0 XX	m
F1 XX	m + (B1)
F2 XX	m + (B2)
F3 XX	m + (P)
F4 XX	(m)
F5 XX	(m) + (B1)
F6 XX	(m) + (B2)
F7 XX	-m + (P)

Output Block Transfer	F8 XX through FF XX

This instruction transfers data from storage to the coupler interface beginning at storage location M (address after modification). This instruction requires that the initial contents of the A register (A_i) be set to the two's complement of the total number of words to be transferred.

If the READY signal on the coupler interface is a 0 at the time this instruction is read from storage, the condition bit is forced false (0 state) and an immediate normal exit is performed with the initial contents of the A register (A_i) left unchanged.

If the READY signal on the coupler interface is a 1 at the time this instruction is read from storage, the execution address (m) is formed and transferred to the S register. A storage reference cycle is performed to read each output word from the address location contained in the S register, increment the contents of the A register by one, and increment the S register by one. Each output word is present on the coupler interface until either a REPLY signal or a TERMINATE signal is received on the interface.

This instruction terminates the block transfer, performs a normal exit, and forces the condition bit true (1 state) at the time the last output word is accepted on the coupler interface.

This instruction will also terminate the block transfer, perform a normal exit, and force the condition bit false (0 state) at the time an output word is rejected by a TERMINATE signal on the coupler interface.

If the initial contents of the A register (A_i) are equal to zero (0000_{16}), the number of words to be transferred is translated as $65,536_{10}$.

The following table lists the codes for the eight addressing modes used with this instruction.

Hexadecimal Code	Execution Address
F8 XX	m
F9 XX	m + (B1)
FA XX	m + (B2)
FB XX	m + (P)
FC XX	(m)
FD XX	(m) + (B1)
FE XX	(m) + (B2)
FF XX	-m + (P)

SECTION 3

FUNCTION CODES

FUNCTION CODES

NORMAL CHANNEL INTERFACE

The normal channel interface of the subsystem processor is tied directly to the control logic and the system coupler. These interface lines are defined in the FA710 Disk Controller Customer Engineering Manual, the FA722-A/B Disk Controller Maintenance Manual and the FA723-A/B Disk Controller Maintenance Manual.

SUBSYSTEM PROCESSOR/SYSTEM COUPLER

Normal input channels 00 through 03 and output channels 00 through 07 are dedicated to the system coupler. Coupler programming is described in section 9.

SUBSYSTEM PROCESSOR/CONTROL LOGIC

The normal channel interface lines to the control logic are dedicated to specific purposes. The channel assignments and code descriptions are discussed in the remainder of this section.

FUNCTION CODES (NORMAL OUTPUT CHANNEL 08)

Function codes from the subsystem processor to the control logic are carried on normal output channel 08. Only bits 10 through 15 are active. All other bits must be set to zero. The hexadecimal function codes and their descriptions are as follows:

Hexadecimal Code	Description
0018	Load director buffer - This function causes the control logic to start loading its director buffer from memory.
0020	Master Clear - This function code generates a 15 microsecond delay to allow completion of any memory references. It then issues a 2 microsecond pulse which clears or voids all control logic registers and drops all tag and select lines to the disk drive.

Hexadecimal Code	Description
0022	Execute directors - This function code causes the control logic to begin executing the directors in its director buffer.
0024	Stop execution - This function code causes the control logic to stop executing directors. Directors in process will complete and then turn off the write and erase heads in the disk drive. It does not master clear the control logic.
0025	Clear parity error - This function code clears the parity error flip-flop and drops the composite status bit if no other error conditions exist. The 0018 and 0022 function codes are required to continue operation.
0028	Enter clock step mode - This function code inhibits the internal clock and puts it under program control.
0029	Exit clock step mode - This function code disables the clock step mode and puts the clock in normal internal mode.
002A	Step clock - This function code is used to step the clock during the clock step mode.
0038	Stop loading directors - This function code causes the control logic to stop loading the director buffer. An 0018 code is required to resume director loading.

SPECIAL FUNCTION CODES (NORMAL OUTPUT CHANNEL 09)

Special function codes from the subsystem controller to the control logic are carried on normal output channel 09. The hexadecimal function codes and their descriptions are as follows:

Hexadecimal Code	Description
0080	Clear internal status - This code clears the following status flags: checkword error, compare done, correctable

Hexadecimal Code	Description
	error, uncorrectable error, RAP abort, lost data, sector length error, and compare condition not met.
0800	This code generates simulated read data transitions during a simulated read operation. It is used in conjunction with clock step mode only.

STATUS SELECT CODES (NORMAL OUTPUT CHANNEL 09)

Status select codes from the subsystem controller to the control logic are carried on normal output channel 09. The status remains selected until a new select code is output. All status indications are dynamic. Only normal operating status word 1 should be used during normal operation. The other status indications are used only following fault conditions or for maintenance purposes (clock step mode). Status from the control logic to the subsystem controller is input on normal input channel 04. Status bit assignments and descriptions are contained in section 4. The hexadecimal status select codes and their descriptions are as follows:

Hexadecimal Code	Description
0000	Select normal operating status word 1
1000	Select register file bus status
2000	Not used
3000	Select word counter status
4000	Select normal operating status word 2
5000	Select data buffer output status
6000	Not used
7000	Select bits 16 through 31 of CRC register
8000	Select bits 00 through 15 of CRC register
9000	Select disk status
A000	Select director register bits 00 through 07 and 24 through 31

Hexadecimal Code	Description
B0000	Select byte counter status
C0000	Select bit counter/pointer status
D000	Select address register status
E000	Not used
F000	Select director bits 08 through 23

SPECIAL CONSIDERATIONS (NORMAL OUTPUT CHANNEL 09)

Although the functions on normal output channel 09 are considered as 16-bit hexadecimal codes, they are actually bit oriented. The status select codes use only bits 00 through 03, the test data code uses only bit 04, and the clear internal status code uses only bit 08. This causes an interaction between these functions. For example, when the test data code (0800) or the clear internal status code (0080) are used, the upper 4 bits are all zero and normal operating status one is automatically selected. If a specific status is desired during test data operations, the function codes must be combined. For example, code 5800 will select the data buffer output status during test data operations.

SECTION 4

STATUS REPORTING

STATUS REPORTING

INTRODUCTION

All status from the control logic to the subsystem processor is input on normal input channel 04 after being selected on normal output channel 09 (section 3). Channel 09 can select up to 15 status words. Each of these status words is described below. The description includes a discussion of individual bit assignments, where applicable, and the select code which is output on channel 09 to select the particular status word.

NORMAL OPERATING STATUS WORD 1 (SELECT CODE 0000)

Bit assignments for this status word are as follows:

Bit	Status Indication When Set
00	Read in progress - This bit indicates that the control logic is executing a read or a compare type director. A normal director sequence will begin with a RAP (read address pattern) director which will be followed by additional RAP directors, the read or compare director, and a read checkword director. The read in progress bit will be set from the beginning of the first RAP director to the completion of a read checkword director unless the sequence is abnormally terminated by lost data, RAP abort, or sector length error. These conditions will clear the read in progress bit.
01	Write in progress - This bit indicates that the control logic is executing a write type director. The write in progress bit is set by the first WAP director in a write sequence, and is cleared by the first nonwrite data director unless abnormally terminated by sector length error or lost data, either of which will clear the write in progress bit.

Bit	Status Indication When Set
02	Checkword error - This bit indicates that the immediately preceding data transfer was unsuccessful. The bit is updated following each read checkword operation.
03	Composite status - This bit indicates that one or more of the following conditions has occurred. • Checkword error (see bit 02) • Parity error in control logic (see bit 04) • RAP abort (see bit 05) • Lost data (see bit 06) • Sector length error (see bit 07) • Compare done and condition not met (see bits 11 and 12)
04	Parity error in control logic - This bit indicates that the control logic has detected a parity error. Director loading and execution will stop, but a director in process will continue to completion. The subsystem processor will continue to cycle.
05	RAP abort - This bit indicates that the control logic was not able to successfully complete a read address pattern operation and has stopped operation. Two conditions may cause this error. • The correct address pattern was not found following a RAP specify. • The correct address pattern was not found within two sector marks.
06	Lost data - This bit indicates that data was lost in an operation using the data buffer, and that the control logic has stopped operation. Directors in process will not complete. One of two conditions has occurred. • During a write operation, data was required for the disk, but the data buffer was empty.

Bit	Status Indication When Set
	• During a read operation, data was available from the disk, but the data buffer was full.
07	Sector length error - This bit indicates that the byte count has exceeded the physical sector size and the control logic has stopped operation. Directors in process will not complete.
08	Execute - This bit indicates the control logic has been enabled to execute directors. It will clear if control logic operation is terminated by a parity error, RAP abort, lost data, or sector length error and by a terminate director or a stop execution function code.
09	Not used - (Applies only to controllers without 10333-1 Double Density Option.) Double density 844 unit - This bit indicates that a double density disk unit is selected. (Applies only to controllers with 10333-1 Double Density Option.)
10	Not used
11	Compare done - This bit indicates that the compare decision has been made and that the status indication at bit 12 is valid.
12	Compare condition not met - This bit indicates that the compare condition was unsuccessful (condition not met). This bit is not valid unless bit 11 is set. NOTE Bits 11 and 12 are both cleared at beginning of the next compare operation.
13	Correctable error - This bit indicates that a previously detected disk recording error is correctable. Status select code 8000 will input the correction vector. Status select code B000 will input the information needed to compute the correction vector placement in the data buffer.

Bit	Status Indication When Set
14	Uncorrectable error - This bit indicates that a previously detected disk recording error is uncorrectable.

NOTE

Bits 13 and 14 are cleared at the beginning of the next Initiate error correction director.

15	Not used

REGISTER FILE STATUS (SELECT CODE 1000)

Bits 00 through 15 of this status word contain the contents of the register file bus.

WORD COUNTER STATUS (SELECT CODE 3000)

Bits 00 through 15 of this status word contain the contents of the word counter.

NORMAL OPERATING STATUS WORD 2 (SELECT CODE 4000)

Bit assignments for this status word are:

Bit	Status Indication When Set
00	Director buffer empty
01	Data buffer empty
02	Read decode
03	Function decode
04	Write decode
05	Compare decode
06	Disk status enable decode
07	Disk function decode
08	Disk select decode
09	Load decode
10	Data output from writer (data to disk)

Bit	Status Indication When Set
11	Not used
12	Serial input data to checkword generator
13	Not used
14	Not used
15	~~Not used~~ HALT [illegible]

DATA BUFFER OUTPUT STATUS (SELECT CODE 5000)

Bits 00 through 15 of this status word contain the contents of the data buffer output lines.

CHECKWORD STATUS (SELECT CODES 7000 and 8000)

These select codes input the lower (16 through 31) and upper (00 through 15) bits of the 32-bit checkword generator respectively. (See Figure 8-1.)

DISK STATUS (SELECT CODE 9000)

This inputs the status of the disk drive. Individual bit assignments may be found by referencing the enable disk status director in section 5.

SELECT DIRECTOR REGISTER (SELECT CODE A000)

This status word contains bits 00 through 07 and 24 through 31 of the director register in bit positions 00 through 15 respectively.

BYTE COUNTER STATUS (SELECT CODE B000)

Bits 00 through 15 contain the contents of the byte counter.

If bit 13 (correctable error) of normal operating status word 1 is set, the byte counter contains the remainder of the shift count supplied by the initiate error correction director (section 5).

BIT COUNTER AND POINTER STATUS (SELECT CODE C000)

Bits 00 through 03 contain the position of the director buffer input pointer. Bits 04 through 07 contain the position of the director buffer output pointer. Bits 11 through 13 contain the contents of the bit counter. Bits 08 through 10, 14, and 15 are not used.

ADDRESS REGISTER STATUS (SELECT CODE D000)

Bits 00 through 15 contain the current memory address.

SELECT DIRECTOR REGISTER (SELECT CODE F000)

This status word contains bits 08 through 23 of the director register.

SECTION 5

CONTROL LOGIC INSTRUCTIONS

CONTROL LOGIC INSTRUCTIONS

INTRODUCTION

The control logic is a special purpose processor which executes a unique set of instructions called directors. A director sequence is selected by the subsystem processor depending on the operation required. All directors are 32 bits in length except for two 16-bit directors and one 64-bit director. The directors are divided into two classes: data directors and support directors. The formats and functions of each class of directors are discussed separately in the following paragraphs. Information on director sequencing is provided in section 6.

DATA DIRECTORS

These directors are executed in real-time with the rotation of the disks. They control the disk format and the transfer of data to and from the disk. All data directors are 32 bits in length.

DATA DIRECTOR FORMAT

The general format used for data directors is shown in Figure 5-1. Since all directors are made up of 16-bit syllables, each syllable is shown separately. A data director is made up of the top syllable and one of the three bottom syllables. The overlapping areas indicate alternate field identification for certain bits and groups of bits. The use of each field and its individual bits is described following the format. All unused bits in a director must be set to zero.

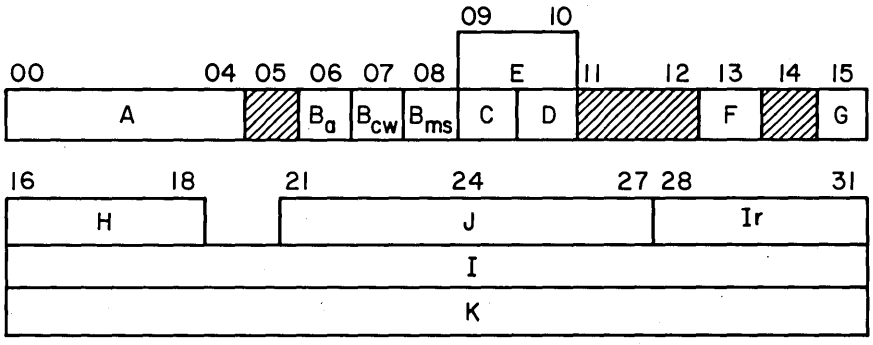


Figure 5-1. Data Director General Format

Field	Description
A	Function Code This is a 5-bit field which identifies the function to be performed. Bit 00 must always be zero for a data director. Bits 01 and 02 identify the basic function, that is, read, write, or compare. Bits 03 and 04 identify subfunctions of the basic function, that is, normal, address pattern, checkword, or delay.
B_a	Compare and Save When bit 06 is set in a compare director, information read from the disk will be saved in the register file beginning with register location zero. The I/O length of the compare operation can not exceed 32 8-bit bytes or 6-bit characters. Fields D and G_r cannot be set when B_a is set.
B_{cw}	Generate Checkword Bit 07 set requires the generation of a 32-bit checkword. If clear, checkword generation will be inhibited.
B_{ms}	Mark Start This is bit 08 and is used only with a delay director. When set, it indicates that execution of the director will coincide with the leading edge of the pulse defined in field E.
C	Byte Handling (within DDC) This field is bit 09 and controls byte handling. Since the hardware is 16-bit oriented, this bit allows beginning of an operation at either the odd or even byte. When this bit is clear, the first data byte is taken from the upper byte position (bits 00 through 07) of the 16-bit source register or placed in the upper byte position of the destination register. When this bit is set, the first data byte is taken from the lower byte position (bits 08 through 15) of the source register or placed in the lower byte position of the destination register.

Field	Description
D	Select Register File This field is bit 10 which selects the register file for use during a data handling operation. When this bit is set, data is transferred to or from the register file beginning with register location zero. When this bit is used, the I/O length of the data operation must be specified in field I of the director and cannot exceed 32 8-bit bytes or 6-bit characters. If this bit is clear, data is transferred to or from the data port selected by a preceding address director.
E	Mark Start Condition This field is bits 09 and 10 which define the start condition required by the B_{ms} field. The bits are coded to designate the condition which will cause the director to begin execution. The codes and their conditions are:

Bit 09	Bit 10	Condition
0	0	Index mark
0	1	Sector mark
1	0	Sector alert
1	1	Not used

F	Compare Condition Bit 13 set causes a compare for data equal, indicating the data supplied from the data port is equal to the data read from the disk. During the compare operation, data supplied from the data port is compared bit by bit with the data being read from the disk. The compare decision is made when the first nonequal bit is encountered, or when the byte counter has decremented to zero. Bits 11 and 12 (compare done and compare condition not met, respectively) of normal operating status word 1 provide the results of the compare operation.

Field	Description
	NOTE Data lengths must be equal for compare operation.
G	This is bit 15 of the director - It is interpreted differently depending upon the type of director being executed. The bit notations for the various usages and descriptions of the usages are given below.
G_h	Select head advance - If this bit is set in a read checkword or write checkword director, it will cause a head advance pulse to be sent to the selected drive.
G_p	Select pattern byte - This bit allows selection of one 8-bit byte from the register file location specified in bits 28 through 31. If this bit is clear, bits 00 through 07 of the register file location will be selected. If this bit is set, bits 08 through 15 will be selected. This bit is used in the RAP and WAP directors. In 6-bit mode, only bits 02 through 07 or 10 through 15 are used.
G_r	Register file pointer - If this bit is set, the I/O length for the data operation must be obtained from the register file location specified by field I_r (bits 28 through 31).
H	Function Condition 2 This field is bits 16 through 18 of the first WAP director that control operating parameters during data operations. The field entries are described as follows: H1 through H3 control the write and erase heads during write operations. Bit assignments are: H1 - Bit 16 set indicates that head control operations are required. H2 - Bit 17 set turns on the write head. When clear, it turns off the write head.

Field	Description
	H3 - Bit 18 set turns on the erase head. When clear, it turns off the erase head.
	Hg - Bit 16 of the first read address pattern director in a read operation. When set, this bit turns on the read gate in the selected drive.
I	I/O Length
	This field is bits 16 through 31 that specify the number of byte/characters of data to be transferred. This field is not used if the data length is referenced to a register file location.
I_r	Register File Location
	This field is bits 28 through 31. If bit G_r (bit 15) is set, this field contains the register file location which contains the I/O length.

NOTE

If field D (bit 10) specifies a data transfer to or from the register file, the I/O length cannot be referenced to the register file.

J	Function Condition 3
	This field is bits 21 through 27 that control hardware operations during a read address pattern director. These bits provide the detailed compare instructions which allow a sequence of RAP directors to establish bit and byte synchronization with the serial data stream being read from the disk. The individual field entries and their use are described as follows.

Field	Description
	J1 - Bit 21 determines at what point in the serial bit stream a compare will be made. When set, a compare is attempted at every bit position (RAP anyplace) without reference to the bit counter. When clear, the compare is attempted only when the bit counter is at position 5 (6-bit mode) or position 7 (8-bit mode). This is a RAP specific. J2 - Bit 22 set causes the bit counter to be set to the bit position contained in field Jf. J3 - When bit 23 is set, the compare condition will be satisfied by any pattern except the specified pattern. (This allows a search for NOT pattern). J4 - When bit 24 is set, a full byte time will expire before a RAP anyplace (bit 21 set) is attempted. Jf - This field is bits 25 through 27 that contain the count to which the bit counter will be set if J2 (bit 22) is set.
K	Delay Director Byte Count This field is bits 16 through 31 that contain a count of one byte less than the bytes of delay required. NOTE If field B_{ms} (mark start) is set, the byte count delay will not begin until the mark start condition has been satisfied.

DATA DIRECTOR DESCRIPTIONS

The following paragraphs describe the use and format of each of the data directors. The format indicates the fields applicable to each director. The fields themselves are described above. Each director description lists the purpose of the director and any pertinent characteristics (programming considerations) which will aid the programmer in using the directors most effectively. The director descriptions are listed in the order of their function codes and are identified by their function names. Table 5-1 lists the function code and name of each of the data directors. General considerations concerning the use of data directors immediately follows the data director descriptions.

TABLE 5-1. DATA DIRECTORS

Function Code (Binary)	Director Name
00100	Read normal
00101	Read skip
00110	Read checkword
00111	Read address pattern
01000	Write normal
01001	Write Nx
01010	Write checkword
01011	Write address pattern
01100	Compare normal
01111	Delay

NOTE

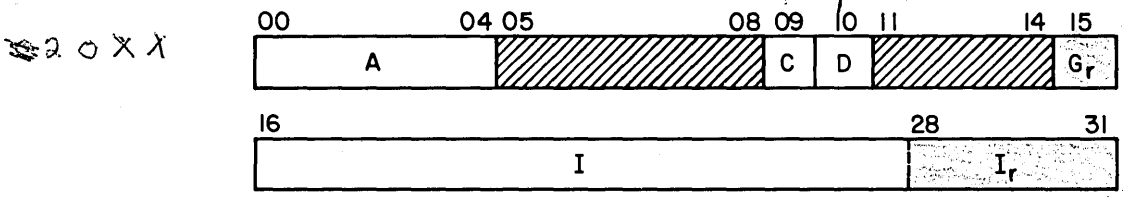
Loading requires 3.6 microseconds (maximum). All data directors require one byte/character time to decode and execute. Based on a bit time of 147 nanoseconds, in 6-bit mode this will be about 900 nanoseconds and in 8-bit mode, about 1.2 microseconds. Variations from these times will be included in the programming considerations for the individual directors.

257

READ NORMAL

Function Code - 00100

0 = EVEN NUMBER OF BYTES
1 = LOAD DATA TO R.F.

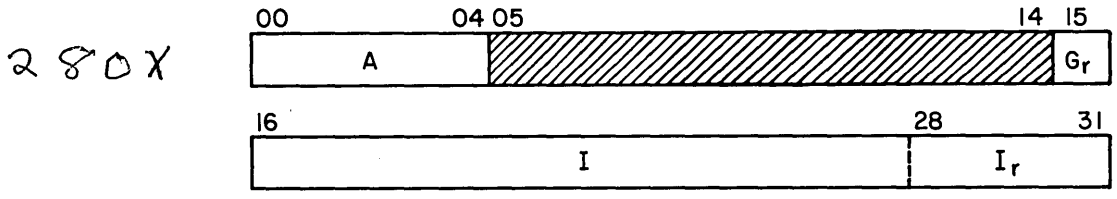


Purpose - This director transfers data from the disk to the destination designated by the associated address director. The I/O length is determined by field I (bits 16 through 31) or is obtained from the register file location designated by field I_r.

READ SKIP

270

Function Code - 00101



Purpose - This director is identical to the read normal director except that no data is transferred. Even though no data is transferred during execution of this director, a checkword is generated. The I/O length is determined by field I (bits 16 through 31) or is obtained from the register file location specified by field Ir.

Programming Considerations

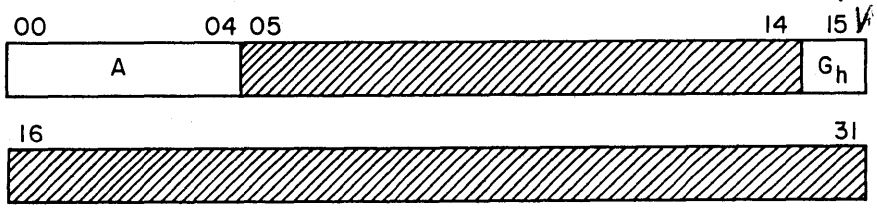
1. This checkword can be used to verify a sector of data. The read skip director will cause generation of the checkword without transferring any data to or from the HLP. A read checkword director can then be used to verify the data. This director can be used in place of a compare director for rapid verification without requiring the HLP to retransmit the data.

READ CHECKWORD 271

Function Code - 00110

1=ADVANCE HEADS

300X



Purpose - This director causes the hardware to treat the next 32 bits from the disk as a checkword and to compare it with the checkword generated by the hardware. The results of the compare are saved in a flip-flop for use by subsequent directors. The contents of the checkword generator will be retained until execution of the next RAP director. If bit 15 is set, the controller will generate a head advance pulse to the selected drive, providing the function tag specifying control select has been enabled via an enable disk function director. A 0.5-microsecond head advance pulse is sent to the drive after the completion of the read checkword director and causes the head counter to be incremented by one. This allows rapid switching of heads when reading from one track to the next.

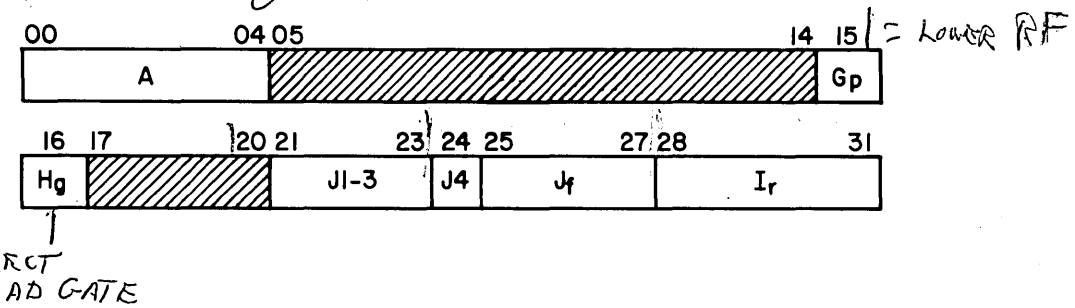
RAP	J1	J2	J3	J4	Hg	
RAP 1	1	0	0	1	1	Check for "0" = pattern
RAP 2	1	1	1	0	0	Look for sync bit
RAP 3	0	0	0	0	0	Read sync

READ ADDRESS PATTERN (RAP)

Function Code - 00111

275

380X



Purpose - This director is used to establish bit and byte synchronization with serial information received from the disk drive. It is actually a compare operation which compares data from the disk with address patterns previously loaded into the register file. If Gp is clear, the pattern is taken from bits 00 through 07 of the specified register file. If Gp is set, the pattern is taken from bits 08 through 15. Through proper selection of J field entries, a list of RAP directors can be developed which will establish synchronization. Additional information regarding the use of the RAP director is provided in section 6. Bit 24 must be set in the first RAP to ensure that a full byte time expires before a compare is attempted.

Programming Considerations

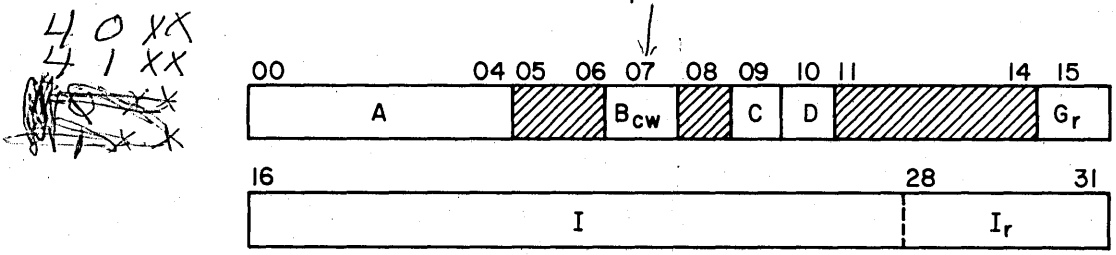
1. The last RAP director in a RAP sequence must be a RAP specific. If the proper pattern is not found, the control logic will cease operation and will set the RAP abort and composite status bits.
2. The register file must be loaded with all required patterns prior to the decoding of any RAP directors.
3. A RAP sequence will be abnormally terminated if one of the following occurs.
   a. The correct pattern is not found during execution of a RAP specific director.
   b. A sector mark occurs during a RAP sequence.

   Either of these conditions will clear the read in progress bit, set the RAP abort status bit, and set the composite status bit.

4. The first RAP director in a sequence must have field Hg set (bit 16) to turn on the read gate. The read gate will remain on until a nonread data director is decoded. This will normally be the director immediately following the read checkword director.

5. With bit 24 set, the first RAP in a sequence (RAP anyplace) will not attempt to make a valid compare until one byte time has elapsed. This allows the residue in the shift register to be shifted out and only valid data used for the compare.

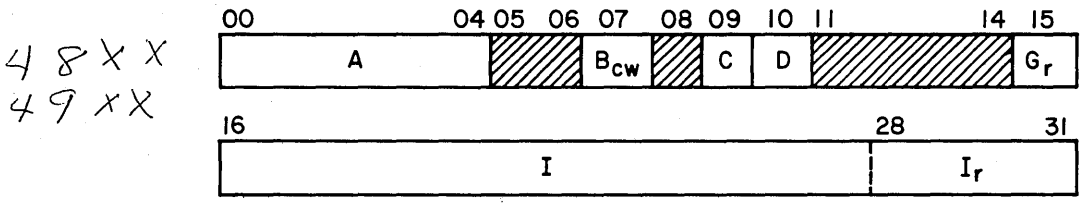
WRITE NORMAL
Function Code - 01000



Purpose - This director transfers data to the disk from the data port designated by the associated address director. The I/O length is determined by field I (bits 16 through 31) or is obtained from the register file designated by field I_r.

WRITE NX
Function Code - 01001

Format



Purpose - This director is identical to a write director except that it causes one 6-bit character or 8-bit byte to be written consecutively on the disk surface. The length of the write operation is obtained from the I (bits 16 through 31) field or from the index register designated by the I_r field.

Programming Considerations

1. This director may be used to write long fields of redundant information, such as an area of zeros preceding a field for synchronization purposes.
2. If bit 10 (field D) is set, the write data will be taken from the byte position, specified by field C, in register file location zero. All references will be made to this location until the I/O length is satisfied.

NOTE

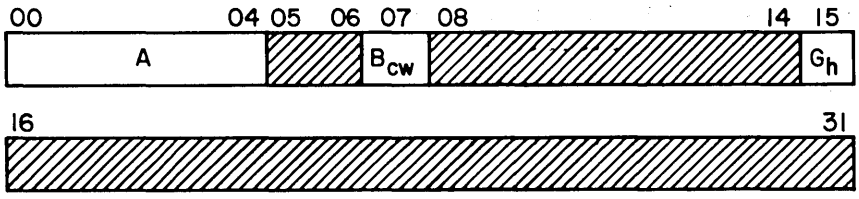
If a write Nx from register file is specified (field D set), the I/O length cannot be referenced to the register file.

WRITE CHECKWORD

Function - 01010

Format 294

510X



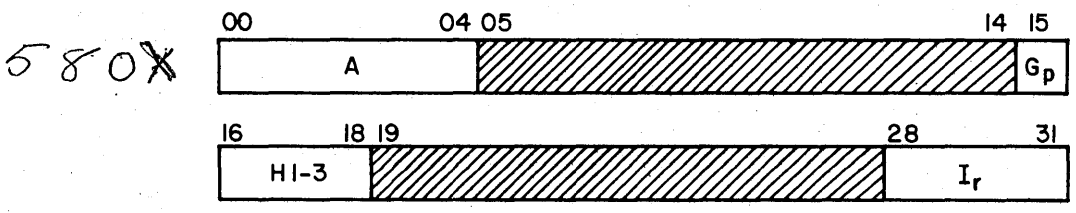
Purpose - This director will cause the contents of the 32-bit redundancy field generator to be transferred to the disk. This will be the checkword generated during the immediately preceding write operation. Bit 15 (G_h) provides an optional head advance pulse. Bit 07 must always be set.

Programming Considerations

1. All write operations must end with a write checkword director.
2. If another write type director does not follow the write checkword director, the following actions will occur.
   a. Drop erase gate*
   b. Write the checkword
   c. Write pad byte
   d. Drop write head
   e. Drop write in progress bit
   f. Advance head (only if selected)
3. If another write type director follows the write checkword director, the checkword is written, and the next write type director is executed.

WRITE ADDRESS PATTERN
Function Code - 01011

Format



Purpose - This director allows writing special synchronization patterns on the disk.

*Applies only to controllers without 10333-1 Double Density Option.

Programming Considerations

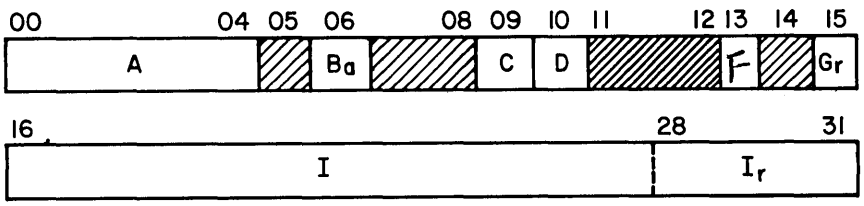
1. The required pattern is taken from bits 00 through 07 of the specified register file location if Gp is clear. Bits 08 through 15 are used if Gp is set.
2. The first WAP director in a write sequence will delay five bit times before executing the data portion of the director. This allows byte synchronization to occur. Head switching will occur at the beginning of the execution time without any delay.
3. Since this director only requires one byte time to execute, do not attempt to execute a support director during a WAP sequence.

COMPARE NORMAL
Function Code - 01100

303

Format

60XX
62XX



Purpose - This director is similar to a combination read/write operation. Data is read from the disk and compared with data being supplied from the system coupler, core memory, or register file. Bits 11 and 12 of normal operating status word 1 (compare done and compare condition not met) provide the result of the compare operation. The length of the operation is obtained from field I (bits 16 through 31) or from the index register specified by field Ir.

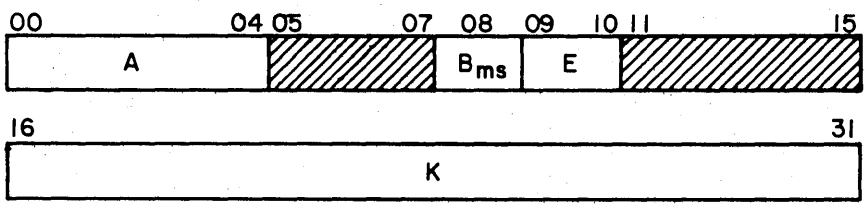
Programming Considerations

1. Each time a compare is entered, the compare condition flip-flops are cleared.
2. Bits 06, 10, and 15 provide three separate uses of the register file during this director. Only one of these bits may be set at a time. If more than one is set, the operation is undefined.

DELAY
Function Code - 01111

311

78X0



Purpose - This director causes a delay which is one byte greater than the byte count (8-bit mode) or character count (6-bit mode) contained in the K field. The following data director must be made available during the delay period so that it can begin execution when the delay period has expired. Support directors may be executed during the delay period.

Programming Considerations

1. If a mark start is specified (field B), the byte count delay will not start until the mark start condition is satisfied.
2. The unit must be selected before a mark start delay is executed to ensure starting on the leading edge of the disk condition specified by field E.

General Considerations for Data Directors

1. All data directors except read skip, write from register file, or read to register file preceded by an associated address director to establish data port and the starting address.
2. A data director which uses the register file as a data port cannot use the register file for referencing the I/O length.
3. All read sequences and compare sequences must be preceded by at least three read address pattern directors to establish byte synchronization.
4. All write and write Nx sequences must be preceded by at least one write address pattern director.

5. The desired disk drive must be selected and all necessary functions initiated through the use of support directors before any data directors involving that drive can be executed. The control select tag line must remain enabled during execution of the data directors so that the heads can be controlled by the data directors.

6. Two separate and distinct clocks are used during data handling operations. An internal write clock is used for write operations (write, write Nx, write checkword, and write address pattern). All compare and read type operations use a read clock generated by the disk drive. The delay director uses the internal clock.*

   Three separate and distinct clocks are used during data handling operations. A write clock generated by the disk drive is used for write operations (write, write Nx, write checkword, and write address pattern). All compare and read type operations use a read clock generated by the disk drive. The delay director uses an internal clock. **

7. When operating in 6-bit mode, the two most significant bits of each 8-bit byte are discarded. The 6-bit bytes are comprised of bits 02 through 07 and 10 through 15. Bits 00, 01, 08, and 09 are set to zero.

8. If data length expires with an odd byte (bits 00 through 07), that data buffer rank will be considered used.

9. The read gate is activated by setting bit 16 of the first RAP director in a read sequence. It will drop automatically when a nonread data director is decoded.

10. The write and erase gates are both turned on by a WAP director. The erase gate is turned off by a write checkword director. The write gate is turned off when the WIP bit drops (bit 01 of normal operating status word 1). A conditional branch director provides a special means of turning on the erase gate.*

    The write gate is turned on by a WAP director and is turned off when the WIP bit drops (bit 01 of normal operating status word 1). **

11. Ensure that another director is ready for execution at the time the previous director has completed execution.

*Applies only to controllers without 10333-1 Double Density Option.
**Applies only to controllers with 10333-1 Double Density Option.

SUPPORT DIRECTORS

Support directors are used to establish hardware conditions which must exist before the data directors can be executed. They are not dependent upon disk rotation and can, therefore, be executed at any time providing they do not interfere with any data handling which might be in progress.

SUPPORT DIRECTOR FORMAT

Support directors vary in length from 16 to 64 bits. The function code is always contained in bits 00 through 07. The remainder of the director is basically bit-oriented and the format is shown for each individual director.

NOTE

Unused bits in director fields (or related register file entries) must always be set to zero.

SUPPORT DIRECTOR DESCRIPTIONS

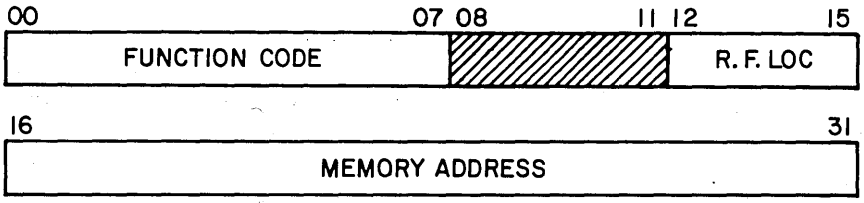
The following paragraphs describe the use and format of each of the support directors. The discussion includes descriptions of the various director fields, the purpose of the director, director execution time, and any pertinent characteristics which will aid the programmer in utilizing the directors effectively. Table 5-2 lists the function code and name of each of the support directors. General considerations concerning the use of support directors immediately follow the support director descriptions.

TABLE 5-2. SUPPORT DIRECTORS

Function Code (Hex)	Director Name	Execution Time (μsec)*
00	Load register file from memory	$1.234^{+1.553}_{-0.073}$
02	Address	1.018 min to 1.482 max.
04	Conditional branch	$2.1^{+2.073}_{-0.073}$
08	Terminate	
10	Subtract	1.250 ± 0.073
12	Add	1.250 ± 0.073
14	Test (with decrement)	1.250 ± 0.073
16	Complement	1.250 ± 0.073
80	Load register file	0.514 ± 0.073
82	Key point	0.220 ± 0.073
84	Load polynomial	0.220 ± 0.073
86	Load hardware conditions	0.220 ± 0.073
88	Copy disk status	1.249 ± 0.073
92	Stop loading directors	
94	Unconditional branch	
98	Store register file	$1.234^{+1.553}_{-0.073}$
C0	Enable disk functions	3.6 ± 0.073
C8	Disable disk functions	3.6 ± 0.073
E0	Enable/disable disk status	0.220 ± 0.073
F0	Initiate error correction	0.147 per bit (max)

*See individual director descriptions for variations to the times listed.

LOAD REGISTER FILE FROM MEMORY
Function Code - 00_{16}

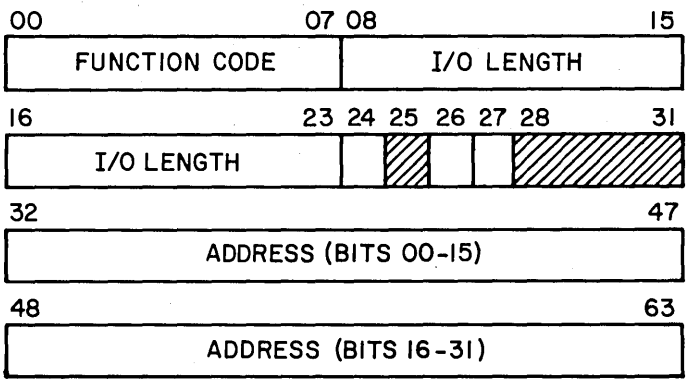


Purpose - This director transfers the contents of a memory location specified by bits 16 through 31 into the register file location designated by bits 12 through 15.

Programming Considerations

1. This director must not be executed during a read, write, compare, RAP, or WAP, unless the address director I/O lengths (word count) are satisified.
2. The minimum execution time of this director is 1.161 microseconds. This time could increase to 2.787 microseconds depending on the length of time required to complete the memory reference.

ADDRESS
Function Code - 02_{16}



Purpose - This is a 64-bit director which establishes the path for a data transfer operation. It also determines the direction of data flow, the data port to be used, and whether or not buffering will be required. The field entries for the director are:

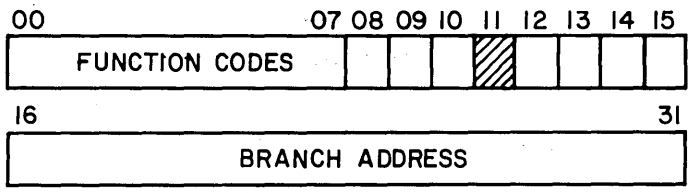
- Bits 00 through 07 contain the function code.
- Bits 08 through 23 contain the length of the data transfer in 16-bit words.
- Bits 24 specifies the data port. A 0 designates the system coupler; a 1 designates core memory.
- Bit 26 specifies data flow direction. A zero designates data flow from the data buffer to the shift register (write or compare operations) and a 1 designates data flow from the shift register to the data buffer (read operations).
- Bit 27 - When set, this bit specifies that no end of record pulse will be sent or is expected. When clear, this bit specifies an EOR must be sent or received after the data transfer. This bit applies to coupler transfers only and has no affect on transfers to or from memory.
- Bits 32 through 63 contain the starting address of the data operation.

Programming Considerations

1. Use this director to provide address references for read, write, write Nx, and compare directors.
2. A second address director can be issued before the first has completed its execution. If the data port to be addressed cannot accept address information, execution of this director will wait until the data port can be accessed.
3. The execution time for this director assumes that the entire director (64 bits) is in the director buffer when execution begins. If execution of the director begins before the entire director is loaded, additional time must be added to allow for the additional memory references required.

CONDITIONAL BRANCH

Function Code - 04_{16}



Purpose - This director will cause the hardware to branch to the specified memory address if the flag bit indicated by the specified condition is set. The field entries for the director are:

- Bits 00 through 07 contain the function code.
- Bit 08 causes a branch if the "condition equal" flag is set.
- Bit 09 causes a branch if the "condition greater than" flag is set.
- Bit 10 causes a branch if the "condition less than" flag is set.
- Bit 12 causes a branch if the compare condition is not met.
- Bit 13 causes a branch if checkword does not equal zero.
- Bit 14 causes a branch if the bit compare flag is set.
- Bit 15 causes execution of the conditional branch director to wait until the read in progress status bit drops.
- Bits 16 through 31 contain the branch address.

Programming Considerations

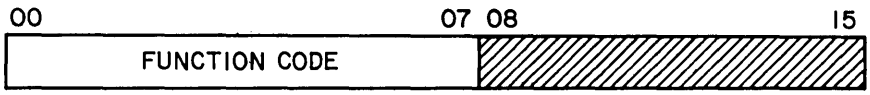
1. The flags which are checked by the conditional branch director must be conditioned by a preceding test director, compare director, or read checkword director. More than one flag can be checked by one director.
2. If the conditional branch is made, the branch address is transferred to the address register and the director buffer is cleared (input and output pointers equal and director buffer empty). The director buffer will then start loading from the new address location.

3. If bit 15 of the conditional branch is set, the conditional branch will be decoded and the execution stopped until the read in progress status bit drops indicating the end of a read operation. At that time, the conditional branch director will resume its execution. If the specified branch condition is not made,

   If the specified branch condition is not made, signals will be sent over the disk interface port to enable the erase head of the drive. The drive must have had the control select tag sent to it by a previous operation. It will be necessary to complete the execution of a write address pattern data director within 18 microseconds to turn on the write head to prevent an 844 drive from detecting a fault condition. *

4. The execution time for this director is 2.1 ± 0.073 microseconds. If bit 15 is used to delay execution of this director, the execution time could increase to 6.8 ± 0.073 microseconds to allow completion of the read checkword director.

TERMINATE
Function Code - 08_{16}



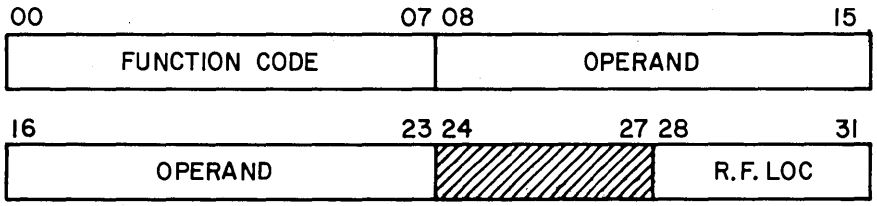
Purpose - This director causes the hardware to stop executing directors as soon as it is executed. This is a 16-bit director.

Program Considerations

1. It stops director execution as soon as it is executed. Director loading will continue.
2. The subsystem processor must issue an execute directors function code (0022 on normal output channel 08) to restart execution. Execution will then begin at the director buffer location existing at the time of the terminate.
3. Data directors which are in process will continue until both the address director and data director data lengths (word count and byte count) are satisfied.

*Applies only to controllers without 10333-1 Double Density Option.

SUBTRACT
Function Code - 10_{16}

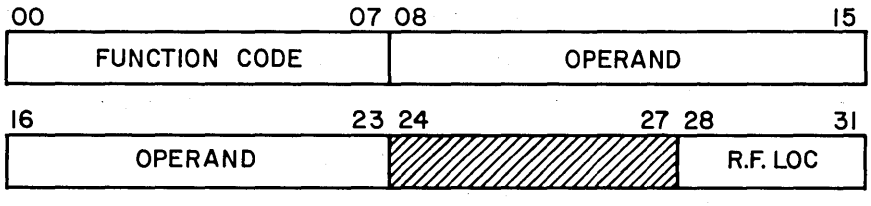


Purpose - This director subtracts the operand in bits 08 through 23 from the contents of the register file location specified by bits 28 through 31. The difference is then placed back in the specified register file location.

Programming Considerations

1. All quantities used in the subtract operation must be positive.
2. All subtract operations are performed in two's complement mode.
3. The hardware has no provision for sensing overflow.

ADD
Function Code - 12_{16}



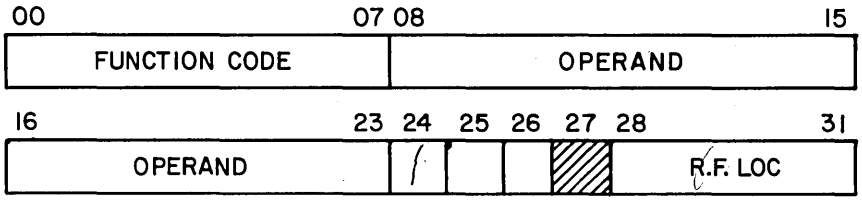
Purpose - This director adds the operand in bits 08 through 23 to the contents of the register file location specified by bits 28 through 31. The sum is then placed back in the specified register file location.

Programming Considerations

1. All quantities used in the add operation must be positive.
2. ~~All add operations are performed in two's complement mode.~~
3. The hardware has no provisions for sensing overflow.

TEST

Function Code - 14_{16}



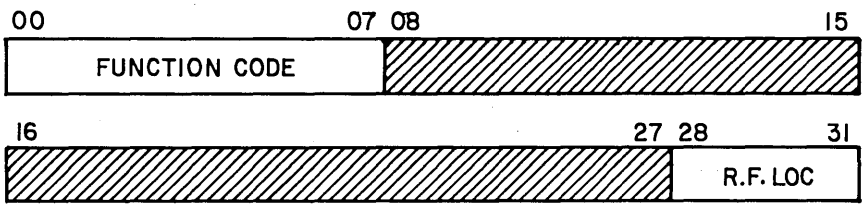
Purpose - This director subtracts the operand in bits 08 through 23 from the contents of the register file location specified by bits 28 through 31. Bits 24 through 26 establish the compare conditions as follows (only one bit may be set at a time).

- Bit 24 - When set, this bit causes a compare for relative magnitude, that is, equal, register file greater than operand, or register file less than operand. The compare conditions set flags for subsequent use by a conditional branch director.
- Bit 25 - When this bit is set, it causes a bit compare in which a flag is set if there are any 1 bits in the register file corresponding to the 1 bits in the operand field. Other bits in the register file are disregarded.
- Bit 26 - When this bit is set, it causes a test and decrement. The contents of the register file are tested in the same manner as for bit 24. The respective flag (equal, greater than, or less than) is set following the test. If the quantities were not equal, the contents of the register file are decremented by one.

Programming Considerations

1. This director pre-clears all test flags before making any tests. The flags will remain set until the next test director. This allows more than one conditional branch director to follow each test director.

COMPLEMENT
Function Code - 16_{16} 3-240

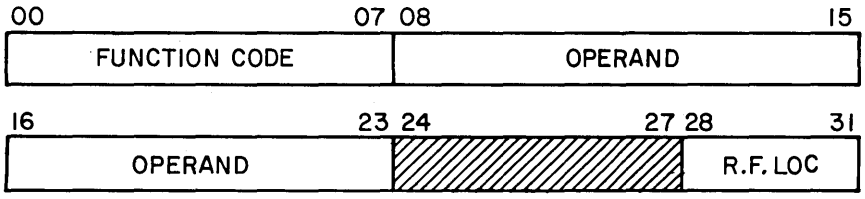


Purpose - This director causes the contents of the register file location specified by bits 28 through 31 to be complemented and then replaced in the register file.

Programming Considerations

1. Complement operations are performed in one's complement mode. (This is a bit by bit complement.)

LOAD REGISTER FILE
Function Code - 80_{16} 3-241

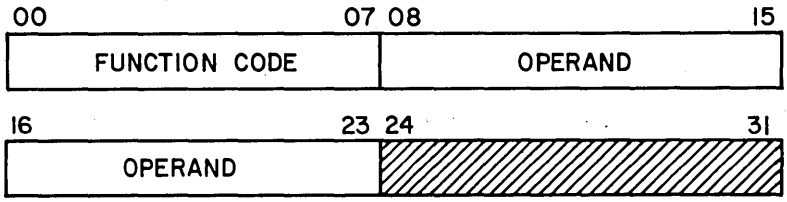


Purpose - This director causes the operand in bits 08 through 23 to be loaded into the register file location specified by bits 28 through 31.

Programming Considerations

1. This director cannot be used if the register file is being used for data handling operation.

KEY-POINT
Function Code - 82_{16}

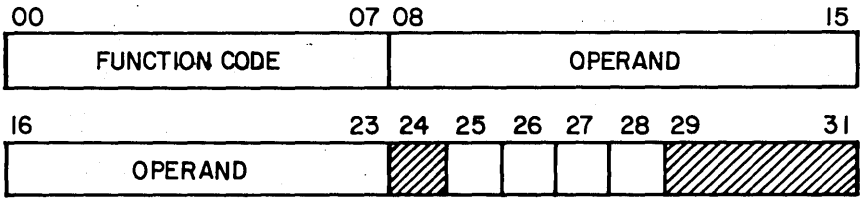


DIRECTOR Purpose - The keypoint instruction is a no-operation instruction that loads the ~~function data~~ SUPPORT catch register with operand bits 08 through 23 from the director. The outputs of this register are available on test points on boards B01 and B02 and in the backpanel for instrumentation purposes.

This instruction provides a negative going pulse at C09 testpoint 32 which is 100 to 140 nanoseconds wide. The operand bits are stable at the testpoints on B01 and B02 on the positive edge of the trigger pulse.

Bit	B01 Testpoint	Bit	B02 Testpoint
08	07	16	07
09	06	17	06
10	05	18	05
11	08	19	08
12	32	20	32
13	31	21	31
14	26	22	26
15	21	23	21

LOAD POLYNOMIAL
Function Code - 84_{16}



Purpose - This director loads the operand in bits 08 through 23 into the proper error control register. Bits 25 through 28 identify the operand and the associated error control generator.

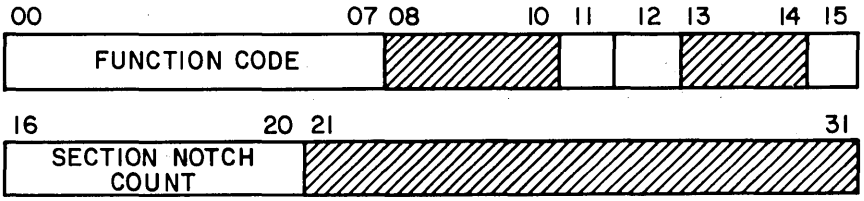
- Bit 25 - Load the divide polynomial associated with bits 00 through 15 of the 32-bit generator.
- Bit 26 - Load the divide polynomial associated with bits 16 through 31 of the 32-bit generator.
- Bit 27 - Load the multiply polynomial associated with bits 00 through 15 of the 32-bit generator.
- Bit 28 - Load the multiply polynomial associated with bits 16 through 31 of the 32-bit generator.

Programming Considerations

1. Because multiply and divide polynomials are 32 bits in length, two directors are required to load each polynomial.
2. Once loaded, the polynomial will remain until a new polynomial is loaded.
3. If the polynomial operands are identical, they can be loaded simultaneously by setting two or more of the select bits.
4. Figure 8-1 illustrates the registers associated with checkword generation.
5. See section 8 for use of this director.

LOAD HARDWARE CONDITIONS

Function Code - 86_{16}



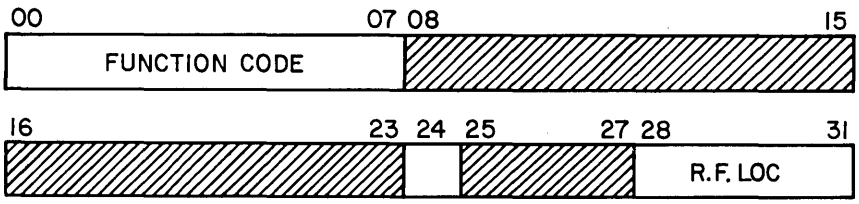
Purpose - This director preconditions the hardware for conditions which will remain relatively static for all hardware operations. The selectable conditions are:

- Bit 11 - When set, this bit presets every stage of the 32-bit checkword generator. When clear, this bit preclears every stage of the 32-bit checkword generator.
- Bit 12 - When set, this bit selects 8-bit mode. When clear, this bit selects 6-bit mode.
- Bit 15 - When set, this bit will cause a zero fill to the end of the sector upon receipt of an end of record (EOR).
- Bits 16 through 20 contain a sector notch count which is one less than the number of physical sector notches required per logical sector. A zero count equals one physical sector notch per logical sector.

COPY DISK STATUS

Function Code - 88_{16}

5-246



Purpose - This director copies status from the selected disk drive and places it in the register file location designated by bits 28 through 31. The status word will always contain the status previously selected by an enable disk status director. If bit 24 is set, it will also contain the unselected status for the selected disk drive.

Programming Considerations

1. This director must be preceded by an enable disk status director which enables the contents of a selected register within the disk drive. If bit 24 is clear (no unselected status), bits 00 through 05 of the status word will contain zeros.
2. If bit 24 is set, bits 00 through 05 of the status word will contain the following unit status.

   Bit 0 - Sector alert
   Bit 1 - Seek error
   Bit 2 - Unit busy
   Bit 3 - Unit selected
   Bit 4 - Unit ready
   Bit 5 - Unit on-line

3. Bit 06 of the status word is unused.
4. Bits 07 through 15 of the status word contain the selected status. See Table 5-3 and the bit descriptions which follow it.
5. If no unit is selected, the status word will be all zeros.
6. The sampling of drive information takes place 1.176 microseconds after the start of execution.

TABLE 5-3. STATUS SELECT LINE FUNCTIONS (UNIT TO CONTROL LOGIC)

Status Bus	Read Cyl. Sel.	Read Diff. Cnt.	Read Head Reg.	Read Sec. Cntr.	Read Sec. Reg.	Read Interlock	Read Position	Read Fault	Read Control
Bit 15	1	1	1	1	1	Pack on	Forward	$\overline{W+E+R\cdot On\ Cyl.}$ ①	Sector
Bit 14	2	2	2	2	2	Sector Block ②	Reverse	(W+E)·R ③	Pack Unsafe
Bit 13	4	4	4	4	4	Heads Loaded	Cyl. Pulse	Current	Seek Error ④
Bit 12	8	8	8	8	8	Brush Cycle	End of Travel	+ Volt	On Cylinder
Bit 11	16	16	16	16	16	Start Switch	Fine Servo	- Volt	Index
Bit 10	32	32				Local Remote	Speed	Seek	Amplitude Monitor 1 ⑤
Bit 09	64	64				Spindle Motor On		AC Write Fault ⑥	End of Cylinder
Bit 08	128	128				Power Supply Temp. ⑦			Amplitude Monitor 2 ⑧
Bit 07	256	256		Cntr Valid		Logic Temp. ⑧			Amplitude Monitor 3 ⑧
Bit 06⑥	512 ⑥	512 ⑥							

① This is W+R· $\overline{On\ Cyl.}$ for double density option.
② This is control interlock for double density option.
③ This is W·R for double density option.
④ This is seek incomplete for double density option.
⑤ This is EOT seek error for double density option.
⑥ Used only with 10333-1 Double Density Option.
⑦ This is logic temperature for double density option.
⑧ Not used with double density option.

Status Select Line Bit Descriptions

1. Read cylinder select (control logic to unit). This signal gates the contents of the unit cylinder register to the control logic on the status bus.

2. Interlock select (control logic to unit). This signal gates unit interlock status to the control logic with bit assignment as follows on the status bus.

	Signal	Function
Bit 15	Pack on	Indicates a pack is mounted on the spindle.
Bit 14	Sector block or	Indicates the unit cover is closed and sector block is in position to sense sector disk.
	Control interlock	Indicates the unit cover is closed, all dc circuit breakers are closed, and the start switch is on. (Applies to double density option only.)
Bit 13	Heads loaded	Indicates heads have been loaded on the disk pack.
Bit 12	Brush cycle	Indicates pack brush cycle is in progress.
Bit 11	Start switch	Indicates start switch is on.
Bit 10	Local/remote	Indicates unit sequencing power is under control of the disk drive (local).
Bit 09	Spindle motor	Power is applied to the spindle motor.
Bit 08	Power supply temp. or	Indicates power supply temperature normal.
	Logic temp.	Indicates logic chassis temperature normal. (Applies to double density option only.)
Bit 07	Logic temp.	Indicates logic chassis temperature normal. (Not used with double density option.)

3. Read head register select. This signal gates the contents of the unit head register to the control logic on the status bus.

4. Read difference counter select. This signal gates the one's complement of the contents of the unit difference counter to the control logic on the status bus.

5. Read sector counter select. This signal gates the contents of the unit sector counter to the control logic on the status bus. Bit 7 is sector counter valid signal. When this line is true, the output of the sector counter is not in the process of changing. The counter contents are one number ahead of the actual sector count.

6. Read sector register select. This signal gates the contents of the unit sector register to the control logic on the status bus.

7. Read position status select. This signal gates positioner control status to the control logic on the status bus with bit assignments as follows:

	Signal	Function
Bit 15	Forward	Indicates the forward latch is set.
Bit 14	Reverse	Indicates the forward latch is cleared.
Bit 13	Cyl. pulse	Transmits cylinder pulses during a positioning function.
Bit 12	End of travel	Indicates the heads have been positioned beyond the usable recording field in either the forward or reverse direction.
Bit 11	Fine servo	Indicates the heads are being positioned under fine servo control and are less than 1/2 track from the final destination.
Bit 10	Speed & mtr on	Indicates the pack is rotating at a speed safe for flying the heads.

8. Read fault status select. This signal gates the contents of the fault register to the control logic on the status bus with bit assignments as follows:

	Signal	Function
Bit 15	$\overline{W+E+R\cdot \text{On Cyl.}}$	Indicates a write, erase, or read gate has been received while unit was off cylinder.
	or	
	$\overline{W+R\cdot \text{On Cyl.}}$	Indicates a write or read gate has been received while unit was off cylinder. (Applies to double density option only.)
Bit 14	$(W+E)\cdot R$	Indicates a write or erase gate has been received while the read gate is true.
	or	
	$W\cdot R$	Indicates a write gate has been received while the read gate is true. (Applies to double density option only.)
Bit 13	Current	Indicates one or more of the following fault conditions. a. More than one head selected. b. Erase and no write driver on. c. Write driver on and no erase. d. Both write drivers on. e. Write gate without write data.
Bit 12	+ Volt	Indicates a below normal condition of the positive voltages in the unit.
Bit 11	- Volt	Indicates a below normal condition of the negative voltages in the unit.
Bit 10	Seek	Indicates a seek error has occurred.
Bit 09	AC write fault	Indicates write gate without write data. (Applies to double density option only.)

9. Read control status select. This signal gates the following control functions to the control logic on the status bus.

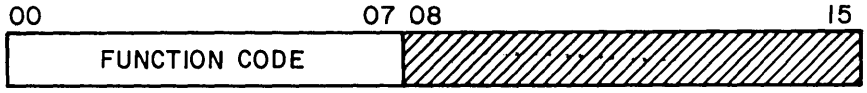
	Signal	Function
Bit 15	Sector mark	Indicates start of record.
Bit 14	Pack unsafe	Indicates a unit fault has occurred.
Bit 13	Seek error	Indicates unit was unable to complete a move within 500 milliseconds or that the carriage has moved to a position outside the recording field.
	or	
	Seek incomplete	Indicates unit was unable to complete a move within 575±175 milliseconds or that the carriage has moved to a position outside the recording field. (Applies to double density option only.)
		A return-to-zero seek command will clear the seek error or seek incomplete condition, return the heads to cylinder zero, and enable an on-cylinder signal to the control logic.
Bit 12	On cylinder	Indicates unit has completed a move to the addressed cylinder.
Bit 11	Index	Track reference signal. This signal occurs once per revolution. The leading edge is coincident with the leading edge of the sector zero sector mark.
Bit 10	Amplitude monitor 1	Indicates the average amplitude of the data signal envelope has dropped below a selected minimum value.
	or	
	EOT seek error	Indicates that the carriage has moved to a position outside the recording field. A return-to-zero seek command clears the EOT seek error condition, returns the heads to zero, and enables an on-cylinder signal to the control logic. (Applies to double density option only.)

	Signal	Function
Bit 09	End of cylinder	This line indicates the head counter has been advanced beyond head 18.
Bit 08	Amplitude monitor 2	This signal is a monitor of short duration changes in data signal amplitude with a threshold of approximately 25 percent of the average amplitude and indicates that marginal data may follow. (Not used with double density option.)
Bit 07	Amplitude monitor 3	This signal is a monitor of short duration changes in data signal amplitude with a threshold of approximately 50 percent of the average amplitude. This function is intended as an aid in certifying the pack during the formatting process. The format pattern should be all ones or zeros. This line should be monitored only during the data field. (Not used with double density option.)

NOTE

Amplitude monitors 2 and 3 have a pulse forming circuit in the output with minimum pulse duration of 5 microseconds.

STOP LOADING DIRECTORS
Function Code - 92_{16}

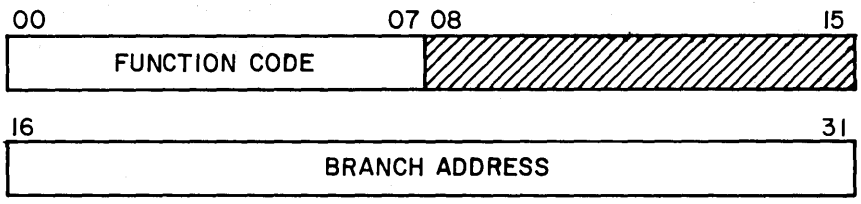


Purpose - This is a 16-bit director which will cause the control logic to stop loading directors. This director will not affect director execution.

Programming Considerations

1. This director is not loaded into the director buffer. It is decoded in a look-ahead register and immediately executed.
2. Director loading will resume when initiated by an 0018 function on normal output channel 08 (load director buffer).
3. Requires loading time only.

UNCONDITIONAL BRANCH
Function Code - 94_{16}



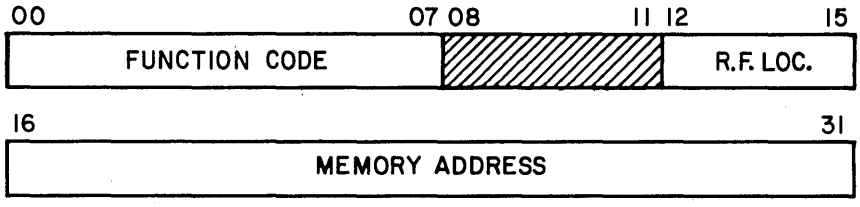
Purpose - This director will cause the control logic to unconditionally branch to the address specified by bits 16 through 31.

Programming Considerations

1. This director is not loaded into the director buffer. The director is decoded in a look-ahead register and executed immediately. Succeeding directors are loaded from the branch address location.

STORE REGISTER FILE 3-248

Function Code - 98_{16}



Purpose - This director causes the contents of the register file location specified by bits 12 through 15 to be transferred to the memory location specified by bits 16 through 31.

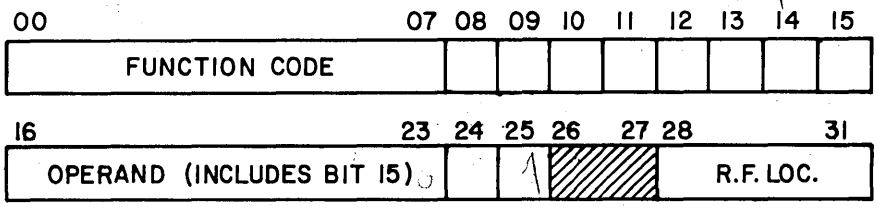
Programming Considerations

1. This director must not be executed unless the word counter is at zero. This indicates there is no data transfer in progress between the control logic and core memory.

2. The minimum execution time for this director is 1.161 microseconds. This time could increase to 2.787 microseconds depending on the length of time required to complete the memory reference.

ENABLE DISK FUNCTION

Function Code - $C0_{16}$



Purpose - This director sends control information to the disk interface. Bits 08 through 14 set tag lines which specify how the information is to be used. The required information can either be supplied from the operand field (bits 15 through 23) or from the specified register file location (bits 07 through 15). The field entries and their purposes are:

- Bit 08 Enable unit select
- Bit 09 Disable unit select
- Bit 10 Load difference register
- Bit 11 Load cylinder register
- Bit 12 Load sector register
- Bit 13 Load head register
- Bit 14 Enable control select line
- Bit 24 Disable reserve on opposite channel
- Bit 25 Use register file location specified by bits 28 through 31

Programming Considerations

1. Figure 5-2 shows the relationship between the tag lines (bits 08 through 14) and the operand field bits. If the register file is used (bit 25 set), bits 07 through 15 of the specified register file location correspond to the operand field.

2. This director requires 3.6 microseconds to execute, but the information is presented to the drive 588 nanoseconds after execution begins.

3. Only one of the tag line bits (8 through 14 and 24) may be set at one time.

4. Director bit 08 enables the select line for the unit specified by bits 21 through 23. It will also pulse the release and clear fault lines if bit 18 or 19 is set.

5. Director bit 09 disables the unit select line.

6. Director bits 10 through 13 generate a 1.0-microsecond minimum pulse which loads the designated quantity into the specified disk drive register.

7. Director bit 14 enables the control select line which controls specific actions associated with the selected drive. When this bit is set, any operand bits which are set will enable the associated line to the drive and any operand bits which are clear will disable the associated line to the drive. Two control select functions must be issued to create a pulse; for example, one to enable the required line and one to disable the line. This method is used for the three seek operations (forward, reverse, and RTZ). When data strobe margins or carriage offset operations are required, the associated line must remain enabled for the entire operation.

NOTE

After a carriage offset instruction is issued, there must be a programmed delay to allow the positioner to reach the offset position before attempting to read data from the drive.

8. The control select line must be enabled during all data transfers to or from the drive to allow control of the read, write, and erase heads during data director execution.

9. To exit the control select sequence, a disable control select director (C8) must be executed.

10. Director bit 24 issues a pulse which releases all drives from the reserve imposed by the opposite channel. This bit should be used only when it is known that the controller on the opposite channel is inoperative and cannot release the units it has reserved.

OPERAND FIELD BITS	TAG LINE SELECT BITS						
	BIT 08	BIT 09	BIT 10	BIT 11	BIT 12	BIT 13	BIT 14
23	Unit no. 2^0	NU	2^0	2^0	2^0	2^0	NU
22	Unit no. 2^1	NU	2^1	2^1	2^1	2^1	NU
21	Unit no. 2^2	NU	2^2	2^2	2^2	2^2	Seek forward or offset
20	NU	NU	2^3	2^3	2^3	2^3	NU
19	Clr fault	NU	2^4	2^4	2^4	2^4	NU
18	Release	NU	2^5	2^5	NU	NU	Seek reverse or offset
17	***	NU	2^6	2^6	NU	NU	Return to zero seek
16	***	NU	2^7	2^7	NU	NU	Data strobe early
15	***	NU	2^8	2^8	NU	NU	Data strobe late
- ****	NU	NU	2^9*****	2^9*****	NU	NU	NU

*Bits 15 through 23 of the operand field correspond to bits 07 through 15 of the specified register file location.
**NU indicates the bit is not used.
***Reserved for external programming use.
****Not possible to load cylinder select and difference select bit 2^9 from operand field.
*****Used only with 10333-1 Double Density Option.

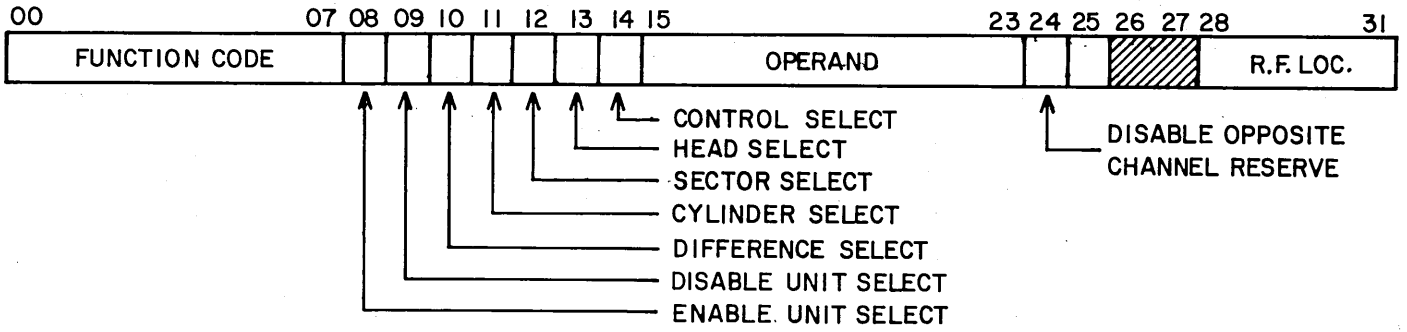
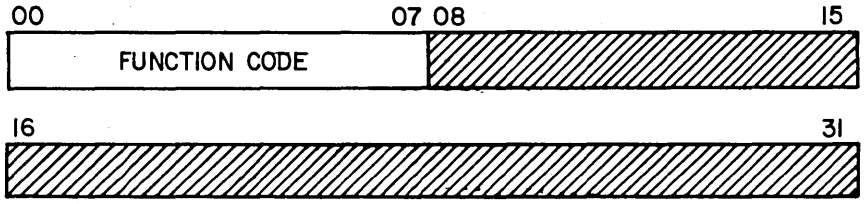


Figure 5-2. Director Bit Relationships for Disk Functions

DISABLE CONTROL SELECT

Function Code - $C8_{16}$



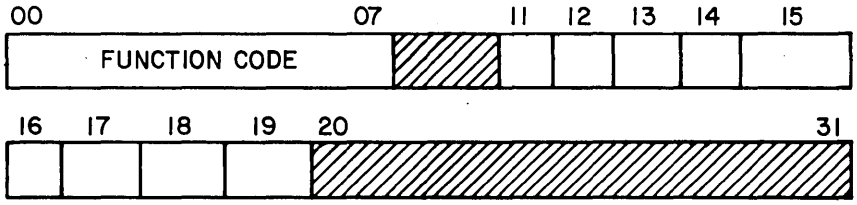
Purpose - This director drops the control select line previously enabled by the enable disk function director.

Programming Considerations

1. This director must be executed to disable the control select tag line from an enable disk function director.

ENABLE/DISABLE DISK STATUS

Function Code - $E0_{16}$



Purpose - This director enables a specific disk drive status to the hardware. The desired status is specified by setting one of the following bits.

- Bit 11 - Read cylinder register
- Bit 12 - Read difference register
- Bit 13 - Read head register
- Bit 14 - Read sector counter

- Bit 15 - Read sector register
- Bit 16 - Read interlock
- Bit 17 - Read position
- Bit 18 - Read fault
- Bit 19 - Read control

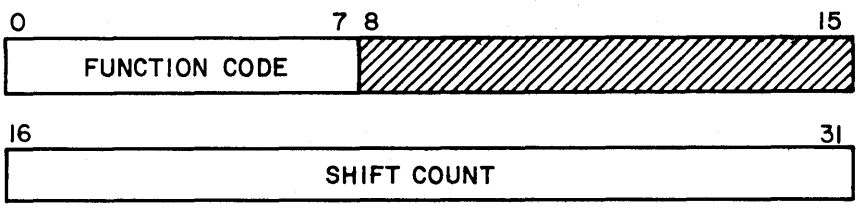
Any clear bits disable the associated status.

Programming Considerations

1. This director enables the status from the selected disk drive to the hardware. A copy disk status director must be executed to gate the selected status into a register file location.
2. One director can be used to disable a previous status selection while also selecting new status.

INITIATE ERROR CORRECTION
Function Code - $F0_{16}$

3-256



Purpose - This director tests the 32-bit checkword generator for all zeros in bit positions 11 through 31. If these bits are not all zeros, the checkword generator is shifted end-around one position, the shift count is decremented, and the test is repeated. This action is repeated until bits 11 through 31 are all zeros or until the shift count decrements to zero. (See section 8 for an explanation of error correction and a description of associated directors.)

General Considerations for Support Directors

1. Support directors, except initiate error correction, can be executed while a data director is executing, provided that in time critical situations, the support director is completed before the next data director is required. For example, support directors can not be executed during a RAP or a WAP sequence.

2. A conditional branch director will inhibit director memory references until the branch or not-branch decision has been made and the conditional branch director has completed execution.

SECTION 6

DIRECTOR SEQUENCING AND SELECTION

DIRECTOR SEQUENCING AND SELECTION

DIRECTOR SEQUENCING

Directors, once they have been loaded and execution has started, control all operations needed to perform data handling and control of the disk drive. Each director has a specific role to play in performing these operations. The director sequences must, therefore, be formatted so that all necessary operations are performed in the proper order and with due consideration for the real-time constraints imposed by the data handling process.

The directors are divided into two groups: data directors and support directors (see section 5). The data directors include all read, write, and compare directors and the delay director. These directors are executed in relation to the rotation of the disk surface and must be executed in real-time. The support directors establish supporting parameters for the data directors and need not be executed in relation to disk rotation. In most cases, it is possible to execute a support director while a data director is being executed, except if there is a conflict in data paths.

To properly format a list of directors to perform a given operation, the characteristics of each director and the format of the information of the disk surface must be known. Each of these areas should be studied thoroughly before attempting to format a director sequence. The remainder of this discussion assumes that the reader is thoroughly familiar with these areas. In order to create a meaningful discussion, only the disk format and director sequences required for an 844-2/21/41 disk storage unit will be considered.

DISK STRUCTURE FOR 844-2/21/41 DISK STORAGE UNIT

A disk pack consists of several disks on a single spindle which are divided into cylinders, tracks, and sectors. These are further defined as follows (see Figure 6-1).

- A cylinder consists of all the information under all the heads in one position. It includes one track for each disk surface in the disk pack.
- A track consists of all the information under one specific head in one position. It is further divided into sectors.

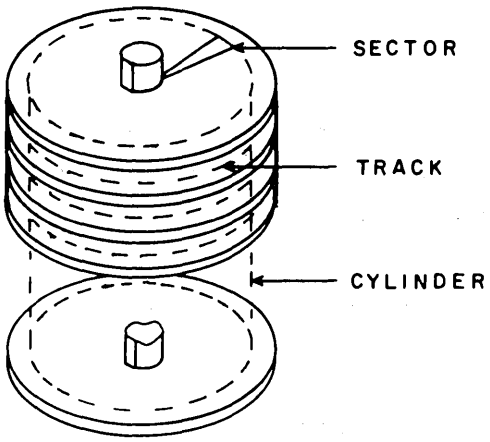


Figure 6-1. Disk Structure

- A sector is the smallest addressable area on a disk and is further divided into information fields. Figure 6-2 shows the sector format for an 844-2/21 disk unit. The total number of bits in each sector is 4722. At 147 nanoseconds per bit, total sector time is 694.134 microseconds. Figure 6-2.1 shows the sector format for an 844-41 disk unit. The total number of bits in each sector is 4485. At 155 nanoseconds per bit, total sector time is 695.2 microseconds.

DIRECTOR USAGE

A disk sector (see Figure 6-2) consists of three types of information:

1. A synchronization area which is used to synchronize the control logic with the data stream
2. A data field which may consist of either address information or actual data
3. A checkword field which contains redundant information which is used to check the validity of its associated data field

Each of these areas has certain directors associated with it. These directors are called data directors because they control data flow and formatting. These directors are listed in Table 5-1. The read address pattern and write address pattern directors are concerned with the synchronization area, the read checkword and write checkword directors are concerned with the checkword area, and the remaining directors are concerned with the data field.

To better understand this, consider a typical write operation for one field. First, the synchronization information must be written. A hardware restriction requires that all write sequences begin with a write address pattern (WAP) director. A write Nx director is then used to write a series of zeros which will be used as the synchronization area. Next, one more WAP director is executed to write the address pattern needed to achieve bit and byte synchronization when the data is read back. This is followed by a write director to write the data field and a write checkword director to write the checkword information. This provides a basic set of directors for writing a field of information as follows:

Director	Usage
WAP	Begin write sequence.
Write Nx	Write zero bytes on the disk surface for synchronizing control logic and drive electronics.
WAP	Write byte synchronization pattern.
Write	Write data field.
Write checkword	Write checkword information on the disk.

This list is not sufficient to perform the actual task of writing the data on the disk but is illustrative of the directors required.

To read this information back from the disk, it is necessary to establish bit and byte synchronization, read the data field, and read the checkword field. The following director list is typical of the directors required.

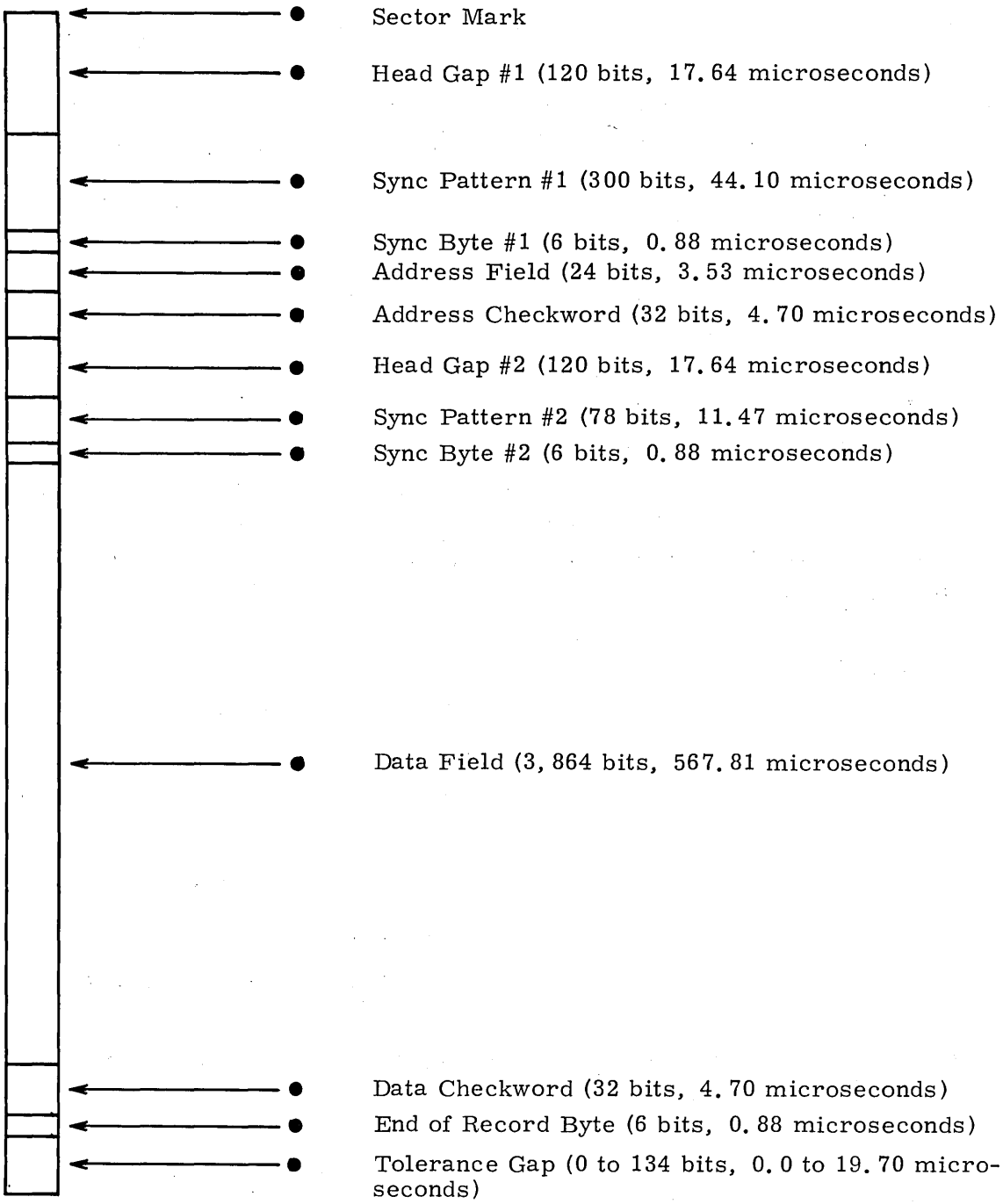


Figure 6-2. Sector Format for 844-2/21 (CYBER)

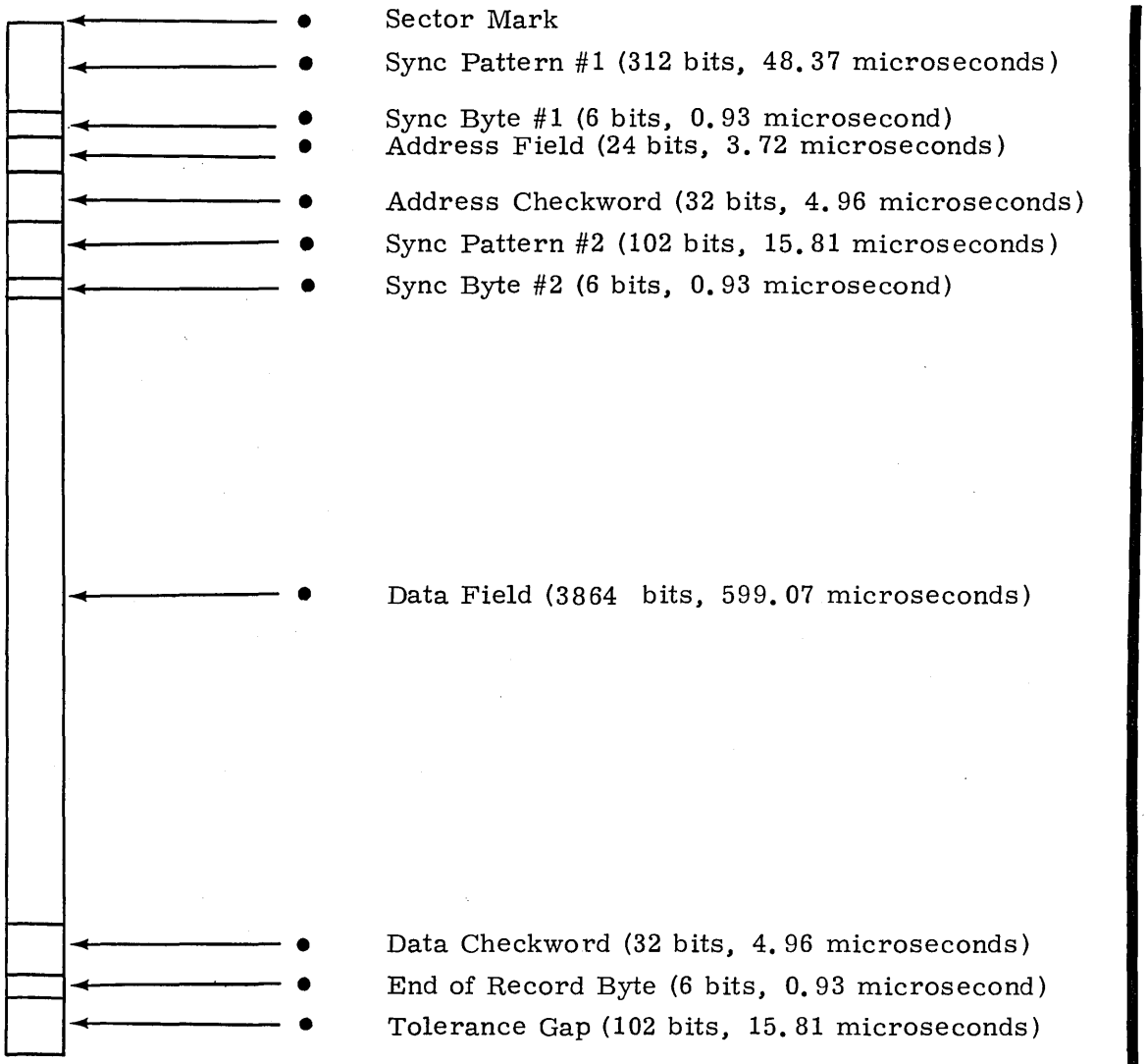


Figure 6-2.1 Sector Format for 844-41 (CYBER)

Director	Usage
Read address pattern (RAP)	This RAP looks for an all zeros pattern to synchronize control logic and disk electronics.
RAP	This RAP finds the bit synchronization pattern.
RAP	This RAP finds the byte synchronization pattern.
Read	Read the data field.
Read checkword	Read the checkword field.

This is the basic director sequence for reading a data field. By substituting a compare director for the read director, the information read from the disk will be compared with information being supplied by the data buffer.

In addition to the data directors, there are the support directors to be considered. These directors are listed in Table 5-2. They are used to augment the action of the data directors, and with certain exceptions, may be executed while data directors are in progress. Typical usages for these directors are:

- Precondition hardware to specify data paths, addresses, and direction of data flow.
- Select the desired disk drive and position its heads to the proper area.
- Enable and input status conditions.
- Branch operations.
- Arithmetic operations.
- Miscellaneous operations such as terminate or stop loading directors.

The remainder of this section explains how to construct detailed director sequences and provides examples of these sequences for specific operations.

DIRECTOR SELECTION

The preceding paragraphs of this section provided an overview of the hardware elements which affect director loading, decoding, and execution and also provided general guidelines regarding director selection. The following paragraphs provide detailed information on how director sequences are constructed and include specific examples for various basic operations.

INITIAL DIRECTOR SELECTION

The first requirement in building a director sequence is to determine all operations which are required to achieve the desired end result. In disk operations, these include the following:

1. Selection of the desired disk drive.
2. Positioning the heads to the desired cylinder and track.
3. Preconditioning hardware to select data paths and to specify the required hardware conditions.
4. Locate the proper address (sector) on the disk track.
5. Perform the desired operation (read, write, compare).
6. Read or write checkword for data verification.

When all of the above operations have been determined, directors should be selected which will perform the required actions. These directors can then be separated into three categories as follows:

1. Those operations which are completely divorced from real-time constraints such as disk selection and positioning operations.
2. Those operations which are not real-time constrained except that they must be performed in conjunction with data directors which are real-time constrained.
3. Those operations which are directly related to the rotation of the disk are therefore real-time constrained. All data directors fall into this category.

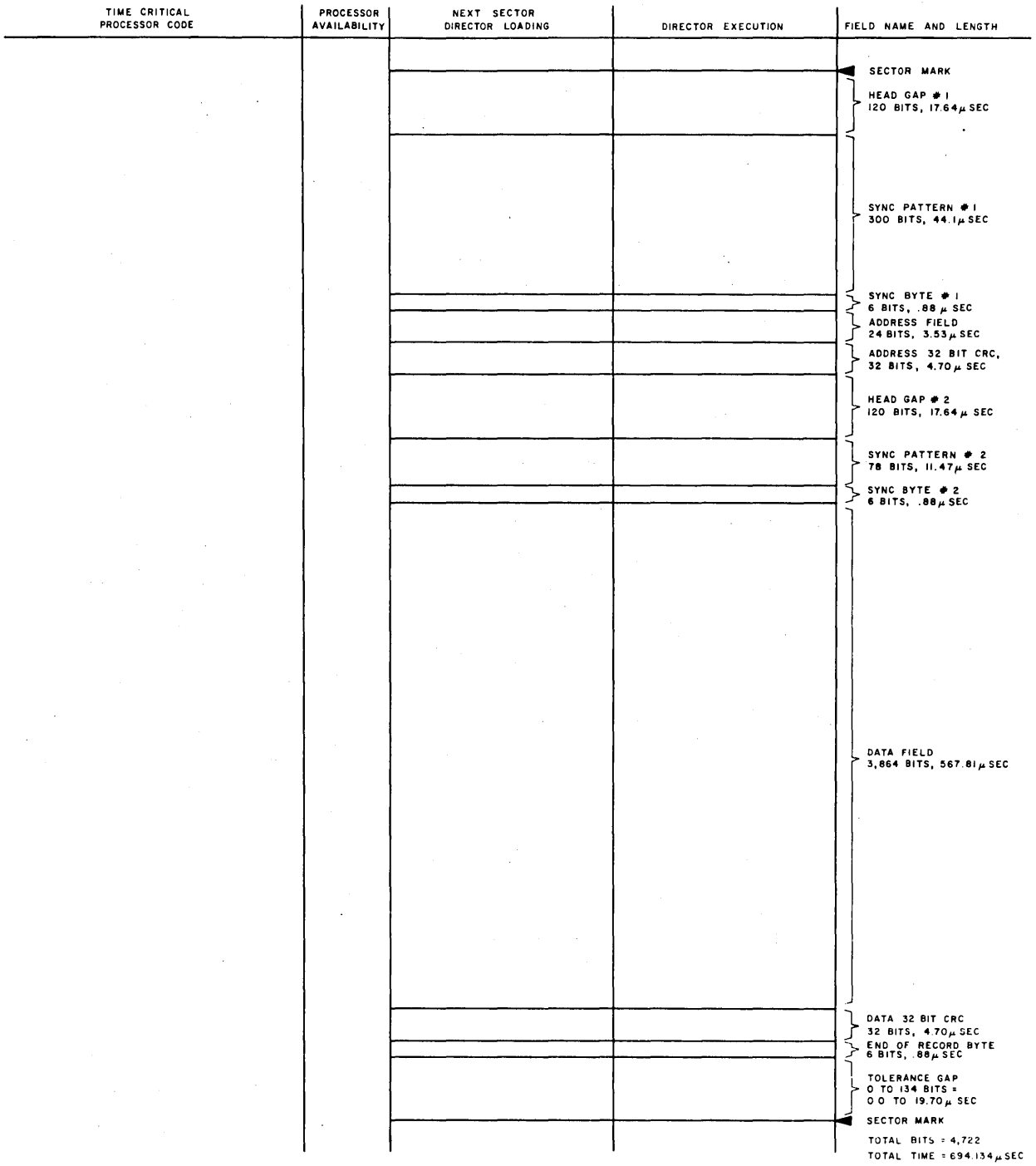


Figure 6-3. Director Timing Chart

Directors in category 1 can be initiated at any time in which a data handling director is not being executed. Normally, a list of directors required to perform several parallel positioning operations will be constructed and executed. The first drive to come "on-cylinder" can then be used for data handling while the others are being positioned. This method will save positioning time between transfers.

Directors in categories 2 and 3 must be interlaced so that the desired operations are performed within the real-time constraints of the disk rotation. The best method of sequencing these directors is to construct a chart similar to that shown in Figure 6-3. This chart shows time flow from top to bottom and contains a row for each of the following actions.

1. Director loading
2. Direction execution
3. Subsystem processor actions

The chart also shows the format of a disk sector so that all operations can be visualized with respect to disk rotation. Proper use of the chart will result in director sequences in which all of the above actions can be performed without conflicts.

The remainder of this section contains director lists which perform specific operations. Each director in the lists is described so that its function in the operation is apparent. A sequencing chart is provided where applicable, so that the reader can understand how the director sequences were derived.

PRELIMINARY SEQUENCES

Prior to performing a data transfer operation (read, write, or compare sequence), the hardware must be preconditioned for the transfer. There is no specific sequence for performing this process as the operations involved are not time-critical. The directors involved are those which fall into category 1 operations which were discussed earlier. These operations will be used to select the desired drive, check its status, load the proper drive registers, and position the heads to the desired track.

Additional directors would be used to precondition the control logic for the specific data transfer. For the read and write sequences which are discussed in subsequent paragraphs, this involves loading the register file with the required address patterns and sector counts and establishing the necessary data paths.

Section 5 provides sufficient detail on the directors required to perform these operations. Since they are not time-critical, no timing chart or sample sequences are provided.

READ SEQUENCE (1:1 INTERLACE)

Figure 6-4 illustrates the use of the director timing chart to create a read sequence. This is a repetitive sequence which will transfer a number of sectors of data using a 1.1 sector interlace. The method of entering and exiting the sequence is not shown. The basic purpose of this chart is to depict the time-critical areas of director loading and director execution. Table 6-1 lists all of the directors used in this sequence with a description of each director's purpose in the sequence.

WRITE SEQUENCE (1:1 INTERLACE)

Figure 6-5 illustrates the use of the director timing chart to create a write sequence. This is a repetitive sequence which will transfer a number of sectors of data using a 1:1 sector interlace. The method of entering and exiting the sequence is not shown. The basic purpose of this chart is to depict the time-critical areas of director loading and director execution. Table 6-2 lists all of the directors used in this sequence with a description of each director's purpose in the sequence.

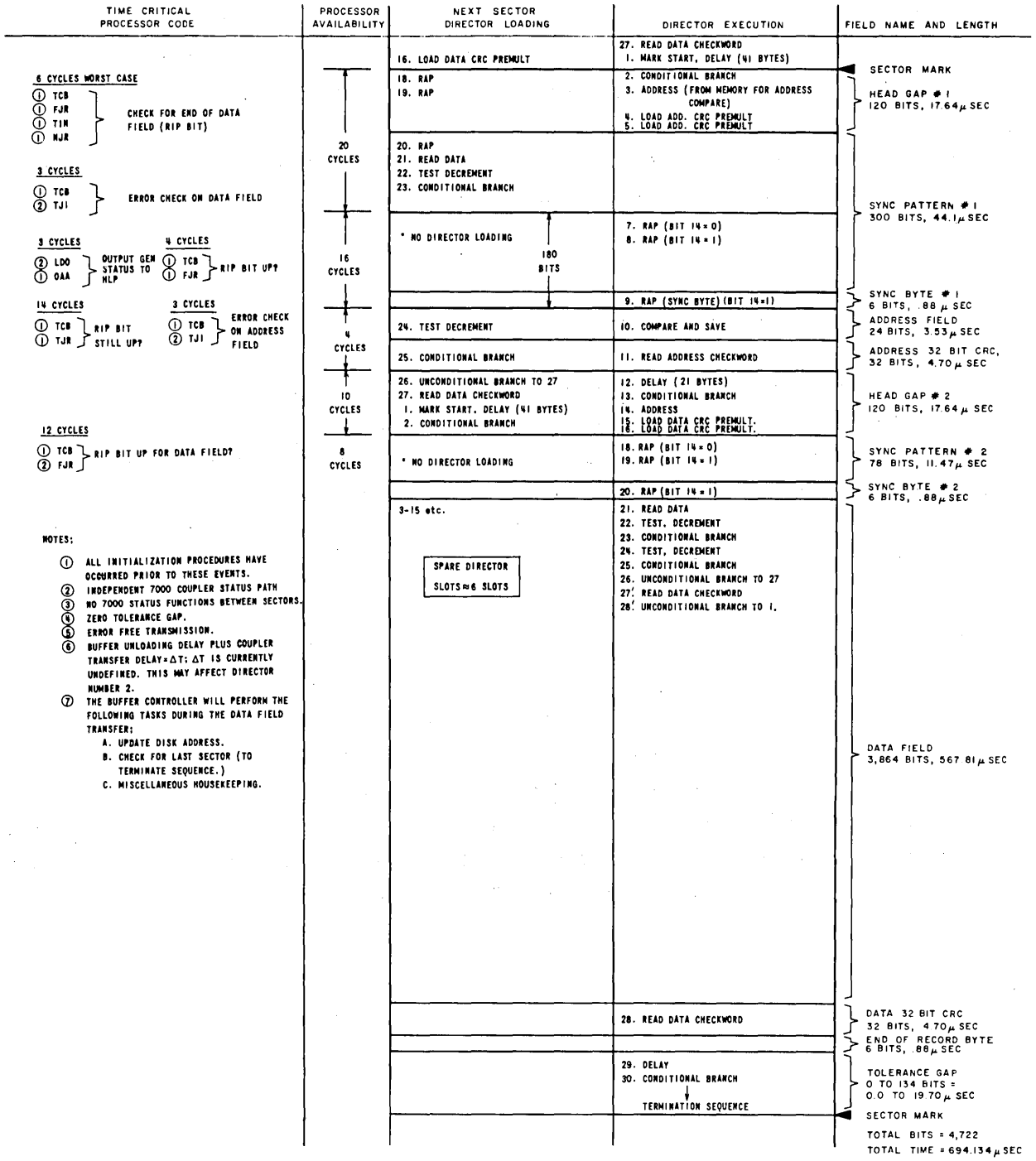


Figure 6-4. Read Sequence Timing Chart for 1:1 Interlace

TABLE 6-1. READ SEQUENCE DIRECTOR LIST (1:1 INTERLACE)

Item	Director Name	Director Usage
27	Read Checkword force 4 in Byte In force 8 bit mode	This director reads the data checkword for the previous sector. It executes when the I/O length for the previous sector decrements to zero.
1	Delay, Mark Start	This director allows the immediately preceding checkword to be analyzed for an error condition. It also keys off the sector notch to ensure that the next data director is decoded and executed within the sync pattern area. Delay period is 41 bytes.
2	Conditional Branch	This director determines if an error occurred in the previous data transfer. If an error occurred, the hardware will branch to a new set of directors. If no error occurred, director execution continues sequentially.
3	Address	This director obtains, from memory, the disk address which was supplied by the higher level processor. This will be compared to the address field read from the disk to ensure that the proper sector is being read. The disk address is loaded from memory into the data buffer upon execution of this director. This loading time preempts director loading time.
4	Load Polynomial	This loads the first 16 bits of the multiply polynomial used to check the address field checkword.
5	Load Polynomial	This loads the second 16 bits of multiply polynomial.
6	Deleted	

TABLE 6-1. READ SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
7	Read Address Pattern 38008483	This is a RAP anyplace which searches for an all zeros pattern.
8	Read Address Pattern 38000713	This is a RAP anyplace which searches for a 1 bit. This should be the start of the sync byte. When the condition is found, the bit counter is set so the sync byte can be interrogated properly after it is shifted into the bit counter. As soon as this director is decoded, all director loading stops.
9	Read Address Pattern 38010003	This is a RAP specific which checks for the sync byte. If the sync byte is not found the sequence will be aborted. If the sync byte is found, the sequence continues sequentially.
10	Compare 64040004	This is a compare and save director which compares the address field data with the information in the data buffer which was obtained by director 3. The address field is saved in the register file in case an error occurs.
11	Read Checkword 30000000	This director reads the address field checkword.
12	Delay	This director provides a 21 byte delay to ensure that the next data director executes within sync pattern 2.
13	Conditional Branch	This director checks for a checkword error condition or a compare error condition. If no error exists, director execution continues sequentially.

TABLE 6-1. READ SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
14	Address	This director establishes the data path for the data which will be read from the data field.
15	Load Polynomial 8469201 0	This director loads the first 16 bits of the multiply polynomial used to check the data field.
16	Load Polynomial 8403480 8	This director loads the second 16 bits of the multiply polynomial.
17	Deleted	
18	Read Address Pattern	Directors 18 through 20 perform the same functions as directors 7 through 9.
19	Read Address Pattern	
20	Read Address Pattern	
21	Read	This director reads the data field.
22	Test	This director checks the sector count in the register field to determine if this is the last sector to be read. It also decrements the count after testing.
23	Conditional Branch	This director checks the results of the preceding test director. If this is the last sector, branch to director 28. If not, continue sequentially.
24	Test	This director checks the sector count in the register file to determine if this is the last sector on the track. It also decrements the count after testing.

TABLE 6-1. READ SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
25	Conditional Branch	This director checks the results of the preceding test director. If this is the last sector on the track, branch to director 27' (27 prime). If not, continue sequentially.
26	Unconditional Branch	This director is not loaded into the director buffer. It causes a branch to the beginning of this sequence (director 27).
27'	Read Checkword	This director reads the checkword for the data field. It is identical to director 27 except that bit 15 is set which will cause the heads to automatically switch to the next track.
28'	Unconditional Branch	This director causes an unconditional branch to director 1. It is not loaded into the director buffer.
28	Read Checkword	This director is executed only if this is the last sector of data to be transferred. It performs the same operation as director 27.
29	Delay	This director allows the immediately proceding read checkword to be analyzed for an error condition at the proper time. The delay period is 40 bytes.
30	Conditional Branch	This director performs the same function as director 2.
-	----------	The remaining directors in this sequence would constitute a termination sequence.

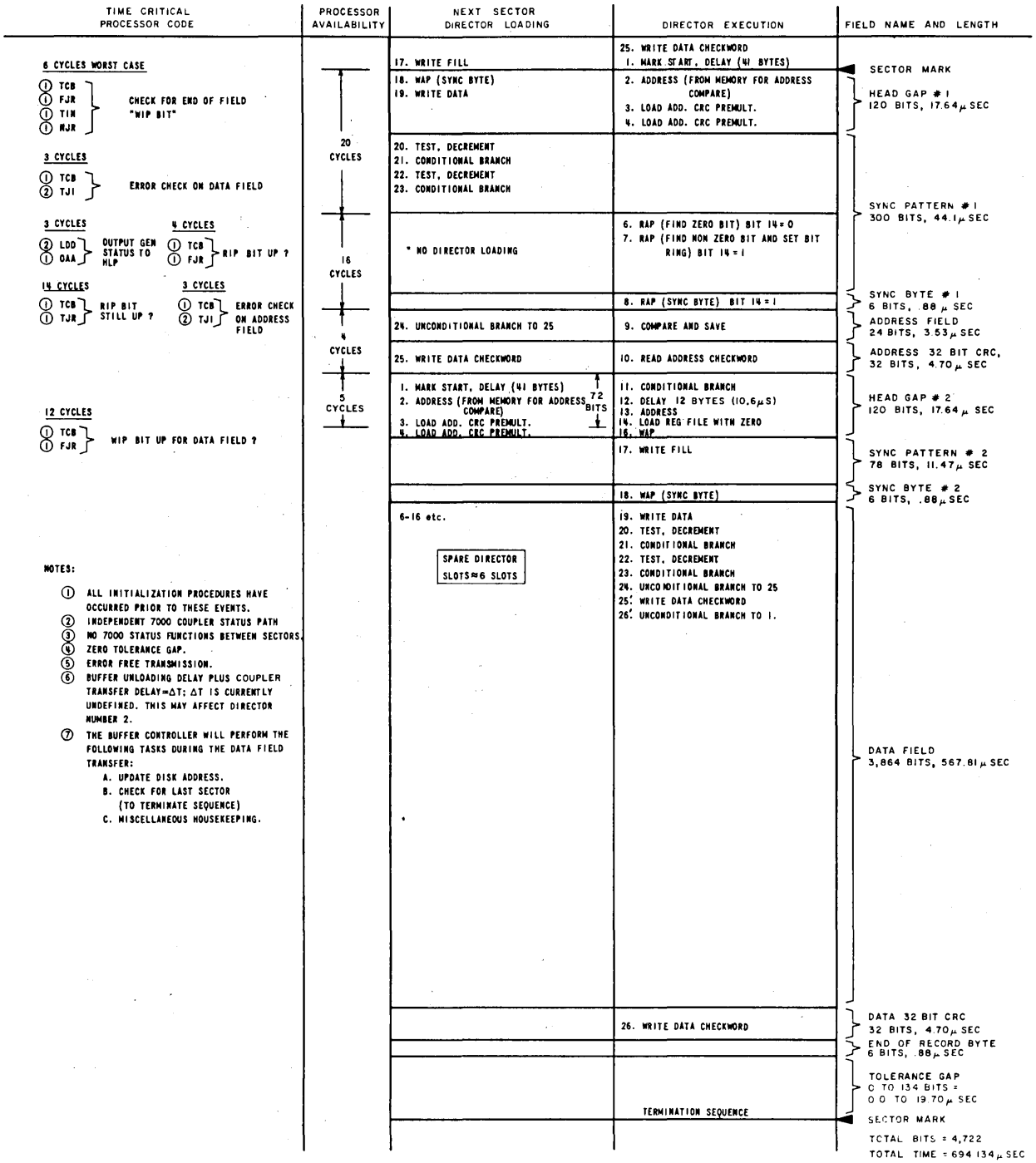


Figure 6-5. Write Sequence Timing Chart for 1:1 Interlace

TABLE 6-2. WRITE SEQUENCE DIRECTOR LIST (1:1 INTERLACE)

Item	Director Name	Director Usage
25	Write Checkword	This director executes when the previous sector's write operation is complete. In addition, it performs the following: • Turns off erase head at beginning of execution. • Writes the 32-bit checkword on the disk. • Writes the pad byte. 6 bits • Clears the write in progress bit. • Turns off the write gate at completion.
1	Delay, Mark Start	This director is used to ensure that the next data director is decoded and executed within the sync pattern. The delay period is 41 bytes.
2	Address	This director obtains, from memory, the disk address supplied by the higher level processor. This will be compared to the address field read from the disk to ensure that the proper sector is being read. The disk address is loaded from memory into the data buffer upon execution of this director. This loading time preempts director loading time.
3	Load Polynomial	This loads the first 16 bits of the multiply polynomial used to check the address field checkword.
4	Load Polynomial	This loads the second 16 bits of the multiply polynomial.
5	Deleted	

TABLE 6-2. WRITE SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
6	Read Address Pattern 3800249B	This is a RAP anyplace which searches for the all zeros pattern.
7	Read Address Pattern 3800074B	This is a RAP anyplace which searches for a 1 bit. This should be the start of the sync byte. When the condition is found, the bit counter will be set so the sync byte can be interrogated properly after it is shifted into the bit counter. As soon as it is decoded, director loading stops.
8	Read Address Pattern 3701000B	This is a RAP specific which checks for the sync byte. If the sync byte is found, the sequence continues sequentially. If not, the sequence is aborted.
9	Compare 6264004	This is a compare and save director which will compare the address field data with the information in the data buffer obtained by director 2. The address field data read off the disk, is saved in the register file in case an error occurs.
10	Read Checkword 3000000	This director reads the address checkword.
11	Conditional Branch	This director checks for a checkword error condition or a compare error condition. If no error exists, director execution continues sequentially. Bit 15 must be set so that the erase heads are turned on by this director.
12	Delay	This director provides a 12-byte delay to ensure that the next data director executes within head gap 2.

TABLE 6-2. WRITE SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
13	Address	This director establishes the data path for the data which will be written in the data field.
13.5	Deleted	
14	Load Register File	This director sets register file location 0 equal to 0000.
15	Deleted	
16	WAP	This director turns on the write gate to establish sync.
17	Write Nx	This director writes 20 bytes of zeros on the disk.
18	WAP	This director writes the sync byte.
19	Write	This director writes the data field.
20	Test	This director checks the sector count in the register file to determine if this is the last sector to be written. It also decrements the count after testing.
21	Conditional Branch	This director checks the results of the preceding test director. If this is the last sector, branch to director 26. If not, continue sequentially.
22	Test	This director checks the sector count in the register file to determine if this is the last sector on the track. It also decrements the count after testing.
23	Conditional Branch	This director checks the results of the preceding test director. If this is the last sector on the track, branch to director 25' (25 prime). If not, continue sequentially.

TABLE 6-2. WRITE SEQUENCE DIRECTOR LIST (1:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
24	Unconditional Branch	This director is not loaded into the director buffer. It causes a branch to the beginning of this sequence (director 25).
25'	Write Checkword	This director writes the checkword for the data field. It is identical to director 25 except that bit 15 is set which causes the heads to automatically switch to the next track.
26'	Unconditional Branch	This director is not loaded into the director buffer. It causes an unconditional branch to director 1.
26	Write Checkword	This director is executed only if this is the last sector to be transferred. It performs the same operation as director 25.
--	----------	The remaining directors would constitute a termination sequence.

READ SEQUENCE (2:1 INTERLACE)

Figures 6-6 and 6-7 illustrate the use of the director timing chart to create a read sequence at a 2:1 interlace. Figure 6-6 shows the operations which must occur during the dead sector. Figure 6-7 shows the operations which must occur during the active sector. The two sheets form a repetitive sequence which will loop until all desired sectors are read. The method of entering and exiting the sequence is not shown. Table 6-3 lists all of the directors used in this sequence with a description of each director's purpose in the sequence.

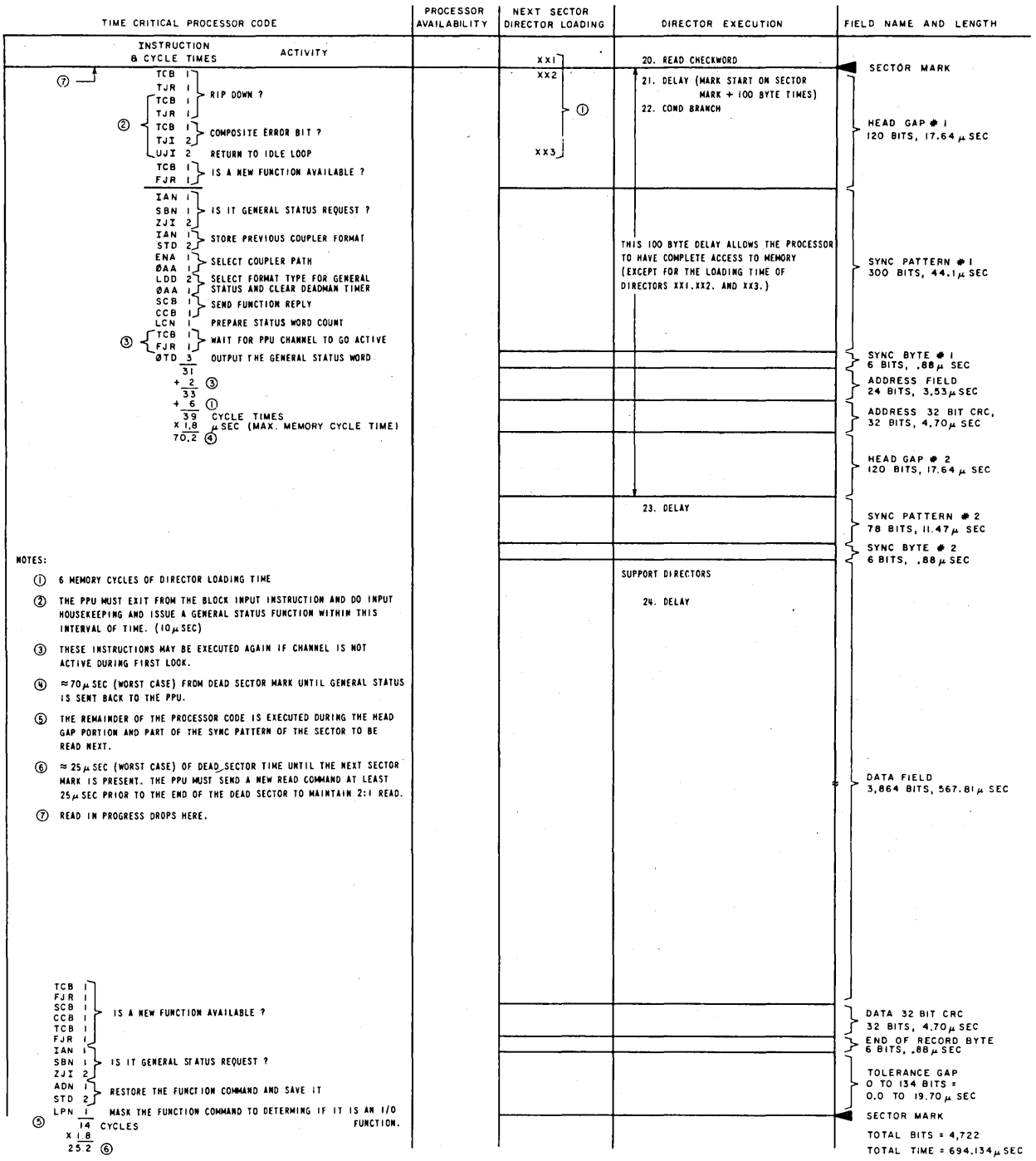


Figure 6-6. Read Sequence Timing Chart for 2:1 Interlace, Dead Sector

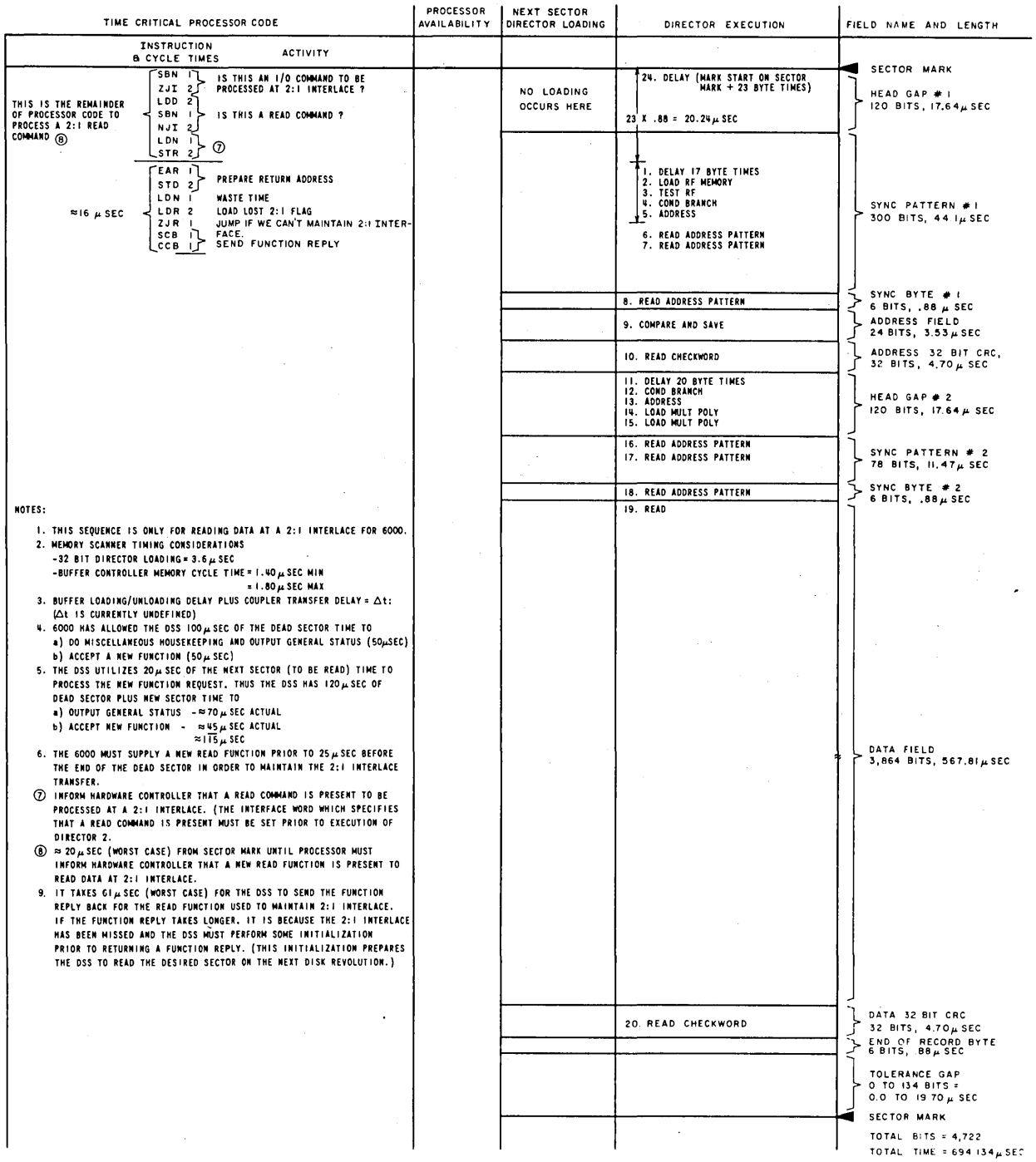


Figure 6-7. Read Sequence Timing Chart for 2:1 Interlace, Active Sector

TABLE 6-3. READ SEQUENCE DIRECTOR LIST (2:1 INTERLACE)

Item	Director Name	Director Usage
20	Read Checkword	This director reads the data checkword for the previous sector. It executes when the previous read command terminates.
21	Delay, Mark Start	This director keys off the sector notch of the dead sector which follows a data sector in 2:1 interlace mode. A delay period of 100 bytes follows the sector notch. This delay allows the subsystem processor to have complete access to memory during the delay period (except for the loading time of the directors which replace those directors executed under the 100-byte delay period).
22	Conditional Branch	This director determines if an error occurred in the previous data transfer. If an error occurred, the hardware will branch to a new set of directors. If no error occurred, director execution continues sequentially.
23	Delay	This is a 1-byte delay which prevents the execution of the following support directors until the previous 100-byte delay has expired.
	Support Directors	These support directors are utilized to prepare the drive unit to handle the next data transfer at the next 2:1 interlace disk address. The address field premultiply polynomial is prepared during this time.
24	Delay, Mark Start	This director keys off the sector notch of the next data sector to be read. A delay period of 23 byte times ($\approx$ 20 μsec) follows the sector notch. This delay allows

TABLE 6-3. READ SEQUENCE DIRECTOR LIST (2:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
24	Delay, Mark Start	the subsystem processor to have complete access to memory during the delay period. Prior to or during this delay period, the subsystem processor must accept a new read command from the HLP and set the appropriate decode flag. Read at 2:1 interlace will not continue if the decode flag is not set prior to the end of the delay period.
1	Delay	This is a 17-byte delay ($\approx$ 15 μsec) which causes the control logic to start searching for the sync pattern 240 bit times from the sector notch. The following four support directors execute under this delay period.
2	Load RF Memory	Load the decode flag from memory.
3	Test	Determine if a new read command has been accepted by the subsystem processor.
4	Conditional Branch	Branch out of the read at 2:1 interlace director sequence if a read command was not present.
5	Address	This director obtains, from memory, the disk address which was supplied by the higher level processor or updated by the subsystem processor. This will be compared to the address field read from the disk to ensure that the proper sector is being read. The disk address is loaded from memory into the data buffer upon execution of this director. This loading time preempts director loading time.

TABLE 6-3. READ SEQUENCE DIRECTOR LIST (2:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
6	Read Address Pattern	This is a RAP anyplace which searches for an all zeros pattern. This director also causes the read gate to turn on.
7	Read Address Pattern	This is a RAP anyplace which searches for a 1 bit. This should be the start of the sync byte. When the condition is found, the bit counter is set so the sync byte can be interrogated properly after it is shifted into the bit counter.
8	Read Address Pattern	This is a RAP specific which checks for the sync byte. If the sync byte is not found, the sequence will be aborted. If the sync byte is found, the sequence continues sequentially.
9	Compare	This is a compare and save director which compares the address field data with the information in the data buffer which was obtained by director 5. The address field is saved in the register file in case an error occurs.
10	Read Checkword	This director reads the address field checkword.
11	Delay	This director provides a 20-byte delay to ensure that the next data director executes within sync pattern 2.
12	Conditional Branch	This director checks for a checkword error condition or a compare error condition. If no error exists, director execution continues sequentially.
13	Address	This director establishes the data path for the data which will be read from the data field.

TABLE 6-3. READ SEQUENCE DIRECTOR LIST (2:1 INTERLACE) (Cont'd)

Item	Director Name	Director Usage
14	Load Polynomial	This director loads the first 16 bits of the premultiply polynomial used to check the data field.
15	Load Polynomial	This director loads the second 16 bits of the premultiply polynomial.
16	Read Address Pattern	Directors 16 through 18 perform the same function as directors 6 through 8.
17	Read Address Pattern	
18	Read Address Pattern	
19	Read	This director reads the data field.

COMPARE SEQUENCE (1:1 INTERLACE)

The compare sequence is very similar to the read sequence (1:1 interlace) except that the compare director is substituted for the read director and tests are performed to determine the results of the compare. The address director (item 14 of Figure 6-4) establishes the data path for the compare data from the memory or system coupler. Since this sequence is identical to the read sequence in all time-critical areas, the individual directors are neither illustrated nor explained.

SECTION 7

SPECIAL HARDWARE CONSIDERATIONS

SPECIAL HARDWARE CONSIDERATIONS

INTRODUCTION

The control logic contains some hardware elements which may affect program execution. The effect of these elements must be considered before any director sequences are constructed. The following paragraphs discuss these elements.

MEMORY SCANNER

Since the memory unit is shared by the subsystem processor and the control logic, a memory scanner is used to avoid all memory conflicts which could arise between these two elements.

The memory scanner assigns priorities as follows:

First priority - Data handling references from control logic

Second priority - Director buffer references from control logic

Third priority - Subsystem processor references

Although the subsystem processor is given the lowest priority, it is not normally tied to any real-time constraints and will have a larger proportion of access time available to it. Each of these priorities is discussed below, including effective memory cycle time.

DATA HANDLING REFERENCES

When data is being transferred to and from memory, the memory scanner will lock out all other references until the data buffer is full (empty when going to memory) or until the word count for the transfer is satisfied. When either of these two conditions occurs during a data transfer, the memory scanner will allow other references to memory until another data request is made. The memory cycle time for data requests is 900 nanoseconds.

DIRECTOR BUFFER REFERENCES

During director buffer loading, the subsystem processor is not locked out, but the director reference is given perference. One director reference will be made for each subsystem processor reference. With an average cycle time of 900 nanoseconds for each director reference, this gives an average director load time (two 16-bit words) of 3.6 microseconds. This time includes two subsystem processor references. A 32-bit director could load in as little as 2.7 microseconds or as much as 3.6 microseconds. The loading time will never exceed 3.6 microseconds.

SUBSYSTEM PROCESSOR REFERENCES

When the subsystem processor is the only unit accessing memory, its memory cycle time is 1.60 microseconds typical (1.40 minimum, 1.80 maximum). When it is sharing the memory with the director references, it will be given one memory access for each director reference for a memory cycle time of 1.8 microseconds.

DIRECTOR BUFFER

As directors are read from memory, they are loaded into the director buffer. This is a circular buffer which will hold 16 32-bit words. The discussion of its operation is divided into the following areas.

1. Director buffer control
2. Director loading and look-ahead decodes
3. Director execution

DIRECTOR BUFFER CONTROL

The director buffer is loaded and unloaded under control of input and output pointers and flags which set if the buffer is empty or full. The buffer is considered to be empty if the input and output pointers become equal when the output pointer is incremented by one. The buffer is considered to be full when the input and output pointers become equal when the input pointer is incremented by one. From a full condition, only the output pointer can be moved. From an empty position, only the input pointer

can be moved. The director buffer is also controlled via the normal channel interface of the subsystem processor. This control is via functions received over normal output channel 08. These function codes and their effects on the director buffer are listed below.

0020 Master clear - This function code clears the director buffer and places it in an empty condition. It also clears the address register so that the first director reference is made from location zero in memory.

0018 Load director buffer - This function code causes the director buffer to start loading directors. It will cause the director buffer to reference memory through the memory scanner. The memory location addressed will be that address which is in the address register. This will be address zero following a master clear function (0020).

0022 Execute directors - This function code causes the director buffer to begin execution. If the buffer is not empty, the first director is extracted and placed in the director register. The output pointer is then incremented by one. The next director will be extracted as soon as the first has completed execution. If the buffer is empty when this function is sent, the next director loaded into the buffer will execute immediately.

DIRECTOR LOADING AND LOOK-AHEAD DECODE

When the director buffer has received an 0018 function code (load director buffer), it accesses sequential locations in memory. After each 16-bit access, the address register is incremented by one. Since these are only 16-bit references, and the director buffer is 32 bits wide, two references are required before the input pointer is moved. The first reference (bits 00 through 15 of the director) always contains the director function code. The function code is examined before the director is loaded into the director buffer to determine if one of the following directors is involved.

1. Unconditional branch - This director is not loaded into the director buffer. Instead, the second 16 bits are obtained and immediately placed in the address register. Succeeding references are then obtained from the new address location.

2. Stop loading directors - This director is not loaded into the director buffer. It is only 16 bits in length and will stop director loading immediately after it is decoded. An 0018 function code is needed to restart director loading. Loading will continue from the address in the address register.

3. Address director - This director is 64 bits in length. When it is decoded by the look-ahead decode, the next three memory references are placed into the director buffer without being examined for any special conditions. This prevents an address reference from being decoded as a director.

If none of the above conditions exist, the first 16 bits of the director are loaded into the director buffer and another memory reference is made. The next 16 bits are not examined by the look-ahead decode since these bits will not contain a function code. As soon as these bits have been loaded into the director buffer, the input pointer is incremented and another memory reference is made.

DIRECTOR EXECUTION

As soon as an 0022 function code (execute directors) is received (assuming the director buffer is not empty), the first director is transferred from the director buffer to the director register and the output pointer is incremented by one. The director is immediately decoded and starts an execution cycle which strobes the director information out of the director register. The next director is immediately transferred from the director buffer into the director register. It is decoded while the first director is completing its execution. When execution of the first director is completed, the second begins execution, and the next is transferred into the director register for decoding. Execution of data directors is considered to be complete as soon as all director parameters have been established, even though the complete data transfer is not done. This allows support directors to be decoded and executed while data is being transferred. Another data director, however, cannot be executed until a data transfer for a previous data director is complete.

If a terminate director is decoded, no further directors are extracted from the director buffer, but the output pointer is incremented by one. Director execution will resume with the next director location as soon as an 0022 function code (execute directors) is received on the normal channel.

REGISTER FILE

This is a buffer which will hold sixteen 16-bit words. It operates under director control and is directly addressed, or used as a buffer depending on the director being executed. All direct access operations have the register file location in the associated director. For those operations which use the register file as a buffer, all accesses begin at register location zero. Input and output references share the same data lines and cannot be made simultaneously.

DATA BUFFER

The data buffer is a stack-type buffer which will hold four 16-bit words. During both read and write operations, data can be entered into one end while being taken from the other end. A data multiplexer allows access to the buffer from either the system coupler or the core memory under director control. The data buffer assembles/disassembles 16-bit words from 8-bit bytes or 6-bit characters.

ADDRESS REGISTER

This register holds the address location in memory from which directors are being loaded into the director buffer. Following a master clear from the subsystem processor, the register is clear and its first reference will be to location zero in the memory. The address register is incremented by one following each memory reference. The contents of the address register can be changed in one of two ways. An unconditional branch director changes the address register immediately so that the next reference comes from the branch address location. A conditional branch director changes the address register only if the branch condition is met.

BRANCH ADDRESS REGISTER

The branch address register is used to catch the branch address for a conditional branch director. When the decision is made to branch, the contents of the branch address register are transferred to the address register.

CLOCK

The control logic has an internal clock which is generated from a 6.8 MHz oscillator and has a 147-nanosecond cell time. This clock controls all operations concerned with writing data on the disks and most internal clocking requirements. A read clock supplied by the disk drive controls read operations and internal requirements related to read operations. (For 7054-41/-42 controllers the write clock is supplied by a 6.451 MHz oscillator in the 844-41 DSU and the clock has a 155-nanosecond cell time.)

SPECIAL PURPOSE ELEMENTS

The control logic also contains several special-purpose elements which operate under direct control of the director software or work in conjunction with the elements discussed earlier. Some of these special-purpose functions are buffer control, compare operations, branch operations, write address pattern operations, shift register, and functions directly related to disk drive operations.

ARITHMETIC OPERATIONS

With one exception, the control logic performs all arithmetic operations in two's complement. The one exception is the complement operation which is done in one's complement. In order to avoid confusion between these two numbering systems, their similarities and differences are discussed below.

Positive numbers have the same representation in both systems. Negative numbers differ by one count. The reason for this difference in the negative numbers is that the one's complement system allows two representations for a zero quantity positive zero (00000) and negative zero (11111). The two's complement system does not recognize a negative zero. The following example shows the difference in the two systems.

Count	Two's Comp	One's Comp
+2	00010	00010
+1	00001	00001
+0	00000	00000
-0	Not used	11111
-1	11111	11110
-2	11110	11101

Note that one's complement representation for a negative number is found by subtracting each bit of the number from one. For example:

```
 11111
-00010
 11101      (complement of +2 equals a -2)
```

This representation can also be obtained by merely substituting 1's for 0's and 0's for 1's.

The two's complement expression for a negative number is found by adding one to the one's complement expression. Using the example above, the two's complement expression for a negative two would be:

```
 11101  (one's complement)
+    1
 11110  (two's complement)
```

SECTION 8

ERROR DETECTION AND CORRECTION

ERROR DETECTION AND CORRECTION

The control logic uses a 32-bit checkword generator for error detection and correction. This generator and its associated polynomials can detect whether or not an error has occurred during transmission of the related data. If the error occurred in a single error burst of 11 bits or less, the correction capability will generate an 11-bit correction vector and provide an address (bit displacement from beginning of data buffer) at which to exclusively OR the correction vector. Prior to a read or write operation, a divide polynomial must be loaded into the checkword generation network. This polynomial establishes the basic characteristics of the coding mechanism. Prior to a read operation, a multiply polynomial must also be loaded into the checkword generation network. This polynomial is directly related to the length of the data field and is used to optimize error correction.

Following a read, the contents of the 32-bit checkword generator are checked. If the residue is equal to zero, the data transmission was error free. If the residue is not equal to zero, the subsystem processor can issue an Initiate Error Correction director. This director includes a shift count which is equal to the number of bits in the sector data field (due to the proper selection of the divide and multiply polynomials). Error correction consists of shifting the checkword generator once for each bit in the shift count. The divide polynomial is used during this operation. The operation continues until bits 11 through 31 of the checkword generator are all zero or until the shift count becomes zero. In the first case the error is correctable and bit 13 of normal operating status word 1 will set. Status select codes 8000 and B000 will input the correction vector and the shift count remainder, respectively. If the shift count becomes zero, the error is considered noncorrectable and bit 14 of normal operating status word 1 will set. Figure 8-1 shows the registers used during error detection and correction and their relationships to each other.

Table 8-1 contains the polynomials and shift counts used for various length data fields.

TABLE 8-1. CHECKWORD GENERATOR DATA

Data Field Length*	Upper Divide Polynomial	Lower Divide Polynomial	Upper Multiply Polynomial	Lower Multiply Polynomial	Shift Count
24 bits	00A0	0805	3DC4	01EE	0018
3840 bits	00A0	0805	E360	0713	0F00
3864 bits	00A0	0805	6920	0348	0F18
32,768 bits	00A0	0805	61E0	230F	8000
42,987 bits	00A0	0805	00A0	0805	A7EB

*This column is decimal, all other columns are hexidecimal.

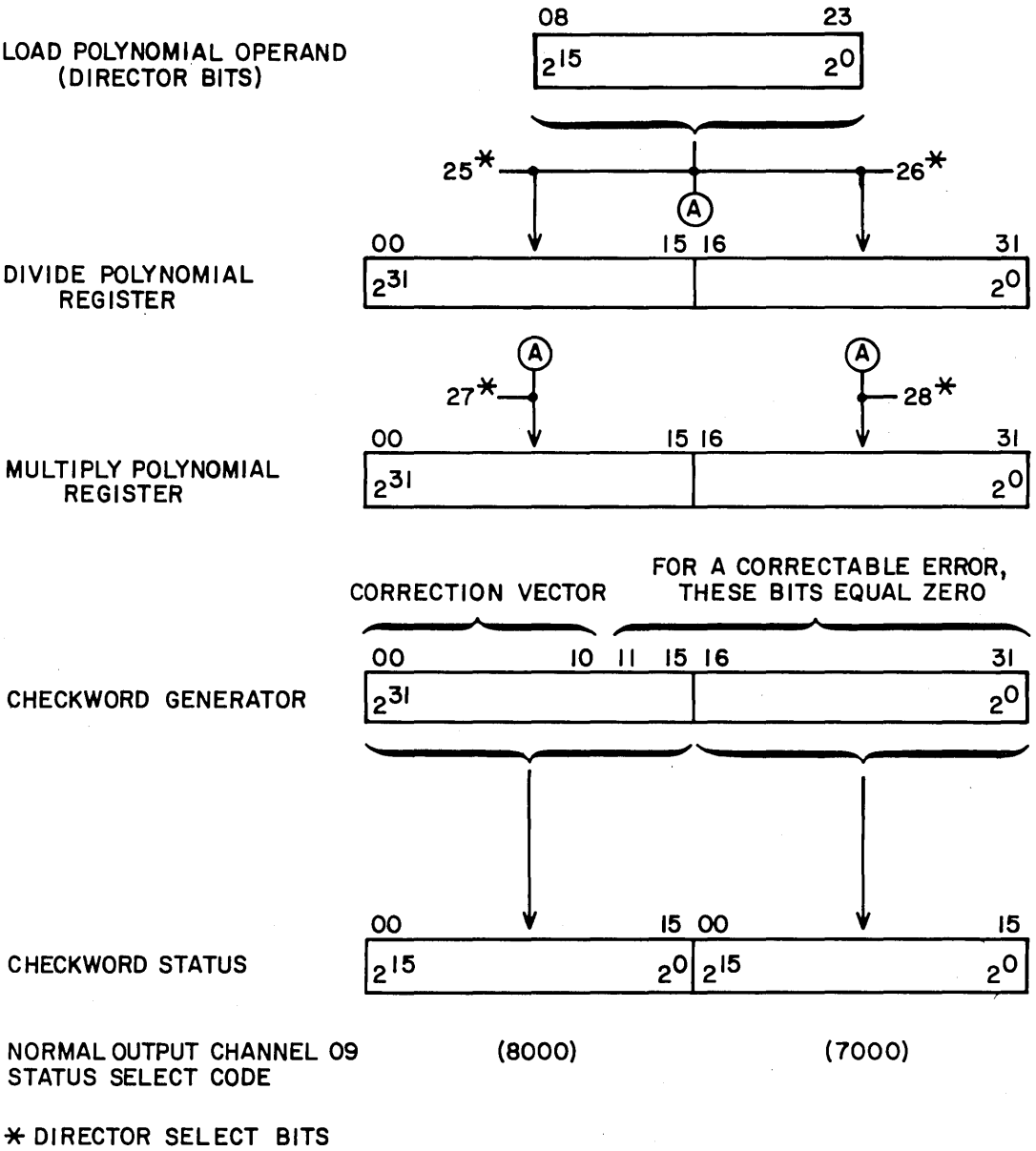


Figure 8-1. Registers Associated With Checkword Generation

SECTION 9

6000 COUPLER PROGRAMMING

6000 COUPLER PROGRAMMING

INTRODUCTION

The 6000 Coupler used with the FA710, FA719, and FA722 Disk Storage Controllers provides the hardware interface between a Control Data 6000 series data channel and the controller. This section describes the coupler/subsystem processor interface, the coupler/control logic interface, autoload, coupler function/reserve/connect, and the coupler deadman timer.

SUBSYSTEM PROCESSOR INTERFACE

The coupler exchanges data and control signals with the block transfer interface, normal channel interface, and station control interface. The block transfer interface exchanges all I/O control signals and contains the normal data path between the coupler and the processor. The normal channel interface receives coupler status and sends out the path select and format select. The coupler sends signals to the station control interface during autoload operations only. (Unless otherwise noted, the subsystem processor will be called the processor in the subsequent text.)

SIGNALS FROM THE COUPLER TO THE BLOCK TRANSFER INTERFACE

Coupler Ready (CREADY)

This signal indicates that the coupler will respond immediately to an input block transfer or output block transfer instruction in the processor.

Coupler Reply (CREPLY)

This pulse indicates that the coupler has accepted a data byte from the processor or that the coupler has a data byte available for the processor. The coupler responds to an input request or output ready with this signal.

Coupler Terminate (CCTERM)

Following an input r. .iest or an output ready, the coupler can terminate an input block transfer or output block transfer instruction by sending this signal. The processor will not send a parity strobe.

Data (CCI-00 to CCI-15)

These 16 lines carry data pulses from the coupler to the processor.

SIGNALS TO THE COUPLER FROM THE BLOCK TRANSFER INTERFACE

Output Ready (OUTRDY)

This pulse indicates that the processor has data available for the coupler.

Input Request (INPREQ)

This pulse indicates that the processor can receive another byte of data from the coupler.

Parity Strobe (CK-PAR)

This pulse indicates that the processor has accepted a data byte from the coupler.

Master Clear (MC-CC)

This pulse clears registers and controls in the coupler.

End of Operation (EOP-CC)

This pulse accompanies the last output ready signal during a block transfer to indicate that the transfer is complete. It will also be returned to the coupler following a terminate signal from the coupler, or following the reply to the last input request during a block transfer.

Data (CC0-00 to CC0-15)

These lines carry data pulses from the processor to the coupler.

NORMAL CHANNEL INTERFACE

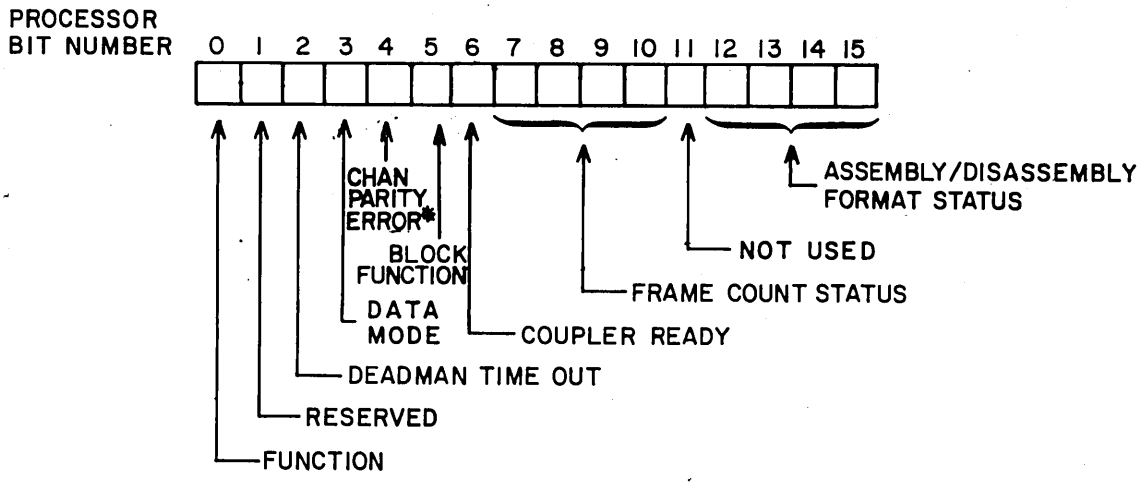
Normal Input Channel Data (I03-00 to I03-15)

These 16 lines provide a multiplexed data path for up to 64 status bits. The channel select signals determine which status byte will be sent to the processor.

Not Select Input Channel 0 (SELI-0)

A 0 on this line gates coupler status through the multiplexer to the processor on the input channel data lines. Bit definitions are listed below.

Processor Normal Input Channel 0



Function BIT 0

A 1 in this bit indicates that a function code is available on normal input channel 3.

Reserved BIT 1

A logical 1 in this bit indicates that the coupler is in an operational mode (not idling). This bit sets when the coupler processes a function code from the PPU and clears at the completion of the subsequent data transfer.

* FA722-A/B only

NICO

Deadman Timeout BIT-2

A 1 in this bit indicates that the deadman timer has expired and the coupler has been disconnected.

Data Mode BIT 3

A logical 1 in this bit indicates that the coupler is reserved and the PPU has activated the channel. The coupler is now ready to transfer data.

Channel Parity Error (FA722-A/B only) BIT 4

A 1 in this bit position indicates that the coupler detected a parity error on the PPU I/O channel. This bit is cleared by bit 04 of output channel 0. (Note: This bit is used in a CYBER 170 configuration only).

Block Function BIT 5

A 1 in this bit position indicates that the coupler will not process a function code from the PPU. Block function does not clear until the processor sends an EOP or sets the inactive bit (bit 2^1 of NOC 0), or if the coupler data mode is cleared by terminating an operation in any way.

Coupler Ready BIT 6

A 1 in this bit indicates that the coupler is ready to transfer data to/from the processor or control logic. This bit is always a 1 if the data path between the control logic and processor is selected.

Frame Count

Bits 7, 8, 9, and 10 carry a code to indicate frame count in the coupler. The processor must use this code along with format to identify valid data bits in the last word of a data transfer from the PPU. The frame count and format must be sampled before selecting another format, and before the PPU activates the channel for the next transfer. Frame count codes are listed below and discussed at the end of this section.

Count	Bit 7	8	9	10
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0

BIT 11 - COUPLER FILL LAST BYTE ALL BITS NOT NECESSARILY VALID

A/D Format

Bits 12, 13, 14, and 15 contain a code indicating in which assembly/disassembly format the coupler is operating during a data transfer with the PPU. This code, along with frame count, identify valid data bits in the last byte of a data transfer from the PPU. The processor must check this before changing formats and before the PPU activates the channel for the next transfer. The format codes are listed below.

Format	Bits 12	13	14	15
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

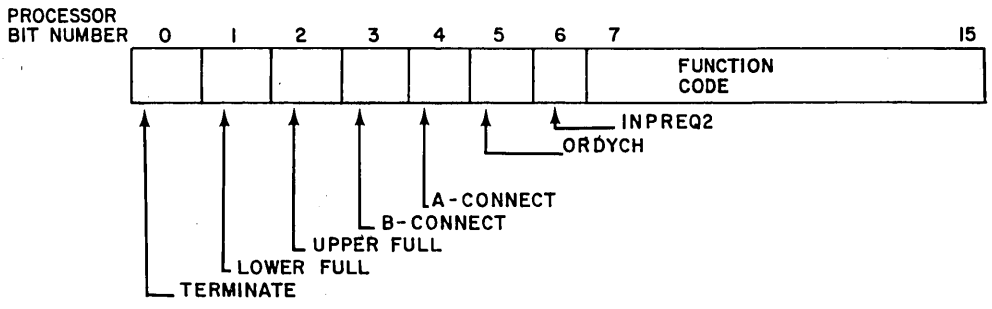
Not Select Input Channel 2 ($\overline{\text{SELI-2}}$)

A 0 on this line gates 16 coupler data bits from the PPU to the processor as status. This allows the processor to monitor data sent from the PPU to the control logic.

Not Select Input Channel 3 ($\overline{\text{SELI-3}}$)

A 0 on this line gates 9-bit function codes from the PPU to the processor. The function bits are placed in the upper 9 bits of the input channel. Function bit 0 (2^0) is placed in channel bit position 15 (2^0).

Status bits are also available to the processor on normal input channel 3. These status bits reflect conditions in the coupler and are defined as follows:



Terminate (Bit 0): A 1 in this bit position indicates that the coupler has terminated an operation. Until this condition is cleared by a PPU function pulse or a master clear, the coupler will not respond to I/O signals from the processor or control logic.

Lower Full (Bit 1): A 1 in this bit position indicates that the lower 16-bit half of the coupler data register contains data.

Upper Full (Bit 2): A 1 in this bit position indicates that the upper 16-bit half of the coupler data register contains data.

B-Connect (Bit 3): A 1 in this bit position indicates that the coupler is connected to PPU B, indicating that only PPU B can access the coupler. This bit is valid only in a dual access coupler.

A-Connect (Bit 4): A 1 in this bit position indicates that the coupler is connected to a PPU A, indicating that only PPU A can access the coupler. This bit is valid only in a dual access coupler.

ORDYCH (Bit 5): A 1 in this bit position indicates that the coupler is receiving an output ready from the processor or a COREQ (output data request) from the control logic. BC or AC

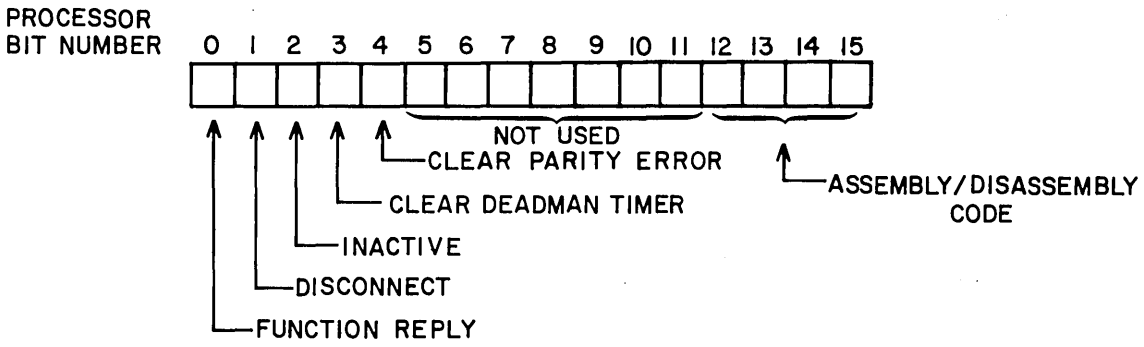
INPREQ2 (Bit 6): A 1 in this bit position indicates that the coupler is receiving an input request siganl from the processor or a CIRPLY (input data reply) from the control logic.

Normal Output Channel Data (O07-00 to O07-15)

These 16 lines provide a multiplexed path for up to 64 control bits. The channel select signals determine the definition of the information on the lines.

Not Set Output Channel 0 (SET O - 00)

The bit definitions of O07-00 to O07-15 when this signal is a 0 are listed below.



Function Reply BIT-0

The processor responds to a function by setting and then clearing this bit.

Disconnect BIT-1

The processor can disconnect an existing PPU/coupler connection by setting and then clearing this bit. This replaces the function reply and serves as both a reply and disconnect.

Inactive BIT-2

The processor can disconnect the PPU and terminate any operation in progress by setting and then clearing this bit.

Clear Deadman Timer BIT-3

By setting this bit the processor clears the deadman timer. If the bit remains set, the deadman timer is disabled. If the bit is set and then cleared, the deadman timer begins to time-out again.

Clear Parity Error (FA722-A/B only) BIT-4

By setting and then clearing this bit, the processor clears bit 4 (parity error) of normal input channel 0. This clears the parity error condition on the coupler/PPU channel that was detected by the coupler. (Note: This bit is used in a CYBER 170 configuration only.)

Assembly/Disassembly Format Code

The processor selects a coupler A/D format by sending a code in bits 12, 13, 14, and 15. The codes are listed on the following page and discussed at the end of this section.

Format	Bit 12	13	14	15
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Not Set Output Channel 4 (SET O - 04)

A 0 on this line allows the processor to select the coupler data path by sending a code in O07-14 and O07-15. The codes are listed below.

Processor Bit Number

14	15	Path Selected
0	0	Processor to PPU
0	1	Controller to PPU
1	0	Processor to Controller
1	1	Local Autoload to Processor

NOT USED PRESENTLY

NOTE

A master clear or autoload function will force the processor to PPU path.

STATION CONTROL INTERFACE

The following signals precede an autoload operation only. All signals go from the coupler to the processor.

Not Cycle Stop ($\overline{\text{CYCLES}}$)

The coupler sends this pulse to the processor to halt execution of the processor program at the end of the current memory cycle (regardless of its relationship to the end of the current instruction).

Not Master Clear ($\overline{\text{MC}}$)

The coupler sends this pulse to clear the processor and its memory.

Enable Output ($\overline{\text{ENOUT}}$)

The coupler holds this signal to a 0 level to enable the block transfer input instruction during an autoload.

Not Force Function ($\overline{\text{FORCEF}}$)

This pulse, in conjunction with enable output, forces the processor to initialize a block transfer input instruction.

Not Go ($\overline{\text{GO}}$)

This pulse starts execution of the block transfer input instruction.

CONTROL LOGIC INTERFACE

SIGNALS TO THE CONTROL LOGIC DATA INTERFACE

Input Data Request (CIREQ)

The coupler sends this signal to the control logic to indicate that the coupler has data available to send to the control logic. This signal clears when the coupler receives an input data reply from the control logic.

Output Data Reply (CORPLY)

The coupler sends this signal to the coupler to indicate that it has accepted a data byte from the control logic. Output data request clears this signal.

End of Record Input (EORIN)

The coupler sends this pulse to indicate that it has sent the last word of a data record to the control logic. It can also indicate that the PPU is terminating a transfer from the control logic.

Data (CIB-00 to CIB-15)

These 16 lines carry data from the coupler to the control logic.

Halt

This pulse indicates that the coupler has received a master clear from the channel, detected a deadman terminate, or decoded an autoload function. This pulse clears the control logic.

SIGNALS FROM THE CONTROL LOGIC DATA INTERFACE

Input Data Reply (CIRPLY)

This pulse indicates that the control logic accepted a data byte.

Output Data Request (COREQ)

This signal indicates that the control logic has data available for the coupler. This signal must remain a 1 until the leading edge of CORPLY.

End of Record Output (EOROUT)

This pulse indicates that the control logic has sent the last word of a record.

Not Write ($\overline{\text{WRITE}}$)

A transition from a 1 to a 0 on this line indicates that the control logic has initiated an input operation.

Data (COB-00 to COB-15)

These 16 lines carry data from the control logic to the coupler. Data must remain stable from the leading edge of COREQ to the leading edge of CORPLY.

FUNCTION /RESERVE/CONNECT

The coupler checks each function code sent to it for the equipment number and legal function. The equipment number of the coupler is hardwired to 0. Bits 9, 10, and 11 of the function code contain the equipment number so these must all be 0's. A legal coupler function is one from 0000_8 to 0477_8. A legal function with the correct equipment number reserves the coupler (if it was not previously reserved). The coupler remains reserved until the PPU issues an inactive signal, the processor disconnects the coupler, the dead-man time expires, or the coupler receives a master clear.

NOTE

The PPU sends an operation complete function to the processor which causes it to disconnect the coupler at the end of an operation.

A dual access coupler must be connected to one of the PPUs before the PPU can initiate a data fransfer. A legal function sent to an unreserved and unconnected coupler automatically connects and reserves the coupler. The connect condition prevents the other PPU from performing any operation with the coupler except checking status. A scanning connect switch in the dual coupler determines which PPU will be connected if both PPUs attempt to send function codes to the coupler simultaneously.

If a PPU is connected to the coupler and the other PPU attempts to send function codes to the coupler, the following action takes place.

1. The coupler checks the function codes for a status request code (0012_8). If it is a status request, the coupler returns an inactive signal to the PPU that sent the request.
2. The PPU activates the channel.
3. The coupler returns a full signal and sets bit 2^{10} on the data lines. The other bits are ignored. Bit 2^{10} indicates that the coupler is connected.
4. The PPU sends an empty to the coupler in response to the full.
5. The coupler sends an inactive signal to the PPU, disconnecting the channel.

If the function does is not a status request, the coupler does not respond and the PPU must wait. The coupler retains the function code and processes it when the other PPU becomes disconnected. The PPU may clear this function by sending an inactive signal or a master clear.

The coupler responds to two functions:

1. Autoload (0414_8)
2. Status request (0012_8)

DEADMAN TIMER

A PPU/coupler connect condition activates the deadman timer. The timer expires if the coupler does not receive a reply from the processor or controller within approximately 4 seconds. When the timer expires, the coupler clears the connect, sends an inactive to the PPU, and sends a terminate to the processor or a halt to the controller. It also sets bit 2 of normal input channel 0. If the processor sets bit 3 of normal output channel 0, the deadman timer is cleared. If bit 3 remains set, the timer is disabled; if it set and then cleared, the timer starts to time-out again.

The coupler responds to the status request only if it is a dual access coupler that is connected, and the unconnected PPU is making the request. The coupler transfers all other functions (and status request if the preceding condition is not true) to the processor on normal input channel 3. The function code is placed in the nine lower order bit positions (2^0-2^8) of NIC 3.

The coupler automatically establishes format and data path during a function. The function code is available to the processor on normal input channel 3. The coupler sets bits 0 of normal input channel 0 to indicate that the function is available to the processor. The processor must monitor this bit to detect a function. It replies by setting and then clearing bit 0 of normal output channel 0. If the processor does not respond to the function, the channel will hang up until the deadman timer expires. When the coupler receives the reply to the function, it responds to the channel by returning an inactive signal.

STATUS

The PPU can not check coupler status except to check bit 2^{10} to determine whether the coupler is connected to the other PPU. To sample coupler status, the PPU must prearrange a status transfer from the processor. The coupler will treat this like a data transfer from the processor to the PPU.

The processor can check status by reading the information on the normal input channels. These bit definitions are listed in a preceding area of this section. The general types of status are:

Normal input channel 0 - Miscellaneous coupler status

Normal input channel 1 - Controller status

Normal input channel 2 - PPU data

Normal input channel 3 - PPU function codes (bits 2^0 to 2^8)
coupler status (bits 2^9 to 2^{15})

AUTOLOAD

The processor memory can be loaded through the coupler from the PPU data channel.

Function code 0414_8 initiates the autoload from the PPU. Autoload causes the following sequence to occur.

1. The coupler forces a coupler to processor data path. Any path selected prior to the autoload sequence will be cleared. The autoload function sends a halt pulse to the control logic.

2. Assembly format 0000 will be automatically selected. Any format selected prior to the autoload will be cleared.

3. The autoload sequence forces the coupler to send the $\overline{\text{Cycle}}$, $\overline{\text{Force Function}}$, $\overline{\text{Go}}$, and $\overline{\text{Master Clear}}$ signals to the processor. This prepares the processor to receive autoload data.

4. The coupler sends an inactive signal to the PPU. Inactive is a response to the autoload function, indicating that the coupler is ready to receive the autoload data. The autoload data will be transferred to the coupler in exactly the same way as a normal PPU data output.

5. Autoload data transfer terminates when the PPU sends an inactive signal to the coupler, or when deadman timer expires.

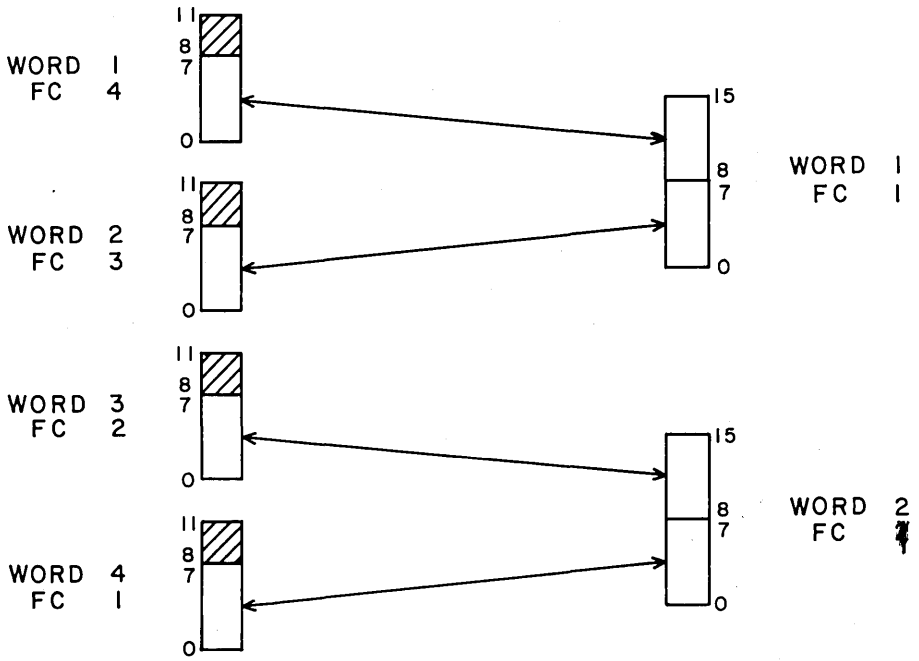
6. The coupler sends a terminate to the processor following either of the conditions listed in step 5. This causes the processor to start execution at address 0001 of its program.

FORMAT AND FRAME COUNT

During data transfers between the PPU and the disk controller, the coupler must assemble 12-bit bytes into 16-bit bytes and disassemble 16-bit bytes into 12-bit bytes. Five assembly and five disassembly formats are available in the coupler. These 10 formats are described and illustrated below. All even formats are used for assembly (12- to 16-bit transfers). All odd formats are used for disassembly (16- to 12-bit transfers).

FORMATS 0000/0001

In format 0000 the coupler assembles two 12-bit bytes into one 16-bit byte. The 4 upper order bits (8 through 11) of each 12-bit byte are disregarded. In format 0001, the coupler disassembles one 16-bit byte into two 12-bit bytes. The 4 upper order bits (8 through 11) of each 12-bit byte are filled with zeros. See Figure 9-1.



AUTO LOAD

Figure 9-1. Formats 0000 (Assembly), 0001 (Disassembly)

FORMATS 0010/0011

In format 0010, the coupler assembles four 12-bit bytes into three 16-bit bytes. In format 0011, the coupler disassembles three 16-bit characters into four 12-bit bytes. See Figure 9-2.

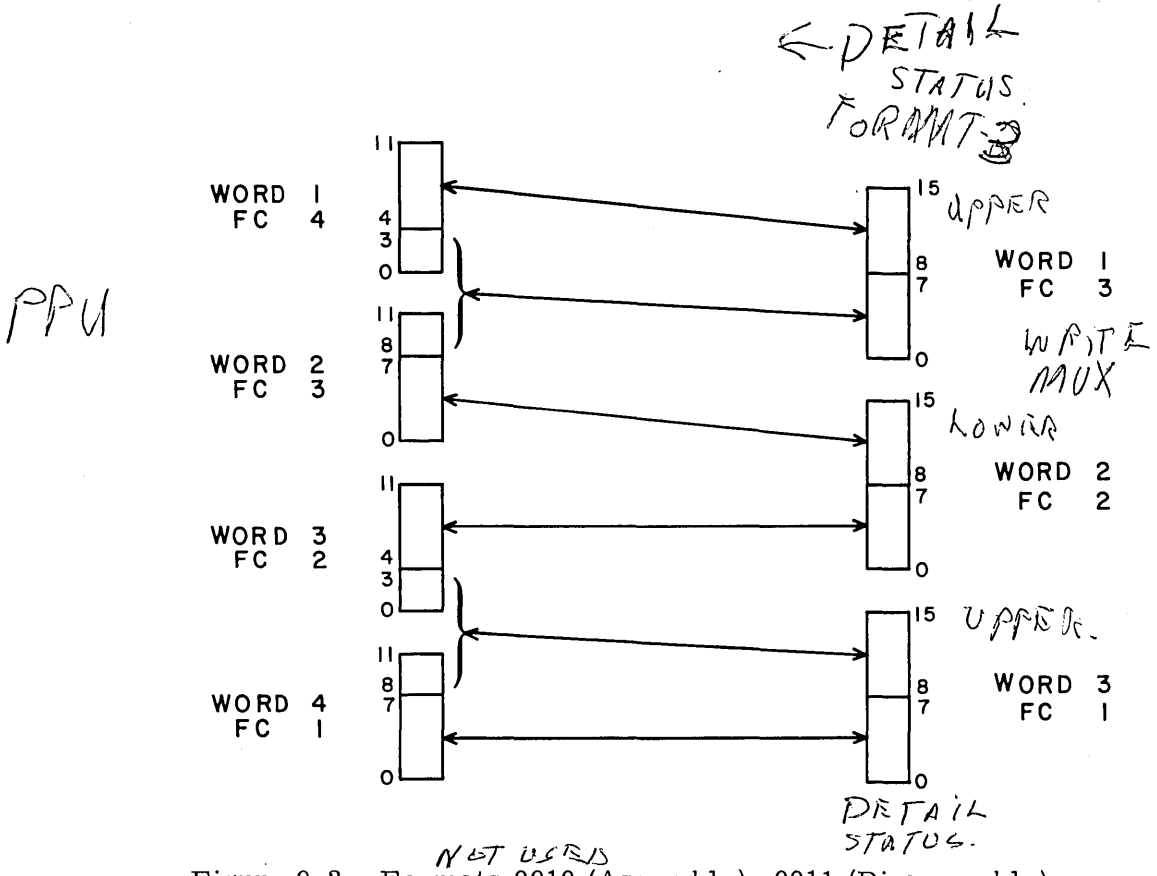


Figure 9-2. Formats 0010 (Assembly), 0011 (Disassembly)

FORMATS 0100/0101

In format 0100 the coupler assembles one 12-bit byte into one 16-bit byte. Bits 6, 7, 14, and 15 of the 16-bit byte are set to zeros. In format 0101 the coupler disassembles one 16-bit byte into one 12-bit byte. Bits 6, 7, 14, and 15 of the 16-bit byte are disregarded. See Figure 9-3.

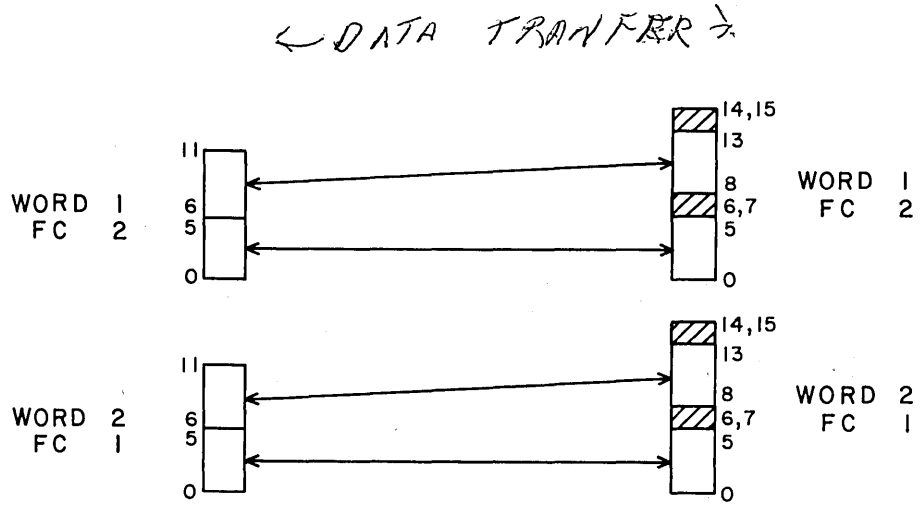


Figure 9-3. Formats 0100 (Assembly), 0101 (Disassembly)

FORMATS 0110/0111

In format 0110, the coupler assembles one 12-bit byte into one 16-bit byte. The 4 upper order bits (12 through 15) of the 16-bit byte are set to zeros. In format 0111 the coupler disassembles one 16-bit byte into one 12-bit byte. The 4 upper order bits (12 through 15) of the 16-bit byte are disregarded. See Figure 9-4.

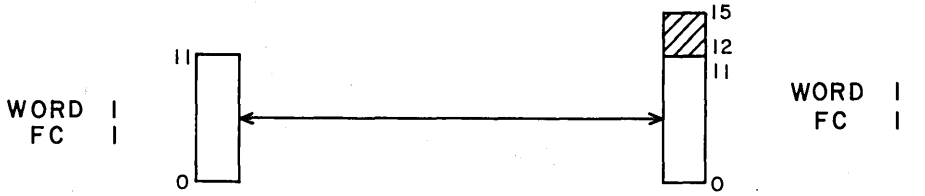


Figure 9-4. Formats 0110 (Assembly), 0111 (Disassembly)

FORMATS 1000/1001

In format 1000, the coupler assembles ten 12-bit bytes into seven 16-bit bytes. The 4 upper order bits (8 through 11) of the first and sixth 12-bit bytes are disregarded. In format 1001, the coupler disassembles seven 16-bit bytes into ten 12-bit bytes. The 4 upper order bits (8 through 11) of the first and sixth 12-bit bytes are set to zeros. See Figure 9-5.

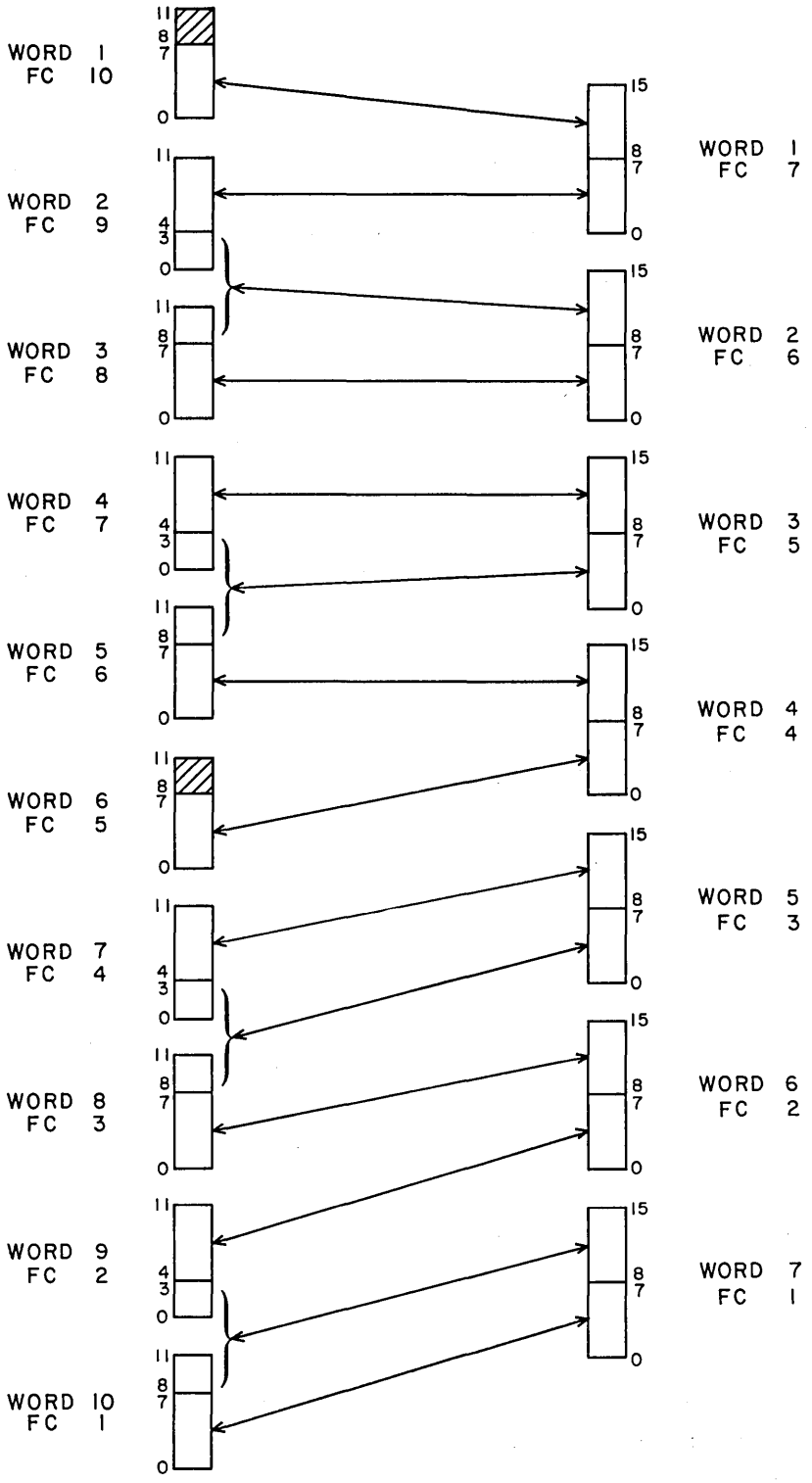


Figure 9-5. Formats 1000 (Assembly), 1001 (Disassembly)

VALID DATA BITS

Table 9-1 lists the valid data bits by format and frame count.

TABLE 9-1. VALID DATA BITS

Format and Frame Count	Valid Data Bits
0000 FC1	*
0001 FC4	0 - 7
0001 FC3	0 - 7
0001 FC2	0 - 7
0001 FC1	0 - 7
0002 FC3	0 - 15
0002 FC2	4 - 15
0002 FC1	8 - 15
0003 FC4	0 - 11
0003 FC3	Not possible
0003 FC2	8 - 11
0003 FC1	4 - 11
0004 FC2	0 - 5, 8 - 13
0004 FC1	0 - 5, 8 - 13
0005 FC2	0 - 11
0005 FC1	0 - 11
0006 FC1	0 - 11
0007 FC1	0 - 11
0008 FC7	0 - 15
0008 FC6	8 - 15
0008 FC5	0 - 15
0008 FC4	8 - 15
0008 FC3	0 - 15
0008 FC2	8 - 15
0008 FC1	8 - 15
0009 FC10	0 - 11
0009 FC9	Not possible
0009 FC8	4 - 11

* The initial word count must be known in this case.

TABLE 9-1. VALID DATA BITS (Cont'd)

Format and Frame Count	Valid Data Bits
0009 FC7	0 - 11
0009 FC6	Not possible
0009 FC5	8 - 11
0009 FC4	0 - 7
0009 FC3	Not possible
0009 FC2	8 - 11
0009 FC1	4 - 11

SECTION 10

7000 COUPLER PROGRAMMING

7000 COUPLER PROGRAMMING

The 7000 coupler contains four interfaces:

- 7000 Channel Interface
- Processor Interface
- Control Logic Interface

Signal definitions and interface characteristics are described in the following paragraphs. All signals are valid at a 1 logic level unless otherwise noted.

7000 CHANNEL INTERFACE

Each 7000 PPU communicates with the coupler through two bidirectional channels. One is dedicated to the transfer of data in both directions, and the other is used to send function codes from the PPU and status information to the PPU.

Write refers to transfers from the PPU to the subsystem; read refers to transfers from the subsystem to the PPU.

Each channel has three control signals. These signals are defined in the following paragraphs.

1. Write Data Word Pulse

   The PPU sends this pulse with each data byte to notify the coupler that a byte is available.

2. Write Data Resume

   The coupler sends this pulse to the PPU to indicate that it has accepted the data byte. Deadman time out forces this signal to a static 1 until the PPU sends a function record pulse. The subsystem processor can force this to a static 1 by setting bit 2^5 on normal ouput channel 0.

3. Write Data Record Pulse

   The PPU sends this pulse to the coupler to terminate either a PPU write or read operation. To terminate a write, this signal must be sent after the resume for the last data word. To terminate a read, it must be sent when the input word flag is set.

4. Read Data Word Pulse

   The coupler sends this pulse to the PPU to indicate that a data byte is available for the PPU.

5. Read Data Resume

   The PPU sends this signal to the coupler to indicate that the PPU accepted the data byte.

6. Read Data Record Pulse

   The coupler sends this pulse to the PPU to indicate termination of the transfer. The coupler sends this at the end of every record, and whenever the deadman timer expires. The processor can generate this pulse by setting bit 2^5 of normal output channel 0.

7. Function Word Pulse

   The PPU sends this pulse to the coupler to indicate that a function code is available on the function channel data lines.

8. Function Resume

   The coupler sends this signal to the PPU under the following conditons.

   a. The processor replies to a function code by setting bit 2^{15} of normal output channel 0.

   b. An automatic response by the coupler if the other PPU has reserved the coupler.

   c. The coupler receives a function record pulse.

   d. The deadman timer expires.

9. Function Record Pulse

   The PPU sends this signal to the coupler to:

   a. Clear the coupler reservation (of the PPU which sent the pulse only).

   b. Clear the static write data resume following a deadman timeout.

   c. Generate a function resume and write data resume. (This initializes PPU channel flags.)

10. Status Word Pulse

    The coupler sends this pulse to the PPU to indicate that status is available on the status data lines. The coupler generates this pulse when the processor sends status on normal output channel 5.

11. Status Resume

    The PPU sends this signal to the coupler to indicate that it has accepted the status byte.

12. Status Record Pulse

    The coupler sends this pulse to the PPU to indicate that the deadman timer has expired.

PROCESSOR INTERFACE

The coupler exchanges data and control signals with the processor block transfer interface, normal channel interface, and station control interface. The block transfer interchange exchanges all I/O control signals and contains the normal data path between the coupler and the processor. The normal channel interface receives coupler status, control logic status, and sends out the path select, and format select. The coupler sends signals to the station control interface during autoload operations only.

SIGNALS FROM THE COUPLER TO THE BLOCK TRANSFER INTERFACE

1. Coupler Ready (CREADY)

   This signal indicates that the coupler will respond immediately to an input block transfer output block transfer instruction in the processor.

2. Coupler Reply (CREPLY)

   This pulse indicates that the coupler has accepted a data byte from the processor or that the coupler has a data byte available for the processor. The coupler responds to an input request or output ready with this signal.

3. Coupler Terminate (CCTERM)

   Following an input request or an output ready, the coupler can terminate an input block transfer or output block transfer instruction by sending this signal. The processor will not send a parity strobe.

4. Data (CCI-00 to CCI-15)

   These 16 lines carry data from the coupler to the processor.

SIGNALS TO THE COUPLER FROM THE BLOCK TRANSFER INTERFACE

1. Output Ready (OUTRDY)

   This 35 to 65 NS pulse indicates that the processor has data available for the coupler.

2. Input Request (INPREQ)

   This 35 to 65 NS pulse indicates that the processor can receive another byte of data from the coupler.

3. Parity Strobe (CK-PAR)

   This 35 to 65 NS pulse indicates that the processor has accepted a data byte from the coupler.

4. Master Clear (MC-CC)

   This signal clears registers and control in the coupler.

5. End of Operation (EOP-CC)

   This pulse accompanies the last output ready signal during a block transfer to indicate that the transfer is complete. It will also be returned to the coupler following a terminate signal from the coupler, or following the reply to the last input request during a block transfer.

6. Data (CC0-00 to CC0-15)

   These lines carry data from the processor to the coupler.

NORMAL CHANNEL INTERFACE

1. Normal Input Channel Data (I03-00 to I03-15)

   These 16 lines provide a multiplexed data path for up to 64 status bits. The channel select signals determine which status byte will be sent to the processor.

2. Not Select Input Channel Zero $\overline{\text{(SEL-IO)}}$

   A logical 0 on this line gates coupler status through the multiplexer to the processor on the input channel data lines. Bit definitions are given in the following paragraphs.

Processor Normal Input Channel Zero

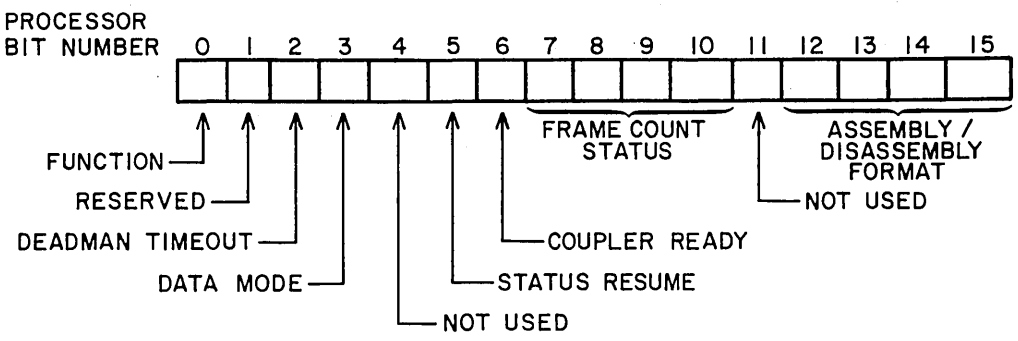


Figure 10-1. NIC 0 Bit Designations

a. Function

A logical 1 in this bit indicates that a function code is available on normal input channel 2.

b. Reserved

A logical 1 in this bit indicates that the coupler is in an operational mode (not idling). This bit sets when the coupler processes a function code from the PPU and clears upon receipt of a function record pulse or when DMTO expires.

c. Deadman Timeout

A logical 1 in this bit indicates that the deadman timer has expired and the coupler reserve has been cleared.

d. Status Resume

This bit changes state each time the PPU does a status input. The bit is not affected by master clear or any other signal.

e. Data Mode

A logical 1 in this bit indicates that the coupler is reserved and a data transfer is in process.

f. Coupler Ready

A logical 1 in this bit indicates that the coupler is ready to transfer data to/from the processor or control logic. This bit is always a 1 if the data path between the control logic and processor is selected.

g. Frame Count

Bits 7, 8, 9, and 10 carry a code to indicate frame count in the coupler. Frame count codes are as follows.

Count	Bit 7	8	9	10
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0

Table 10-1 lists the valid data bits by format and frame count.

3. Not Select Input Channel Two ($\overline{\text{SELI-2}}$)

A logical 0 on this line gates 16 coupler data bits from the PPU to the processor as status. This allows the processor to monitor data sent from the PPU to the control logic. The processor receives functions on this channel also.

TABLE 10-1. VALID DATA BITS

Format and Frame Count	Valid Data Bits
0000 FC1	*
0001 FC4	0 - 7
0001 FC3	0 - 7
0001 FC2	0 - 7
0001 FC1	0 - 7
0002 FC3	0 - 15
0002 FC2	4 - 15
0002 FC1	8 - 15
0003 FC4	0 - 11
0003 FC3	Not possible
0003 FC2	8 - 11
0003 FC1	4 - 11
0004 FC2	0 - 5, 8 - 13
0004 FC1	0 - 5, 8 - 13
0005 FC2	0 - 11
0005 FC1	0 - 11
0006 FC1	0 - 11
0007 FC1	0 - 11
0008 FC7	0 - 15
0008 FC6	8 - 15
0008 FC5	0 - 15
0008 FC4	8 - 15
0008 FC3	0 - 15
0008 FC2	8 - 15
0008 FC1	8 - 15
0009 FC10	0 - 11
0009 FC9	Not possible
0009 FC8	4 - 11
0009 FC7	0 - 11
0009 FC6	Not possible
0009 FC5	8 - 11
0009 FC4	0 - 7
0009 FC3	Not possible
0009 FC2	8 - 11
0009 FC1	4 - 11

* The initial word count must be known in this case.

4. Normal Output Channel Data (O07-00 to O07-15)

These 16 lines provide a multiplexed path on which the channel select signals determine the definition of the information on the lines.

NOTE

Because of the reconfiguration of the normal channel interface, the clear channel bit and set channel bit instructions apply to normal output channel 0 only. Individual channel bits on normal output channels 1, 2 4, and 5 should be set and cleared by doing an output from the A register.

5. Not Set Output Channel Zero ($\overline{\text{SET 0 -00}}$)

The bit definitions of O07-00 to O07-15 when this signal is logical 0 are given in the following paragraphs.

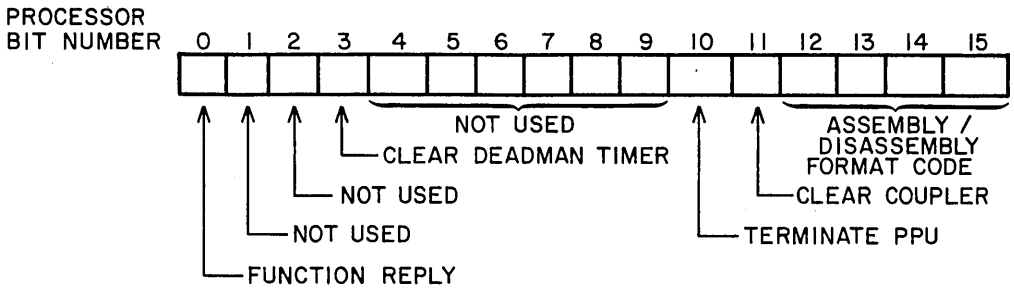


Figure 10-2. NOC 0 Bit Designations

a. Function Reply

The processor responds to a PPU function by setting and then clearing this bit.

b. Clear Deadman Timer

By setting this bit the processor clears the deadman timer. If the bit remains set, the deadman timer is disabled. If the bit is set and then cleared, the deadman timer begins to time out again.

c. Terminate PPU

Setting this bit when a write format is selected causes the coupler to send a static write data resume signal to the reserved PPU. The resume remains static until this bit clears.

Setting this bit when a read format is selected causes the coupler to send a read data record pulse to the reserved PPU.

d. Clear Coupler

Setting this bit clears all control logic in the coupler except PPU reserve.

e. Assembly/Disassembly Format Code

The processor selects a coupler A/D format by sending a code in bits 12, 13, 14, and 15. The codes are as follows:

Format	Bits 12	13	14	15
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

6. Not Set Output Channel Four $\overline{\text{(SET O-04)}}$

A logical 0 on this line allows the processor to select the coupler data by sending a code in O07-14 and O07-15. The codes are as follows:

Processor Bit Number	14	15	Path Selected
	0	0	Processor to PPU
	0	1	Control logic to PPU
	1	0	Processor to control logic
	1	1	Not Used

NOTE

A master clear or autoload function will force the processor/PPU path.

7. Not Set Output Channel Five $\overline{\text{(SETO-05)}}$

A logical 0 on this line gates normal channel bits O07-15 to O07-04 (2^0 to 2^{11}) into the status register in the coupler. This causes the coupler to send a word pulse to the PPU status channel. The PPU can then read these 12 status bits.

STATION CONTROL INTERFACE

The following signals precede an autoload operation only. All signals go from the coupler to the processor.

1. Not Cycle Stop $\overline{(CYCLES)}$

   The coupler sends this pulse to the processor to halt execution of the processor program at the end of the current memory cycle (regardless of its relationship to the end of the current instruction).

2. Not Master Clear $\overline{(MC)}$

   The coupler sends this pulse (8 microseconds minimum) to clear the processor and its memory.

3. Enable Output $\overline{(EN\text{-}OUT)}$

   The coupler holds this signal to a logical 0 level to enable the block transfer input instruction during an autoload.

4. Not Force Function $\overline{(FORCEF)}$

   This pulse in conjunction with enable output, forces the processor to initialize a block transfer input instruction.

5. Not Go $\overline{(GO)}$

   This pulse starts execution of the block transfer input instruction.

CONTROL LOGIC INTERFACE

1. Input Data Request (CIREQ)

   The coupler sends this signal to the control logic to indicate that the coupler has data available to send to the control logic. This signal clears when the coupler receives an input data reply from the control logic.

2. Output Data Reply (CORPLY)

   The coupler sends this signal to the control logic to indicate that it has accepted a data byte from the control logic. Output data request clears this signal.

3. End of Record Input (EORIN)

   The coupler sends this pulse to indicate that it has sent the last word of a data record to the control logic.

4. Data (CIB-00 to CIB-15)

   These 16 lines carry data from the coupler to the control logic.

5. Halt

   This pulse indicates that the coupler has detected a deadman terminate, or decoded an autoload function.

6. Not Master Clear ($\overline{\text{MC-CC}}$)

   The coupler sends this signal to the control logic when it has received a master clear from the processor.

SIGNALS FROM THE CONTROL LOGIC DATA INTERFACE

1. Input Data Reply (CIRPLY)

   This pulse indicates that the control logic has accepted a data byte.

2. Output Data Request (COREQ)

   This signal indicates that the control logic has data available for the coupler. This signal must remain a logical 1 until the leading edge of CORPLY.

3. End of Record Output (EOROUT)

   This pulse indicates that the control logic has sent the last word of a record.

4. Not Write ($\overline{\text{WRITE}}$)

   A transition from a logical 1 to a logical 0 on this line indicates that the control logic has initiated an input operation.

5. Data (COB-00 to COB-15)

   These 16 lines carry data from the control logic to the coupler. Data must remain stable from the leading edge of COREQ to the leading edge or CORPLY.

COUPLER RESERVE

The coupler can be accessed by two PPUs, designated odd and even accesses. A function code sent by either PPU reserves the coupler unless it is already reserved by the other PPU. If the coupler receives function codes from both PPUs simultaneously, a hardware scanner in the coupler determines which PPU reserves the coupler. No priority is given to either PPU.

The PPU which has reserved the coupler can clear the reservation by sending a function record pulse. The other PPU cannot clear the reservation. The processor cannot clear the reservation, but expiration of the deadman timer will.

If the coupler receives a function code from the even access when it is reserved by the odd (or vice versa), the even access will not reserve the coupler, the odd access remains reserved, and the coupler sends a function resume to the even access. Also, the coupler sends a status word pulse and a status byte to the even access. The status word indicates that the function attempt was unsuccessful. It remains on the lines until the PPU reads the status and sends a status resume to the coupler.

If the coupler is a single access version (connected to one PPU), the reserve logic operates exactly the same way.

FUNCTION

The PPU initiates all data transfers by sending a function code to the coupler on the function channel. If this reserves the coupler (or the coupler was reserved by this access before receiving the function), the coupler takes the following action.

1. Determines if it is an autoload code (0414). If it is, an autoload sequence begins.
2. Automatically selects disassembly format 0110.
3. Gates the code to the normal input channel interface.
4. Sets bit 2^{15} of normal input channel 0. The processor must scan this bit to determine if a function code is available from the coupler. The processor must then do an input of normal input channel 2 to receive the function code. When the processor has accepted the code, it replies by setting bit 2^{15} on normal ouput channel 0. If the processor does not reply, the deadman timer expires in approximately 4 seconds.
5. When the coupler receives the reply, it sends a function resume to the PPU.
6. The assembly/disassembly format will revert to its previous selection after the coupler has sent the function resume.

STATUS

The coupler sends status to the PPU and makes status available to the processor. Two different status bytes can be sent to the PPU. Three different status bytes are available to the processor.

STATUS TO THE PPU (The following status bytes are sent to the PPU.)

1. Processor Status

   This is available when the processor does an output on normal ouput channel 5. This causes the coupler to send a status word pulse to the reserved PPU. When the PPU reads the status, it sends a status resume to the coupler. This toggles bit 2^{10} on normal input channel 0. The processor may use the transition of this bit to determine when the PPU accepted the status.

   NOTE

   Normal ouput channel 5 is not bit-alterable. Status may be placed on this channel by using processor instruction OF5X if the status to be sent consists of more than one bit. If only one bit of status must be sent, processor instruction OA5X may be used. All other bits are filled with zeros.

2. Coupler (Busy) Status

   The coupler returns this status byte to the PPU on access A when the coupler is reserved by the PPU on access B, and access A attempts to function the coupler (or vice versa). When this happens the coupler sends a status word pulse to the PPU and puts the following status byte on the lines. (This also occurs when deadman timeout expires.)

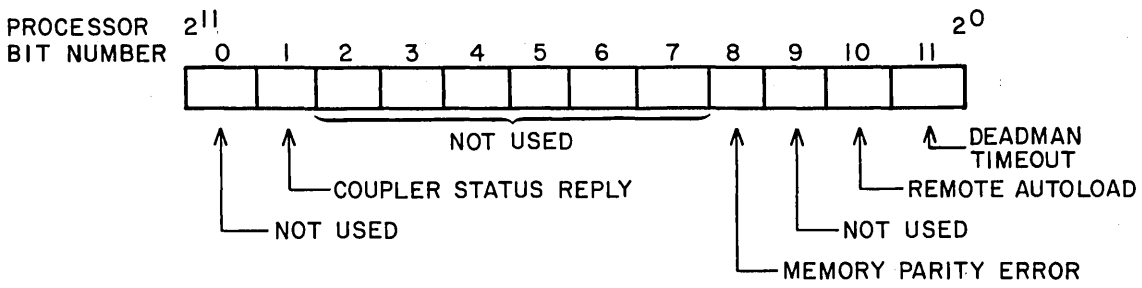


Figure 10-3 Coupler (Busy) Status Bits

a. Coupler Status Reply

A 1 in this bit indicates that the status did not originate in the processor.

b. Memory Parity Error

A 1 in this bit indicates that the processor detected a parity error.

c. Remote Autoload

A 1 in this bit indicates that an autoload operation from the other PPU is in process.

d. Deadman Timeout

A 1 in this bit indicates that the deadman timer has expired and the processor has not cleared it.

STATUS TO THE PROCESSOR

Status is available to the processor on the following normal input channels.

Channel	Status
0	Coupler status (see normal channel interface definitions)
1	Control logic status
2	Coupler data (to the control logic from the PPU). (Also, the processor receives the function data on this channel.)

DATA TRANSFERS

All data transfers are defined relative to the PPU. A write operation indicates that PPU is sending data to the coupler. A read operation indicates that the coupler is sending data to the PPU.

Before a data transfer through the coupler, the PPU must reserve the coupler. Next, the processor must select the assembly/disassembly format and data path.

The processor selects the format by sending a code in bits 12-15 (2^0 to 2^3) of normal output channel 0. It selects the path by sending a code in bits 14 and 15 (2^0 and 2^1) of normal ouput channel 4. The path and corresponding code are as follows:

NOC 4 2^1	2^0	Path
0	0	Processor/PPU
0	1	Control logic/PPU
1	0	Processor/control logic
1	1	Not Used

COUPLER DATA BUFFER CAPACITY

The coupler contains two 16-bit data registers. The effective buffer area is two 16-bit words except assembly/disassembly formats 0, 6, and 7 which have one 16-bit word of buffering.

AUTOLOAD

Processor memory can be autoloaded through the coupler from the PPU data channel. Function code 0414_8 initiates the autoload from the PPU. Autoload causes the following sequence to occur.

1. The coupler forces a coupler to processor data path. Any path selected prior to the autoload sequence will be cleared.
2. The autoload function sends a halt pulse to the control logic.
3. Assembly format 0000 will be automatically selected. Any format selected prior to the autoload will be cleared.

4. The autoload sequence forces the coupler to send the cycle, force function, go, and master clear signals to the processor. This prepares the processor to receive autoload data.

5. The coupler sends a resume pulse to the PPU. Resume is a response to the autoload function, indicating that the coupler is ready to receive the autoload data. The autoload data will be transferred to the coupler in exactly the same way as a normal PPU data output.

6. Autoload data transfer terminates when the PPU sends a write data record flag to the coupler, or deadman timer expires.

7. The coupler sends a terminate to the processor following one of the two conditions listed in step 6. This causes the processor to start execution at processor program address 0001.

DEADMAN TIMER

A PPU reserve condition activates the deadman timer. The timer expires if the coupler does not receive a write data reply or read data reply within approximately 4 seconds. If the processor set bit 2^{12} of normal output channel 0, the timer is disabled. When the timer expires, the coupler takes the following action.

1. Clears the PPU reservation.
2. Sends a record pulse to the reserved PPU status channel.
3. Sends a terminate to the processor (if the processor is in block transfer mode).
4. Sends a halt to the control logic.
5. Sets bit 2^{13} of normal input channel 0.
6. Terminates the reserved PPU by the following methods.
   a. Sends a record pulse to the read channel.
   b. Sends a resume to the function channel.
   c. Sends a static resume to the write data channel.

To clear the static resume (6-c) so that the coupler can receive another function, either PPU must send a function record pulse. The coupler will return coupler status (indicating that the coupler is busy) to a PPU attempting to function the coupler when the static resume is present.

FORMAT AND FRAME COUNT

During data transfers between the PPU and the processor or control logic, the coupler must assemble 12-bit bytes into 16-bit bytes and disassemble 16-bit bytes into 12-bit bytes. Five assembly and five disassembly formats are available in the coupler. These 10 formats are described and illustrated in section 9, Figures 9-1 through 9-5. All even formats are used for assembly (12- to 16-bit transfers). All odd formats are used for disassembly (16- to 12-bit transfers).

The frame count, in conjunction with the format, determines which step of the assembly/disassembly sequence the coupler is in. The processor sends a code in bits 12-15 of normal output channel 0 to select format. This sets the frame counter in the coupler to a value appropriate to the format. The frame count decrements each time a 12-bit word is sent to the PPU or a 16-bit word is accepted by the control logic or processor. When the count decrements to 0, the counter is reset to the starting count. The counter does not stay at 0 but is reset immediately to the starting count (that is, 0 is not an enabling frame count for assembly/disassembly.

COMMENT SHEET

MANUAL TITLE CDC FA710/FA719/FA720-A/B/C/D, FA722-A/B/C, FA723-A/B Disk Controllers Hardware Reference Manual

PUBLICATION NO. 60364500 REVISION H

FROM: NAME: _____

BUSINESS ADDRESS: _____

COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

AA3419 REV. 11/69 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Publications and Graphics Division
4201 North Lexington Avenue
Arden Hills, Minnesota 55112

CUT ALONG LINE

FOLD

FOLD

CORPORATE HEADQUARTERS, 8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A.