EK-RH20-UD-001

RH20
MASSBUS CONTROLLER
UNIT DESCRIPTION

digital equipment corporation • marlborough, massachusetts

1st Printing,August 1976

Printed in U.S.A.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

CONTENTS

CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

TABLES

PREFACE

The RH20 Unit Description contains three levels of description:

1. Overview
2. Function Description
3. Logic Description

The Overview provides a physical description of the RH20 and describes basic operation. The Functional Description describes the RH20 system interface (EBus, CBus, Massbus), the RH20 instruction set, and the MBox channel commands. Controller and drive registers are discussed and basic programming steps are outlined. The logic description is the most comprehensive part of the RH20 Unit Description. With the aid of flowcharts and timing diagrams, RH20 operation is described in terms of its basic logic elements. Print prefixes are also used, thus providing a direct index into the Field Maintenance Print Set.

SECTION 1
OVERVIEW

1.1 INTRODUCTION

The RH20 Massbus Controller (MBC) provides a high-speed synchronous data transfer interface between any of up to eight Massbus-compatible devices (disk-pack drives, tape drives, etc.) and one of eight data channels located in the system MBox. It also allows asynchronous data transfers to/from device registers. Figure 1-1 shows interconnection of the RH20 in a system.

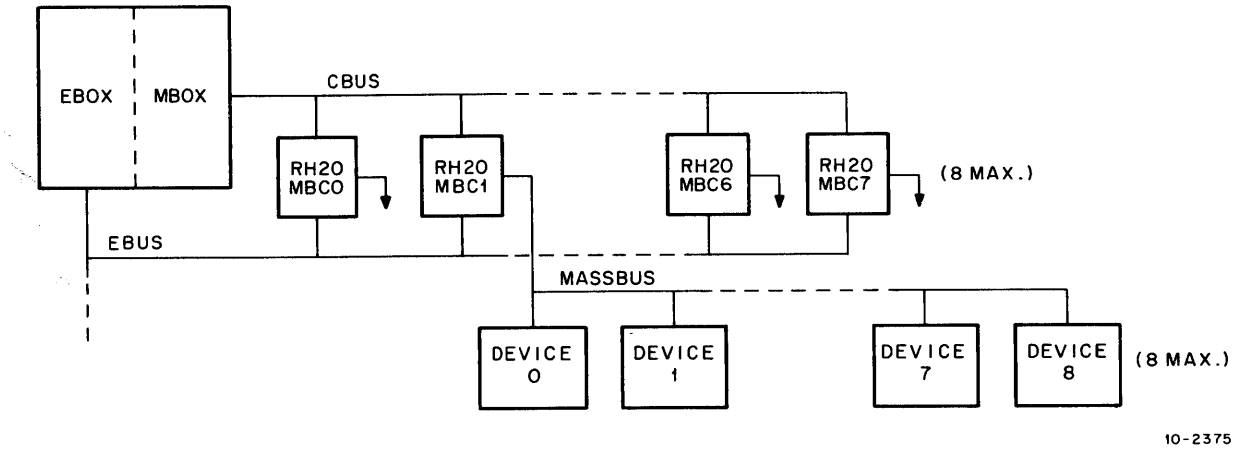


Figure 1-1 RH20 System Interconnection

Data is transferred between the RH20 and the associated data channel over the CBus. Asynchronous control of the RH20 is exercised by the EBox over the EBus. The Massbus connection provides an interface to both single- and dual-ported drives. Up to eight MBCs can be installed in a single processor system and two or more MBCs can transfer data simultaneously over the multiplexed CBus.

1.2 PHYSICAL DESCRIPTION

The RH20 consists of three double-height hexboards (one M8555, one M8556, and one M8557) that are located in the system I/O cabinet. Twenty-four slots are allocated to provide space for eight MBCs. The Massbus cable connector slots for each RH20 are physically located on a (cable connector) PC board beneath each of the corresponding MBC locations (MBC0, MBC1, MBC2, etc.) and are provided with ZIF (zero-insertion-force) connectors which will accept either a BC06S round device cable or a special Massbus terminator. Using a bus terminator instead of a device cable at an MBC cable connector permits diagnostic loop operations to be performed on the Massbus with all devices disconnected. A connector is provided for each MBC, allowing eight separate chains of Massbus devices to be connected. The EBus and CBus cable connections are made through module slots in the backplane and are interconnected with the system by means of cables within the I/O cabinet. RH20 module utilization and Massbus connector locations are shown in Figure 1-2.

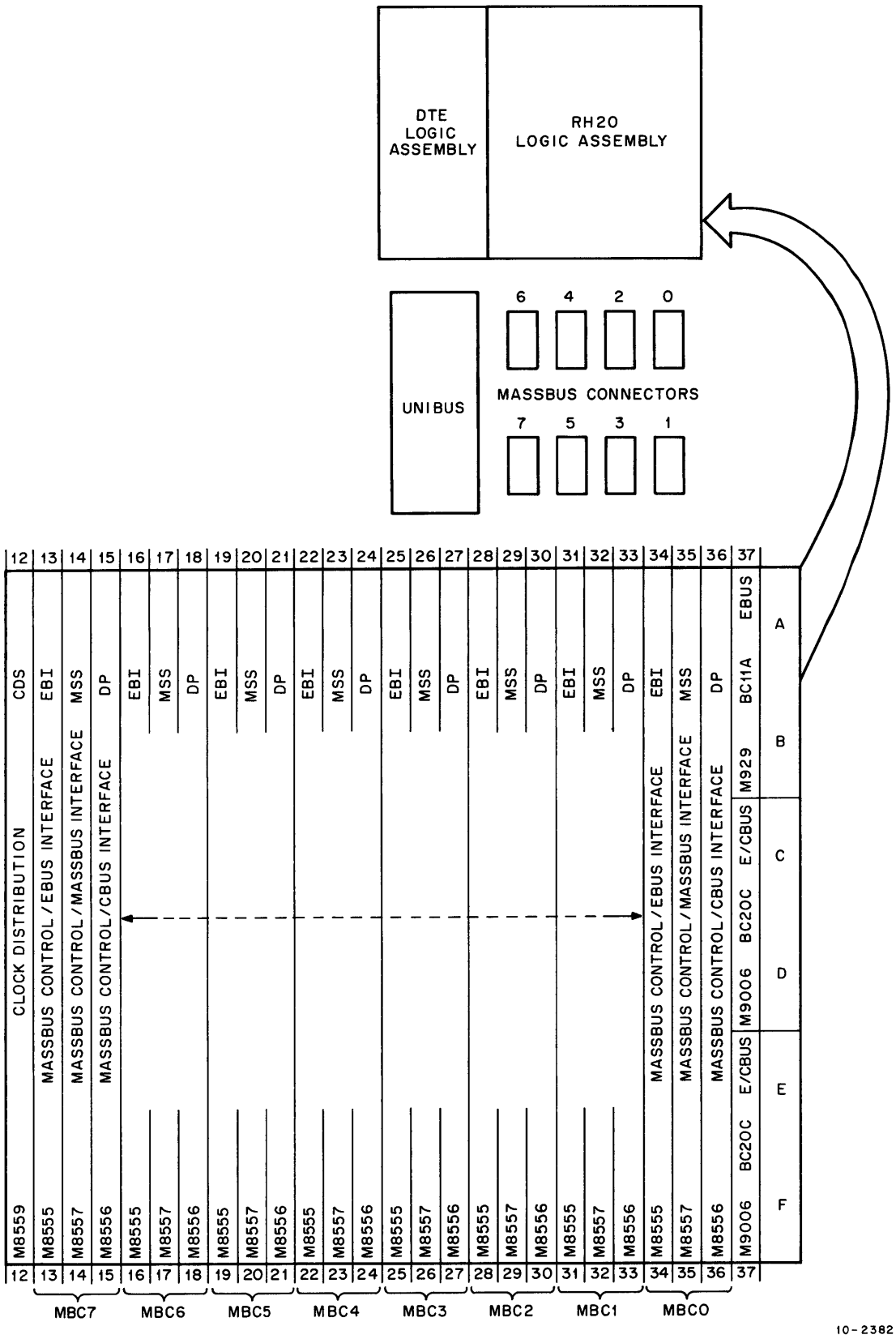


Figure 1-2 RH20 Module Utilization

1.3 OPERATING CHARACTERISTICS

For addressing purposes, each RH20 has been assigned a unique device code and physical number. Both addresses are dependent upon the physical location of the RH20 and are hardwired on the backplane. CONO, DATAO, CONI, and DATAI commands address the RH20 by device code. The physical address is used during execution of the PI ADR IN command. Table 1-1 lists the device codes and physical numbers assigned to each controller.

Although each RH20 controls up to eight drives in parallel on the Massbus, only one drive at a time can transfer data. However, non-data transfer commands (Seek, Rewind, etc.) can be issued when a data transfer is active, provided that the addressed drive is not executing the read/write.

Table 1-1 RH20 Device/Physical Select Codes

Controller	Device Code (Octal)	Symbol	Physical Number
1st	540	MBC0	0
2nd	544	MBC1	1
3rd	550	MBC2	2
4th	554	MBC3	3
5th	560	MBC4	4
6th	564	MBC5	5
7th	570	MBC6	6
8th	574	MBC7	7

SECTION 2
FUNCTIONAL DESCRIPTION

A simplified functional block diagram of the RH20 and a Massbus device is shown in Figure 2-1. The system interface and the addressable registers in both the controller and a typical device are indicated.

2.1 SYSTEM INTERFACE

The RH20 interfaces to the EBus, Massbus, and CBus. A general discussion of bus operation follows. A tabulated signal summary for each bus is provided.

2.1.1 EBus

The EBus interconnects all peripheral device controllers, including the RH20, with the system EBox. EBus commands transfer control data to the RH20 (CONO/DATAO), transfer control status information from the RH20 (CONI/DATAI), and service RH20 interrupts (PI SERVED/PI ADR IN). EBus interface signals and information flow are shown in Figure 2-2. The EBus signals are described in Table 2-1.

During a CONO or DATAO command, the EBox places the device code on the CS lines, the control data on the data lines, and asserts a 0 or 2_8 (CONO/DATAO) on the F lines. The EBox then allows a deskew and decode delay time, and asserts DEMAND. The RH20 decodes the F and CS lines. If the device code matches its hard-wired address when DEMAND is received, the controller asserts ACKN, strobes the control data from the EBus, and executes the command by performing the specified control function (CONO) or by loading the addressed register (DATAO). The addressed register can be in the RH20 (an internal register) or in a Massbus device (an external register). XFER is then asserted by the RH20 to signal that the command has been executed. After receiving XFER, the EBox negates DEMAND. The trailing edge of DEMAND negates ACKN and XFER in the RH20 to terminate bus operation. Timing is shown in Figure 2-3.

During a CONI or DATAI command, the EBox places the device code on the CS lines, asserts a 1_8 or 3_8 on the F lines (CONI/DATAI), and asserts DEMAND after a delay time that allows the RH20 to decode the CS and F lines. If addressed (i.e., CS lines match hard-wired address), and after receiving DEMAND, the controller asserts ACKN and executes the command by placing either the control status information (CONI) or the register data (DATAI) on the EBus data lines. It then asserts XFER. (Both internal and external registers can be read by the DATAI.) When the EBox receives XFER, it strobes the data lines and negates DEMAND. In the RH20, the trailing edge of DEMAND negates ACKN, XFER, and the data lines to end the operation on the EBus. Timing is shown in Figure 2-4.

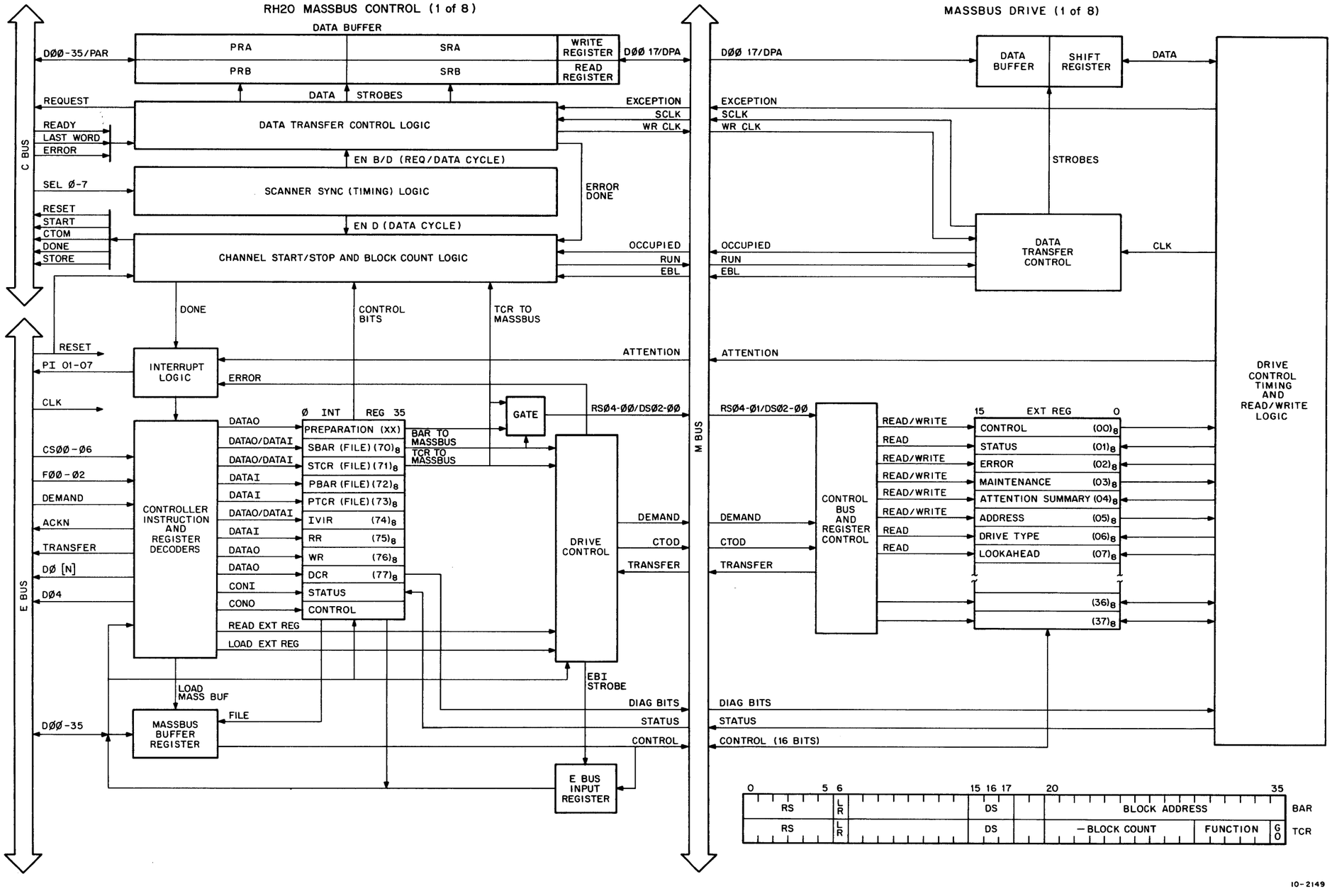


Figure 2-1 RH20/Drive
Functional Block Diagram

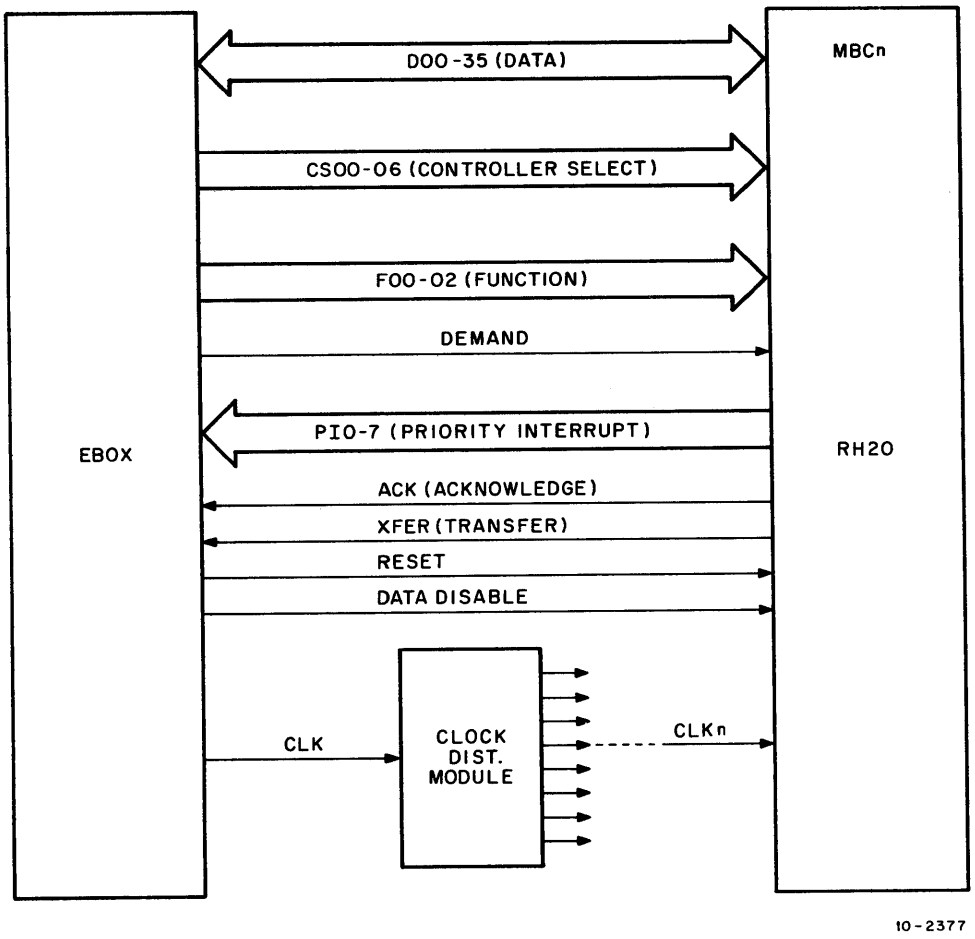


Figure 2-2 EBus Interface

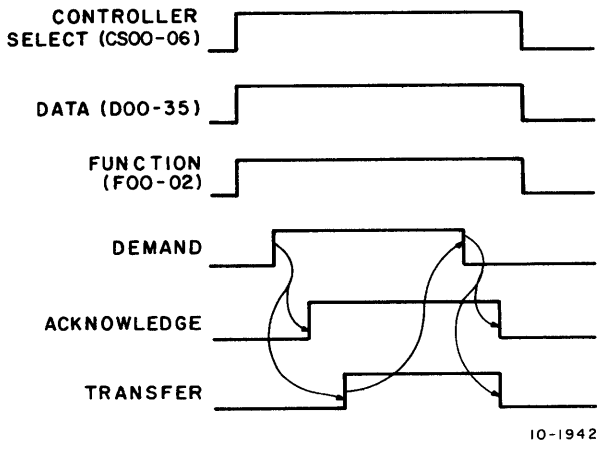


Figure 2-3 CONO/DATAO Operation

Table 2-1 EBus Signal Description

Signal	Direction	Description
Data Lines (D00–35)	Bidirectional	Transfer control data and status information between the EBox and the RH20.
Function Lines (F00–02)	EBox to MBC	Specify the EBus command to be executed. F00 F01 F02 Command 0 0 0 CONO 0 0 1 CONI 0 1 0 DATAO 0 1 1 DATAI 1 0 0 PI SERVED 1 0 1 PI ADR IN
Controller Select Lines (CS00–06)	EBox to MBC	CS00–06 select the RH20 by device code (Table 1-1) during a CONO, DATAO, CONI, or DATAI. CS04–06 specify the interrupting channel number during a PI SERVED (and PI ADR IN). CS00–03 select the RH20 by its physical number (Table 1-1) during a PI ADR IN.
DEMAND	EBox to MBC	Causes RH20 to execute command specified by F lines.
Acknowledge (ACKN)	MBC to EBox	Indicates RH20 has been selected and is executing EBus command. (ACKN is not asserted during PI SERVED.)
Transfer (XFER)	MBC to EBox	Indicates RH20 has executed EBus command. (XFER is not asserted during PI SERVED.)
PI Request Lines (PI01–07)	MBC to EBox	Signals an RH20 interrupt. The line asserted depends on the PI channel assignment (PIA) loaded in the RH20 by a CONO.
DATA DISABLE	EBox to MBC	Disables EBus data line transmitters in RH20. (Asserted to allow transfer of diagnostic control data over the data lines during diagnostic operations.)
RESET	EBox to MBC	Initializes RH20 and asserts INIT on Massbus to initialize Massbus devices connected to RH20.
Clock (CLK n)	EBox to MBC	Sequences RH20 Control logic. CLK n connects to MBC n (n = 00–07) from clock distribution module, where it is adjusted to synchronize RH20 and MBox channel operation.

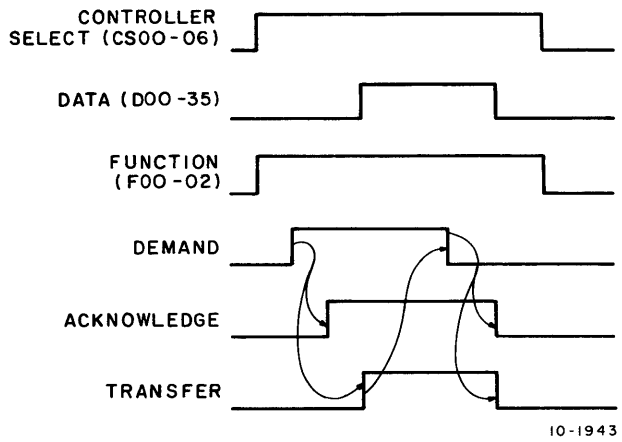


Figure 2-4 CONI/DATAI Operation

The PI SERVED command is issued by the EBox if one or more PI request lines is asserted on the EBus. The channel number corresponding to the highest priority PI request line is placed on CS04–06, 4_8 is encoded on the F lines, and DEMAND is asserted after a time delay. The RH20 (and other EBus controllers), after decoding the F lines and receiving DEMAND, asserts the data line corresponding to its physical number if it is interrupting on the PI channel being served. After a specified delay time, which allows all controllers on the bus to respond, the EBox strobes the data lines and negates DEMAND to end EBus operation. The data lines remain asserted until DEMAND clears. After determining which of the responding controllers has the highest priority, the EBox issues a PI ADR IN command. The channel number is again asserted on CS04–06, the physical number of the controller is asserted on CS00–03, and a function code of 5_8 is asserted on the F lines. DEMAND is then asserted after a delay to allow the controllers to decode the F lines. When an RH20 is addressed; that is, when the physical number asserted on the CS lines matches the RH20's hard-wired address, it asserts ACKN, places an API function word on the data lines, and asserts XFER. Bit format for the API function word is shown in Figure 2-5. The RH20 asserts an EPT address space code (0), a vector interrupt function code (2_8), and a 9-bit interrupt address (IVIR register contents) that is loaded prior to the PI ADR IN command by a DATAO. When XFER is received by the EBox, it strobes the function word from the data lines and negates DEMAND. In the RH20, the trailing edge of DEMAND negates ACKN, XFER, and the data lines to end the operation on the EBus. Timing for both the PI SERVED and PI ADR IN commands is shown in Figure 2-6.

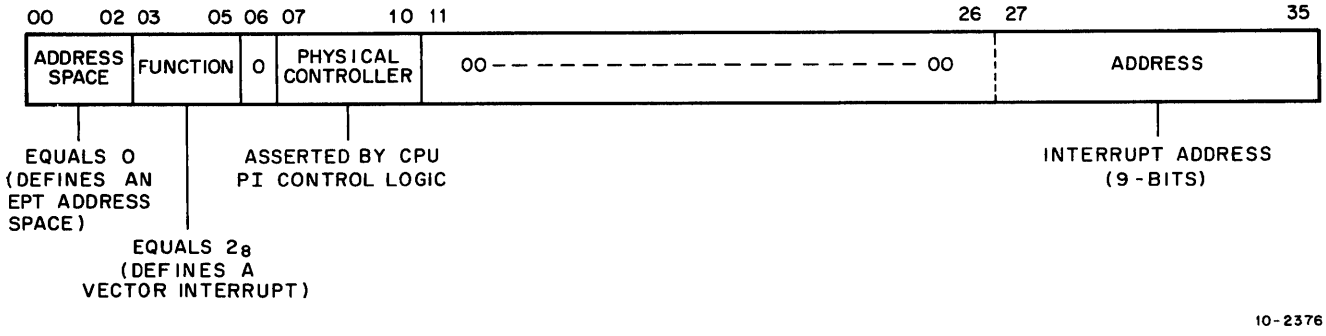


Figure 2-5 RH20 API Function Word Format

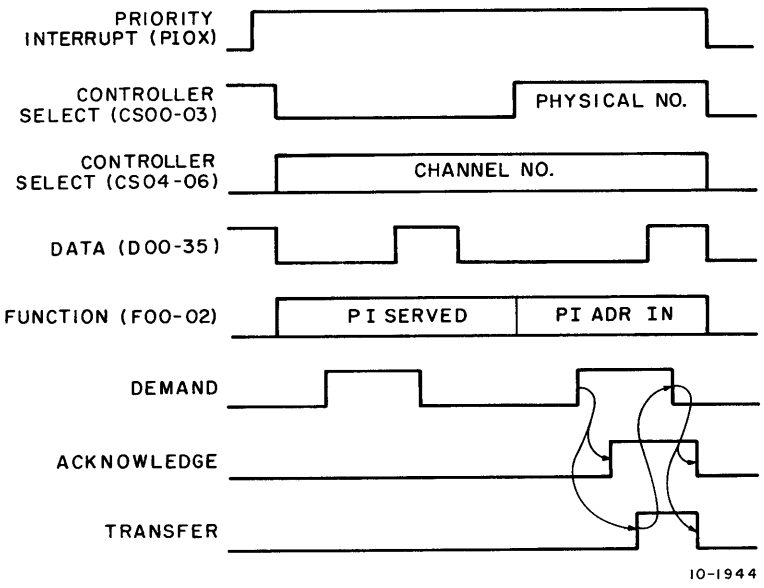


Figure 2-6 PI SERVED/PI ADR IN Operation

2.1.2 Massbus

The Massbus provides a standard interface between controllers and mass-storage devices, such as disks, tapes, and other magnetic or cyclic storage media. It consists of two separate and independent buses, a control bus and a data bus, that allow both asynchronous and synchronous communication between the RH20 and its drives. Bus signals are described in Table 2-2. Information flow is shown in Figure 2-7.

Table 2-2 Massbus Signal Descriptions

Signal	Direction	Description
CONTROL BUS Data Lines (C15–00, CPA)	Bidirectional	Transfer control data and status information between the RH20 and Massbus device (drive). Odd parity (CPA) is simultaneously transferred with the control bus data.
Drive Select Lines (DS2–0)	MBC to drive	Select one of eight (maximum) drives connected to the RH20. These lines are ignored by drives when RS lines = 04_8; that is, when the attention summary register is referenced. All drives respond to this register address.
Register Select Lines (RS4–0)	MBC to drive	Select a register in the drive addressed by the DS lines.
Controller to Drive (CTOD)	MBC to drive	Specifies which direction data is to be transferred on the control bus. CTOD = 1 specifies a control bus write cycle; CTOD = 0 specifies a control bus read cycle.
Demand (DEM)	MBC to drive	Causes a transfer of control data to take place between the RH20 and drive.

Table 2-2 Massbus Signal Descriptions (Cont)

Signal	Direction	Description
Transfer (TRA)	Drive to MBC	Asserted in response to DEM. Indicates data has been strobed from the data lines if CTOD = 1. Indicates data has been placed on the data lines if CTOD = 0.
Attention (ATTN)	Drive to MBC	Indicates a change in drive status or an abnormal condition (end of seek, error, etc.). Can be asserted by more than one drive at a time.
Initialize (INIT)	MBC to drive	Initializes all drives on the bus.
Power Fail (FAIL)	MBC to drive	Indicates a power-up fail condition in the RH20.
DATA BUS Data Lines (D17–00, DPA)	Bidirectional	Transfers synchronous data between the RH20 and drive. Odd parity (DPA) is simultaneously transferred with the data.
Sync Clock (SCLK)	Drive to MBC	Data clock. The RH20 gates and strobes data at the trailing edge. The drive gates data at the leading edge.
Write Clock (WCLK)	MBC to drive	Returning SCLK. Controls the strobing of data during a data bus write cycle. Drive strobes data at the leading edge.
RUN	MBC to drive	Enables the drive to perform a read/write operation. During the data transfer, the drive samples RUN at the end of each data block (trailing edge of EBL). If RUN is still asserted, the drive continues the transfer; if RUN is negated, the drive terminates the transfer.
End of Block (EBL)	Drive to MBC	Normally indicates the end of the data block. It must be asserted at least once after a read/write command has been accepted by a drive. Thus, it is asserted following an error that terminates a data transfer in mid-data block.
Exception (EXC)	Bidirectional	When asserted by a drive, indicates that an error was detected during a read/write operation. NOT asserted by the RH20 during normal operation.
Occupied (OCC)	Drive to MBC	Indicates that a drive has accepted a read/write command. Negated at the trailing edge of the last EBL pulse.

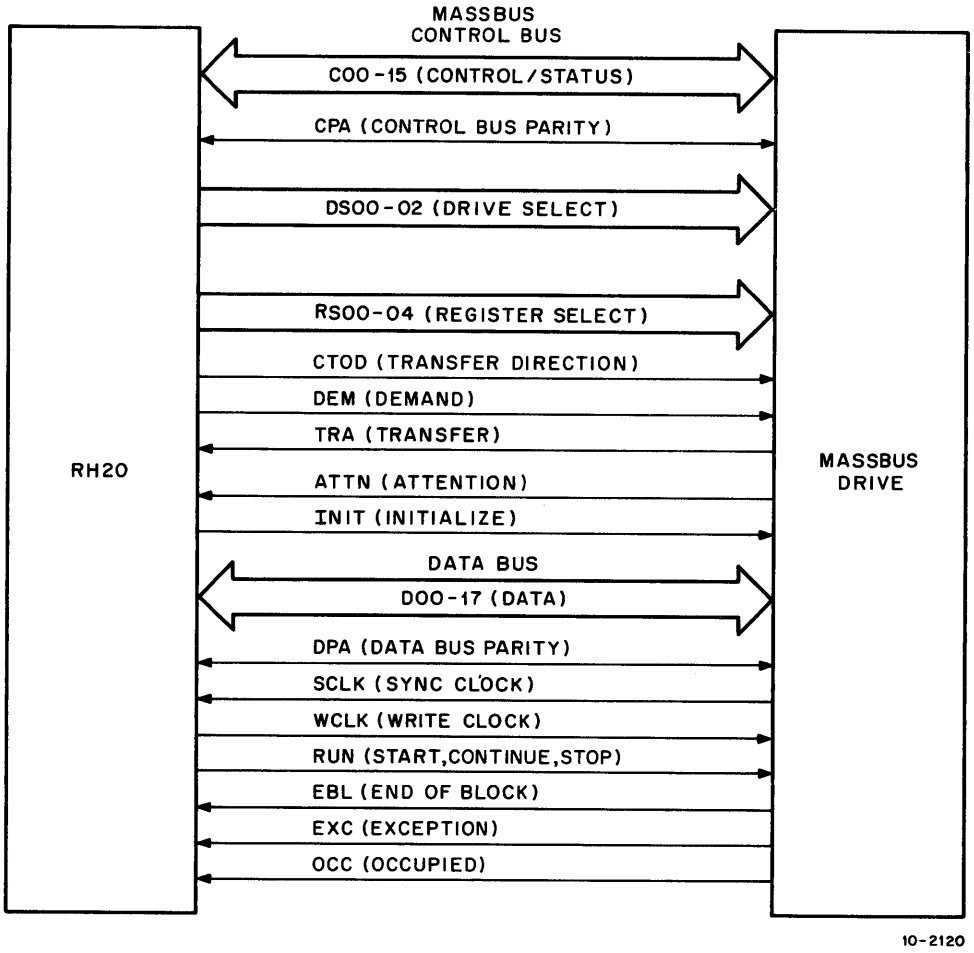


Figure 2-7 Massbus Interface

During a control bus write operation, the RH20 asserts the C lines, the DS lines, the RS lines, and CTOD. After these signals have settled on the bus, the controller asserts DEM to cause the drive to load the control data on the C lines into the addressed register in the selected drive. When the register data has been loaded, the drive asserts TRA. The RH20 then negates DEM, which negates TRA in the drive to end the handshaking sequence. Timing is shown in Figure 2-8.

During a control bus read operation, the RH20 asserts only the DS and RS lines before asserting DEM. When DEM is received by the selected drive, it places the contents of the addressed register on the C lines and asserts TRA. The RH20 uses TRA to strobe the C lines and negate DEM. The trailing edge of DEM then negates TRA to complete the handshake. Timing is shown in Figure 2-9.

The control bus read cycle is initiated by the RH20 as a result of a DATAI command (external register address). The control bus write cycle is initiated by a DATAO (external register address) or by a command file transfer. The hardware-initiated command file transfer provides the means for starting a synchronous data transfer (read/write) in a drive (and RH20). It first loads the drive's data block address register (RS = 05_8). It then loads the drive's control register (RS = 00) with a read/write command to start the data transfer. Non-data transfer commands (e.g., Seek, Search, etc.) are loaded in a drive's control register by a DATAO, not by a command file transfer. Once a read/write command is loaded in a drive, it asserts OCC and begins a search for the specified block address (disks) or starts up the recording medium (tapes). With RUN asserted by the RH20, SCLKs are generated on the data bus portion of the Massbus when the drive search operation completes (disks) or the recording medium is up to speed (tapes).

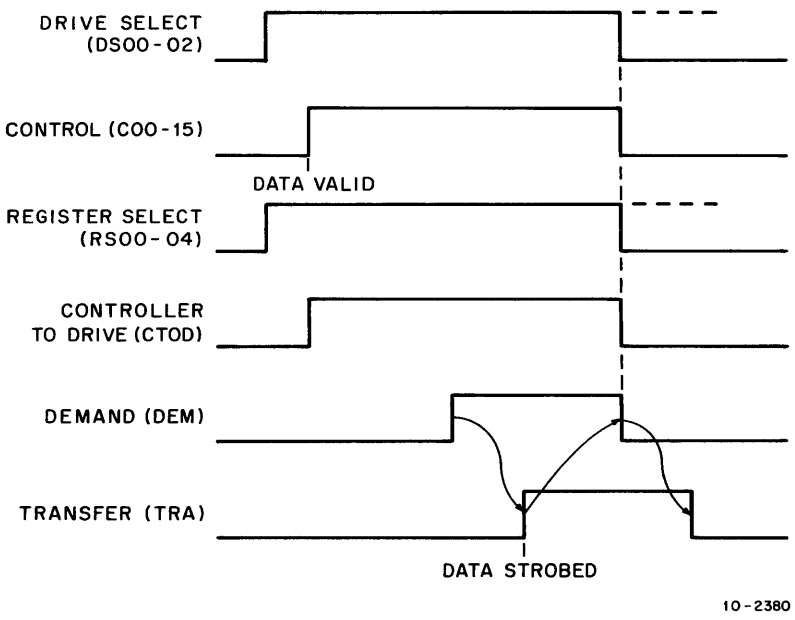


Figure 2-8 Control Bus Write Operation

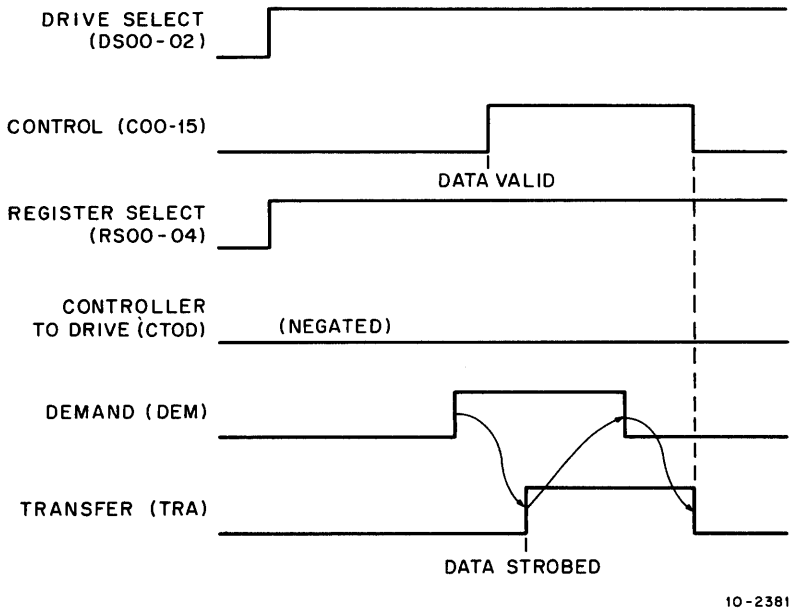


Figure 2-9 Control Bus Read Operation

During the data bus write operation, data is placed on the data lines by the RH20 at the trailing edge of SCLK and strobed by the drive at the leading edge of WCLK. (WCLK is generated in the RH20 by the received SCLK signal.) Timing is shown in Figure 2-10. During the data bus read operation, data is placed on the data lines by the drive at the leading edge of SCLK and strobed from the bus at the trailing edge. Timing is shown in Figure 2-11.

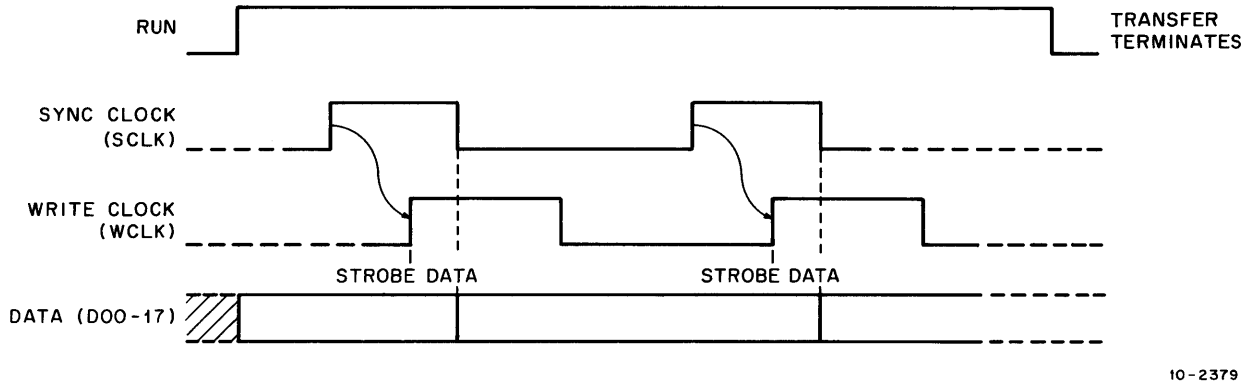


Figure 2-10 Data Bus Write Operation

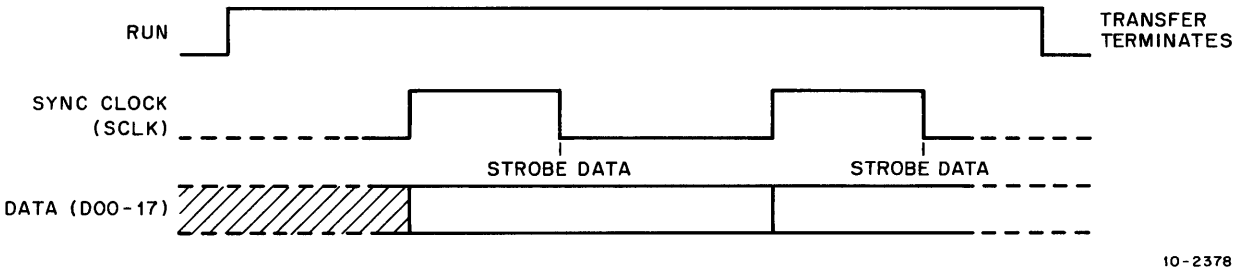


Figure 2-11 Data Bus Read Operation

During the write and read operations, the drive asserts EBL after the last word of each data block is transferred. The RH20 terminates the data transfer by negating RUN at the leading edge of the EBL signal corresponding to the last data block.

EXC is asserted by the drive to signal the RH20 that an error condition has occurred during a data transfer. In addition, the ATTN line is asserted to indicate the error condition or status change in the drive. ATTN may be asserted for any of the following conditions:

a. An error is detected while no data transfer is taking place (asserted immediately).

b. At the completion of a data transfer command if an error occurred during the data transfer.

c. At the completion of a non-data transfer command (e.g., Search).

2.1.3 CBus

The Channel Bus (CBus) is a synchronous bus that connects the integral data channel logic of the MBox to a maximum of eight RH20 mass-storage controllers. The CBus is composed of a 36-bit wide data path and its associated control lines. It transfers high-speed data between the MBox channels and the RH20 controllers. Each CBus line is described in Table 2-3. Figure 2-12 illustrates the CBus configuration.

Table 2-3 CBus Signal Descriptions

Signal	Direction	Description
Data Lines (D00–35)	Bidirectional	Transfers high-speed data. The lines are valid only during a data cycle. The MBox will place zeros on the data lines for an RH20 (during its data cycle) whenever there is no data transfer request from that RH20.
Parity Left/ Parity Right	Bidirectional	Transfers the computed parity for the left- and right-half words on the data lines.
Select Lines (SEL 0–7)	MBox to MBC	Continuously select the RH20 on the bus. Define the beginning of the four channel cycles (Select, Request, Wait, Data) associated with each controller.
RESET	MBC to MBox	Asserted by an RH20 during its data cycle. Causes the MBox to: Clear the data buffers associated with the selected RH20. Reset the command list pointer associated with the selected RH20 to the initial control word address. Negate all status and data lines associated with the selected RH20 after the above two steps are completed.
START	MBC to MBox	Asserted by an RH20 to begin a transfer. START is asserted once during its data cycle, and only when READY is negated.
Controller to Memory (CTOM)	MBC to MBox	Asserted by an RH20 to indicate the transfer direction to the MBox. For a read data transfer, CTOM is asserted; for a write data transfer, CTOM is negated.
READY	MBox to MBC	Asserted by the MBox (during the data cycle) after it detects START from the RH20, and after the MBox is ready for a data transfer. For a write data transfer, the MBox is required to have at least one data word from memory in its buffer before asserting READY. READY will be negated only after sensing DONE, and after the MBox is prepared to start another transfer.

Table 2-3 CBus Signal Descriptions (Cont)

Signal	Direction	Description
REQUEST	MBC to MBox	Asserted by an RH20 during its request cycle when: One of its data buffers is full, during a read data transfer. One of its data buffers is empty, during a write data transfer. An RH20 will not assert REQUEST if: READY is not asserted by the MBox. ERROR has been asserted by the MBox during the current transfer. LAST WORD has been asserted by the MBox during the current transfer. DONE has been asserted by the RH20 during the current transfer. For a read data transfer, the RH20 places data (during its data cycle) on the data lines and the MBox will strobe the lines at the trailing edge of the same data cycle. For a write data transfer, the above operation is reversed.
DONE	MBC to MBox	Asserted by an RH20 during its data cycle to terminate a data transfer. No further data requests will be made after DONE is asserted.
STORE	MBC to MBox	Asserted by an RH20 together with DONE to cause channel status to be stored in the channel's assigned reset and status logout area. Asserted when: The current transfer is terminated due to errors detected in the RH20. The current transfer command in the RH20 specifies that STORE be sent to the MBox at the conclusion of the transfer.

Table 2-3 CBus Signal Descriptions (Cont)

Signal	Direction	Description
LAST WORD	MBox to MBC	Asserted by the MBox during the data cycle (for write data transfer only) at the same time the last data word is sent to a controller. No further data requests will be made by the RH20 after detecting LAST WORD.
ERROR	MBox to MBC	Asserted by the MBox (during the data cycle) to inform the RH20 that the current data transfer must terminate due to an error in the MBox. On sensing ERROR, the RH20 terminates the transfer by not issuing any further requests. The RH20 also asserts DONE during a subsequent data cycle.

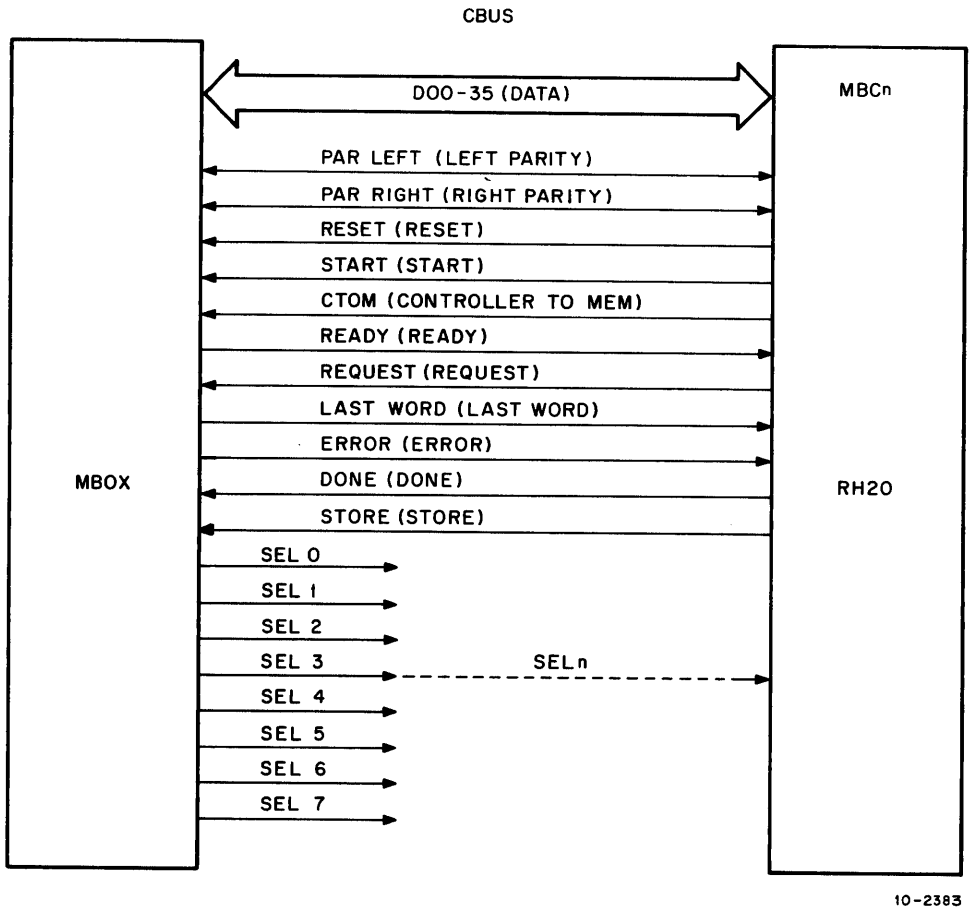


Figure 2-12 CBus Interface

The RH20 selection order is on a set priority basis using a time division multiplexing technique. Selection is of the following order: 0, 1, 2, 3, 4, 5; 0, 1, 2, 3, 6, 7. Each channel associated with an RH20 is assigned a specific time slot within a fixed repetition rate and the corresponding RH20 is selected during the same time slot in successive repetition cycles. Selection is by the CBUS SEL n level (n = 0–7), which defines the first of four channel cycles (Select, Request, Wait, Data) that are used by the MBox and the controller to transfer control information and data during the synchronous data transfer. (Figure 2-13 illustrates RH20 selection and timing.)

a. Select Cycle – The select line for a particular controller is asserted during this cycle.

b. Request Cycle – The selected controller will assert its request line (if data request is needed) during this cycle.

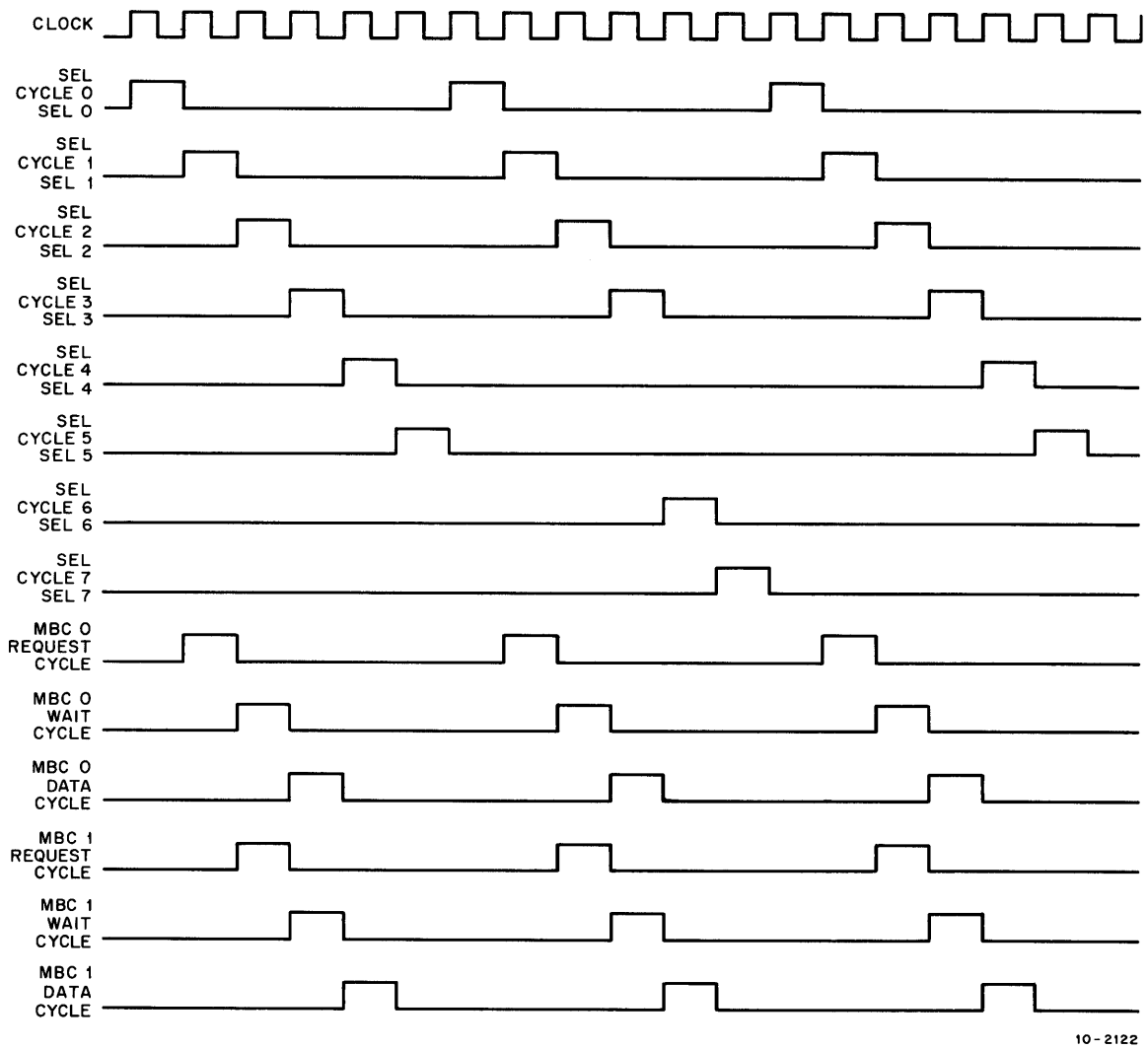


Figure 2-13 Basic RH20 Selection and Timing

c. Wait Cycle – This cycle is used by the MBox to prepare data and status for transmission. Neither data nor control lines are asserted during this cycle.

d. Data Cycle – Data is placed on the data lines by either the MBox or the controller during this cycle. The receiver will strobe the data lines at the trailing edge. All control lines, except the REQUEST line, are asserted during the data cycle.

Figure 2-14 shows CBus timing during a data transfer. If READY is negated, the RH20 starts the channel by asserting START during the data cycle. At the same time, it asserts the RESET line if the channel is to be initialized and it asserts CTOM if the data transfer is a device read operation. When the channel is ready to transfer data over the CBus, it asserts READY during the data cycle. For a device write operation, the channel will have at least one word in its data buffers.

After receiving READY, the RH20 asserts REQUEST during the request cycle whenever it requires a data word from the channel (device write) or whenever it requires that the channel accept a data word (device read). The words are asserted on the DATA lines following the corresponding REQUEST signal. When the channel places the last word on the CBus during a device write operation, it asserts LAST WORD. In response, the RH20 makes no more data requests and asserts DONE during a subsequent data cycle. The RH20 also asserts DONE when it has transferred all data over the CBus during a device read operation and when a transfer error is detected during a read or write. (CBUS ERROR causes DONE to be asserted by the RH20 when the transfer error is detected in the channel.) DONE causes the channel to terminate the operation. READY is negated when the channel is prepared to begin another data transfer.

2.2 DATA CHANNEL RESET AND LOGOUT AREAS

Each of the eight data channels associated with an RH20 has a four-word reset and status logout area in the executive process table (EPT). As shown in Figure 2-15, the reset and logout areas are assigned to consecutive EPT locations $00–37_8$. Locations $00–03_8$ are used by channel 0, locations $04–07_8$ by channel 2, etc.

Word 0 of each channel's reset and logout area is the initial control word address; that is, it is the location from which the channel takes the first channel command (control word) when CBUS RESET has been asserted by the RH20 at the start of the data transfer. Words 1–3 contain the two channel status words and an unused location. The channel loads words 1 and 2 with status information at the end of a data transfer when CBUS STORE has been asserted by the RH20. Word 3 is not currently used to hold status information. However, the program may arbitrarily use the location as the vector interrupt address for the associated channel.

2.3 INTERNAL/EXTERNAL REGISTERS

To permit any type of mass-storage device (disk, tape, etc.) to be interfaced to a common controller (the RH20), the working registers are divided between the controller and the device. Registers located in the RH20 are called internal registers. Registers in the device (drive) are called external registers. Except for the preparation register, which is not addressed directly, all internal/external registers are assigned a 6-bit register select (RS) code. The RS code is asserted as part of the control data (bits 00–05) during a DATAO.

Internal registers have an RS code equal to $70_8–77_8$. External registers have an RS code equal to $00–37_8$. When an external register is addressed, a drive (0–7) must also be specified. The drive address, or drive select (DS) code, is asserted together with the RS code as part of the DATAO control data (bits 15–17). The DATAO loads an internal/external register. The DATAI command reads an internal/external register.

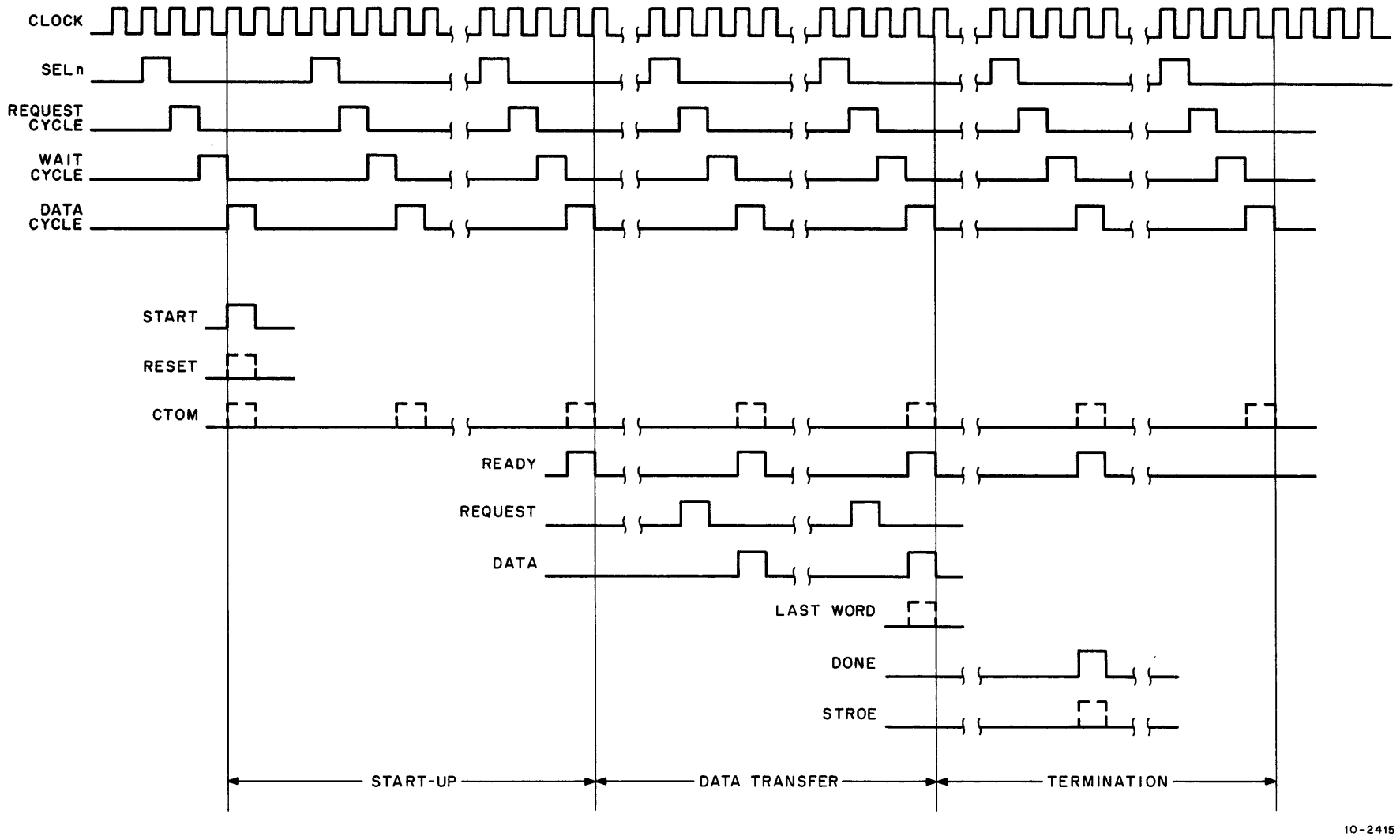


Figure 2-14 CBus Operation

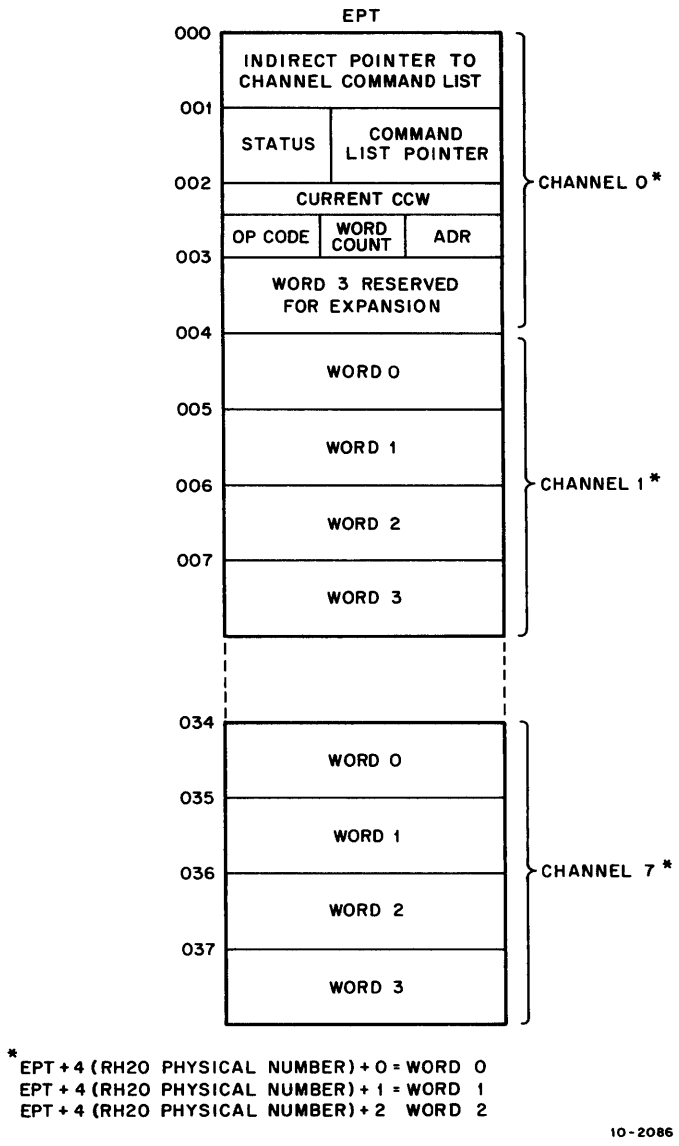


Figure 2-15 Channel Reset and Status Logout Area

The internal registers implemented in the RH20 are listed in Table 2-4 and described below. Bit format is shown in Figure 2-16. Control bit functions are described in Subsection 2.5.

a. Preparation Register – Loaded by all DATAO commands. It stores the RS code and it also stores the DS code (and two control bits) when an external register is referenced. The primary function of the preparation register is to hold the address and control information loaded by a DATAO so that the specified register may be read by a subsequent DATAI.

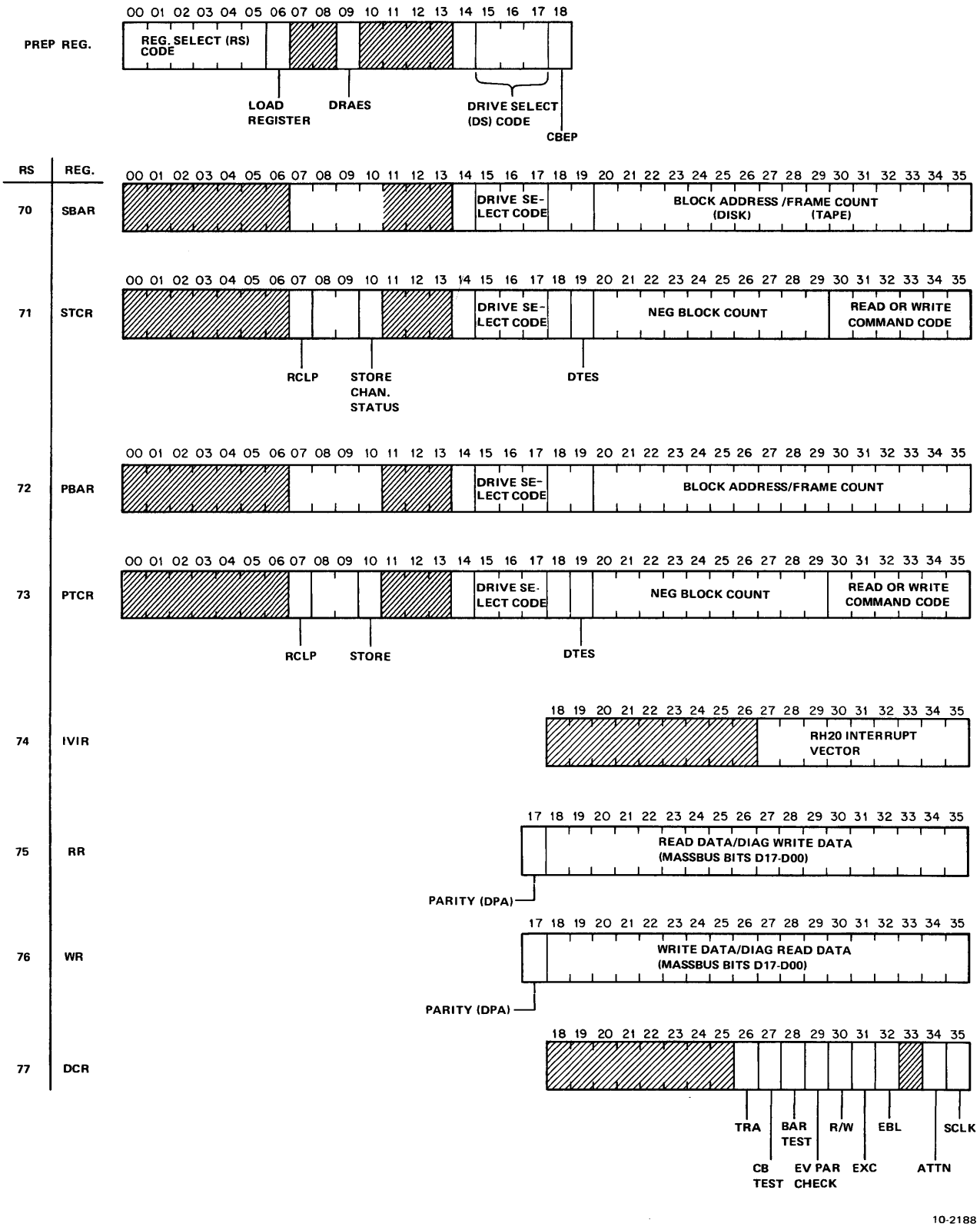


Figure 2-16 Internal Registers

Table 2-4 Internal Register Summary

RS Code (Octal)	Name	Write	Read
xx	Preparation Register (not addressed directly)	x	x
70	SBAR – Secondary Block Address Register	x	x
71	STCR – Secondary Transfer Control Register	x	x
72	PBAR – Primary Block Address Register		x
73	PTCR – Primary Transfer Control Register		x
74	IVIR – Interrupt Vector Index Register	x	x
75	RR – Read Register (Diagnostic Use)		x
76	WR – Write Register (Diagnostic Use)	x	
77	DCR – Diagnostic Control Register (Diagnostic Use)	x	

b. Secondary Command File (SBAR/STCR; RS = 70/71) – Two locations in the four-location command file that are loaded by the program to start a synchronous data transfer. The BAR location contains a drive address and a starting sector/track address (disks) or frame count (tapes). The address or frame count is written in register 05_8 of the specified drive during a command file transfer. The TCR location contains a drive address, a negative block count (specifying the length of the data transfer), a read/write function code, and various control bits. During a command file transfer, the block count and control bits are loaded in the RH20 data transfer control logic and the read/write functions code is loaded in the specified drive's control register (RS = 00_8).

c. Primary Command File (PBAR/PTCR; RS = 72/73) – Read-only registers. The secondary command file is redefined as the primary command file during a command file transfer. The other two command file locations then become the secondary file and may be loaded by the DATAO (RS = 70/71).

d. Interrupt Vector Index Register (RS = 74_8) – Loaded with a vector interrupt address (an EPT location) that is read by the PI ADR IN command following an RH20 interrupt.

e. Read Register (RS = 75_8) – A read-only register. Part of the data path during a normal high-speed read data transfer. The register is read only during diagnostic operation when data is clocked through the RH20 (under program control) at a slow speed. During a diagnostic write operation, data fetched from the channel and transmitted on the Massbus is looped back to the read register where it can be read and verified. During a diagnostic read operation, simulated read data (in the write register) is clocked into the read register. As for the write, the program can read and check the data.

f. Write Register (RS = 76_8) – A write-only register. Part of the data path during a normal write data transfer operation. The register is loaded only during a diagnostic read data transfer. Its contents are looped back to the inputs of the Massbus receivers to provide simulated read data.

g. Diagnostic Control Register (RS = 77_8) – A write-only register. Provides the necessary control functions during diagnostic operations to enable data loop-back paths and to simulate the Massbus control signals normally asserted by a drive.

The number and type of external registers accessed via the RH20 depend on which devices are connected to the Massbus. The external registers in a typical fixed-head drive (RS04), moving-head drive (RP04), and tape drive system (TM02/TU45) are listed in Table 2-5.

Table 2-5 External Register Summary

RS Code (Octal)	Name	Write	Read	Fixed-Head Disk (RS04)	Moving-Head Disk (RP04)	Magtape (TM02/TU45)
00	Control (command)	x	x	x	x	x
01	Status		x	x	x	x
02	Error (1)	x	x	x	x	x
03	Maintenance	x	x	x	x	x
04	Attention Summary	x	x	x	x	x
05	Block Address	x	x	x	x	
	Frame Count	x	x			x
06	Drive Type		x	x	x	x
07	Current Block Address		x	x	x	
	Check Character	x	x			x
10	Error (2)	x	x		x	
11	Offset	x	x		x	
	Tape Control	x	x			x
12	Cylinder Address	x	x		x	
13	Current Cylinder Address		x		x	
14	Serial Number		x		x	x
15	Error (3)	x	x		x	
16	ECC Position		x		x	
17	ECC Pattern		x		x	

2.4 DRIVE COMMANDS

Drive commands are classified into two categories:

a. Non-data transfer commands
b. Data transfer commands

The non-data transfer commands do not cause MBox data channel transfers. These commands are generally used to set up the drive for a subsequent data transfer command. Typically, the drive asserts the Massbus ATTN line to signal that the setup is complete. Data transfer commands are those that cause data channel transfers. These commands are limited to read and write operations.

A drive command is initiated by loading a six-bit command code in the drive's control register. Codes $01–47_8$ are reserved for non-data transfer commands. Codes $51–77_8$ are for data transfer commands; that is, for read, write, and write-check commands. The write-check commands are not implemented in the RH20 system. The commands and corresponding command codes used for a typical fixed-head disk (RS04), moving-head disk (RP04), and tape drive system (TM02/TU45) are listed in Table 2-6.

Table 2-6 Drive Commands

Command Code (Octal)	Drum and Fixed-Head Disk (RS04)	Moving-Head Disk (RP04)	Magnetic Tape (TM02/TU45)
01	No Operation	No Operation	No Operation
03		Unload	Rewind, Off-line
05		Seek	
07		Recalibrate	Rewind
11	Drive Clear	Drive Clear	Drive Clear
13		Release	
15		Offset	
17		Return to Centerline	
21	Readin Preset	Readin Preset	Readin Preset
23		Pack Acknowledge	
25			Erase
27			Write File Mark
31	Search	Search	Space Forward
33			Backspace
*51	Write Check Data	Write Check Data	Write Check Forward
*53		Write Check Header and Data	
*57			Write Check Reverse
61	Write Data	Write Data	Write Forward
63		Write Header and Data	
71	Read Data	Read Data	Read Forward
73		Read Header and Data	
77			Read Reverse

* Note: Not implemented in the RH20.

A non-data transfer command (Seek, Rewind, etc.) is loaded in a drive via the RH20 by a DATAO command that addresses the control register directly (RS = 00_8). A data transfer command (read/write) is loaded by a command file transfer, which is initiated by the RH20 when the STCR (Subsection 2.3) has been loaded (also by a DATAO) and no read/write operation is already in progress. The data transfer command code is part of the STCR control data. The program must not load a non-data transfer code in the STCR. Conversely, the program should not load a data transfer command when addressing the control register directly. The first results in a hung controller and channel. The latter results in a drive error.

2.5 RH20 INSTRUCTION SET

The RH20 and the attached Massbus devices are controlled via the EBus by the CONO, CONI, DATAO, and DATAI commands.

2.5.1 CONO

The CONO command loads miscellaneous control data in the RH20. Control bit format is shown in Figure 2-17.

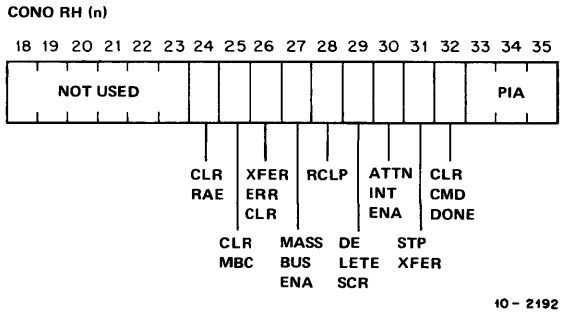


Figure 2-17 CONO Command Format

a. CLR RAE [CRAEI] (bit 24) - Clears RAE (register access error: CONI, bit 24) and the associated RH20 interrupt request.

b. CLR MBC [CMC] (bit 25) - Initializes the RH20 and all attached Massbus devices to the power-up state. If the RH20 is initialized when a synchronous data transfer is in progress, the program must also initialize the channel (CONO: bit 28 = 1) to prevent a hung channel control in the MBox.

c. XFER ERR CLR [TEC] (bit 26) - Clears all transfer error indicators (CONI: bits 18–23, 26).

d. MASSBUS ENA [MBE] (bit 27) - Enables the Massbus transmitters in the RH20. Normally set by the program except in fail-soft (redundant) configurations where more than one controller is connected to the same Massbus. For this case, only one controller should be enabled (bit 27 = 1) at any one time.

e. RCLP [RCLP] (bit 28) - Initializes the MBox channel associated with the RH20. Resets the command list pointer to the initial control word address, clears the data buffers, and clears channel status.

f. DELETE SCR [DSCRF] (bit 29) - Initializes the control logic associated with a command file transfer. Provides a means to clear a hung command file transfer without initializing the controller. A command file transfer would be hung if the previous data transfer (read/write) caused a transfer error (CONI: bits 18–23, 26).

g. ATTN INT ENA [MEAE] (bit 30) - Allows the ATTN (attention) signal on the Massbus to generate an RH20 interrupt.

h. STP XFER [ST] (bit 31) - Terminates the current synchronous data transfer. Should be used only when the RH20 is hung. Status bits are not cleared and can be examined following termination. (Status indicators are cleared when a transfer is terminated by CLR MBC.)

i. CLR CMD DONE [CCMD] (bit 32) - Clears CMD DONE (CONI: bit 32) and the associated RH20 interrupt request.

j. PIA [PIA] (bits 33–35) - Priority interrupt channel number for vector interrupts generated by the RH20. PIA = 0 inhibits RH20 interrupt requests.

2.5.2 CONI

The CONI command reads miscellaneous control status in the RH20. Status bit format is shown in Figure 2-18.

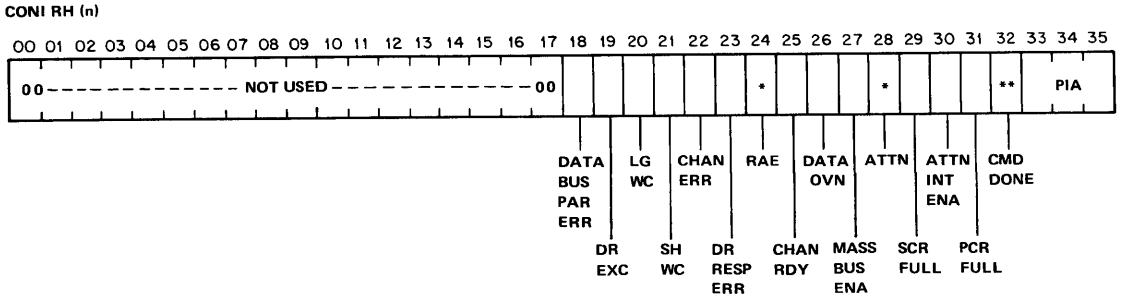


Figure 2-18 CONI Command Format

a. DATA BUS PAR ERR [DPE] (bit 18) - Indicates that bad parity was detected on the Massbus (data bus) data lines during a read/write data transfer. During a write data transfer, the drive should also detect bad parity. The RH20 will set CMD DONE (CONI = bit 32) and generate an RH20 interrupt at the end of the current data block, either because of DATA BUS PAR ERR (read) or DR EXC (write), provided DTES (DATAO STCR: bit 19) = 0. If DTES = 1, a data bus parity error in the RH20 or drive will not terminate the transfer. DATA BUS PAR ERR is cleared by a CONO: bit 26 = 1.

b. DR EXC [DEE] (bit 19) - Indicates that an error was detected in a drive during a read/write data transfer. Set in the RH20 by the assertion of EXC (exception) and EBL (end of block) on the Massbus. When the error causes an immediate termination of the transfer in the drive (a class B error), the RH20 immediately sets CMD DONE (CONI = bit 32) and generates an interrupt. When the drive error does not terminate the transfer immediately (a class A error) and DTES (DATAO STCR: bit 19) = 0, the RH20 sets CMD DONE and generates an interrupt at the end of the current data block. If DTES = 1 when a class A error occurs, DR EXC will not terminate the transfer. The error is cleared by a CONO: bit 26 = 1.

c. LONG WC [LWCE] (bit 20) - Set only during a write data transfer. Indicates that the RH20 terminated the transfer of data from the channel but the channel did not reach the final word count specified by the command list (i.e., channel has long word count). Occurs when the block count loaded by the program (DATAO STCR: bit 20–29) does not correspond to the channel word count or when the RH20 has terminated abnormally. Abnormal termination is usually caused by another error detected in either the RH20 or channel. The RH20 asserts CMD DONE (CONI = bit 32) and generates an interrupt immediately or at the end of the current data block, depending on the condition causing the error. LONG WC is cleared by a CONO: bit 26 = 1.

d. SHORT WC [SWCE] (bit 21) – Set only during a write data transfer and after a DATA OVN error has occured (CONI : bit 26 = 1). Indicates that the channel transferred all the data specified by the channel command list but the controller did not finish the transfer (i.e., channel has short word count). Occurs when the block count loaded by the program (DATAO STCR: bits 20–29) does not correspond to the channel word count. The RH20 sets CMD DONE (CONI = bit 32) and generates an interrupt at the end of the current data block. SHORT WC is cleared by a CONO: bit 26 = 1.

e. CHAN ERR [CE] (bit 22) – Indicates that the MBox channel detected an error during a synchronous data transfer. The RH20 sets CMD DONE (CONI: bit 32) and generates an interrupt at the end of the current data block. CHAN ERR is cleared by a CONO: bit 26 = 1.

f. DR RESP ERR [DR] (bit 23) – Indicates no drive response (non-existent drive) when the RH20 attempted to load a read/write command in a drive during a command file transfer. The RH20 sets CMD DONE (CONI : bit 32) and generates an interrupt. DR RESP ERR is cleared by a CONO: bit 26 = 1.

g. RAE [RAE] (bit 24) – Indicates that an error was detected when an external register (RS = $00–37_8$) was accessed by a DATAO or DATAI. Error conditions checked are:

   1. Control Bus Time-Out – Non-existent drive addressed.

   2. Control Bus Parity Error – Bad parity detected on the Massbus (control bus) data lines during an external register read operation.

Once set, RAE generates an RH20 interrupt and it inhibits further register write operations (to both internal and external registers) if DRAES (DATAO : bit 09) was set to 0 by the instruction causing the error. If DRAES = 1, an interrupt is not generated and subsequent register writes are allowed. RAE is cleared by a CONO: bit 24 = 1.

h. CHAN RDY [CNR] (bit 25) – Indicates that the channel is ready to start a data transfer.

i. DATA OVN [DOE] (bit 26) – During a read data transfer, indicates that a drive sent data to the RH20 and the RH20 buffers were full; that is, the previously received data had not yet been sent to the channels. During a write data transfer, indicates that a drive demanded data from the RH20 and the RH20 data buffers were empty; that is, the data had no yet been received from the channel. DATA OVN causes the RH20 to set CMD DONE (CONI : bit 32) and generate an interrupt at the end of the current data block. DATA OVN is cleared by a CONO : bit 26 = 1.

j. MASSBUS ENA [MBE] (bit 27) – Indicates that the Massbus transmitters are enabled. Set by a CONO: bit 27 = 1.

k. ATTN [MBA] (bit 28) – Indicates the state of the Massbus attention line. The attention line is set by a drive at the completion of non-data transfer commands (Seek, Search, etc.) and when drive errors are detected. ATTN causes an RH20 interrupt if ATTN INT ENA (CONO: bit 30) is set. ATTN is negated by clearing the attention condition in the drive(s).

l. SCR FULL [SCRF] (bit 29) – Indicates that the secondary command file (the TCR location) is loaded. Cleared by a CONO: bit 29 = 1.

m. ATTN INT ENA [MBAE] (bit 30) – Indicates that the Massbus attention line is enabled to generate an RH20 interrupt. Set by a CONO: bit 30 = 1.

n. PCR FULL [PCRF] (bit 31) - Indicates that the primary command file (previously, the secondary file) is full and that a data transfer is in progress. Cleared by a CONO: bit 31 = 1.

o. CMD DONE [CMD] (bit 32) - Indicates that a data transfer has terminated, with or without error. Generates an RH20 interrupt. CMD DONE and the associated RH20 interrupt request are cleared by a CONO: bit 32 = 1.

p. PIA [PIA] (bits 33-35) - Indicates the priority interrupt channel number (0-7) loaded by a CONO: bits 33-35.

2.5.3 DATAO

The DATAO command is used to write both internal and external registers. Bit format for the DATAO is shown in Figure 2-19.

2.5.3.1 Preparation Register Data - The preparation register is not addressed directly. It is loaded with address and control information (bits 00-06, 09, and 14-18) by all DATAO commands provided there has been no previous register access error detected (RAE, CONI: bit 24 = 0) or, if there has been an error, it has been disabled (DRAES, DATAI: bit 09 = 1). When RAE = 1 and DRAES = 0, the preparation register is not loaded and the DATAO command is not executed by the RH20.

a. REGISTER SELECT [RS] (bits 00-05) - Six-bit register select code. RS = $70-77_8$ when addressing an internal register. RS = $00-37_8$ when addressing an external register.

b. LOAD REGISTER [LR] (bit 06) - When equal to 1, causes the addressed register to be written. When equal to 0, only the preparation is loaded, thus providing a means to store address and control information for a subsequent register read operation (DATAI).

c. DRAES [DRAES] (bit 09) - Prevents RAE (CONI: bit 24) from generating an RH20 interrupt request and from inhibiting subsequent register write operations (i.e., loading of the preparation register).

d. DRIVE SELECT [DS] (bits 15-17) - Specifies the drive number ($0-7_8$) when addressing an external register (RS = $00-37_8$).

e. CBEP [CBEP] (bit 18) - When writing an external register (RS = $00-37_8$, LR = 1), causes even parity to be transmitted on the Massbus (control bus) data lines. Used by the diagnostic programmer to cause parity errors in a drive.

2.5.3.2 External Register (RS = $00-37_8$) Data

a. EXTERNAL REGISTER DATA (bits 20-35) - Specifies the register contents when an external register is written.

2.5.3.3 SBAR (RS = 70) Data - The SBAR, together with the STCR (RS = 71), are loaded with the data transfer commands that specify a synchronous data transfer. The SBAR should be loaded first because loading the STCR initiates an immediate command file transfer of the data transfer commands (if a read/write is not already in progress) to start the data transfer.

a. DRIVE SELECT [DS] (bits 15-17) - Specifies the drive number ($0-7_8$) of the device that is to perform the synchronous data transfer.

b. BLOCK ADDRESS/FRAME COUNT [DBA] (bits 20-35) - For disk drives, specifies the starting block address (track and sector). For tape drives, specifies the number of frames (in a record) to be read or written.

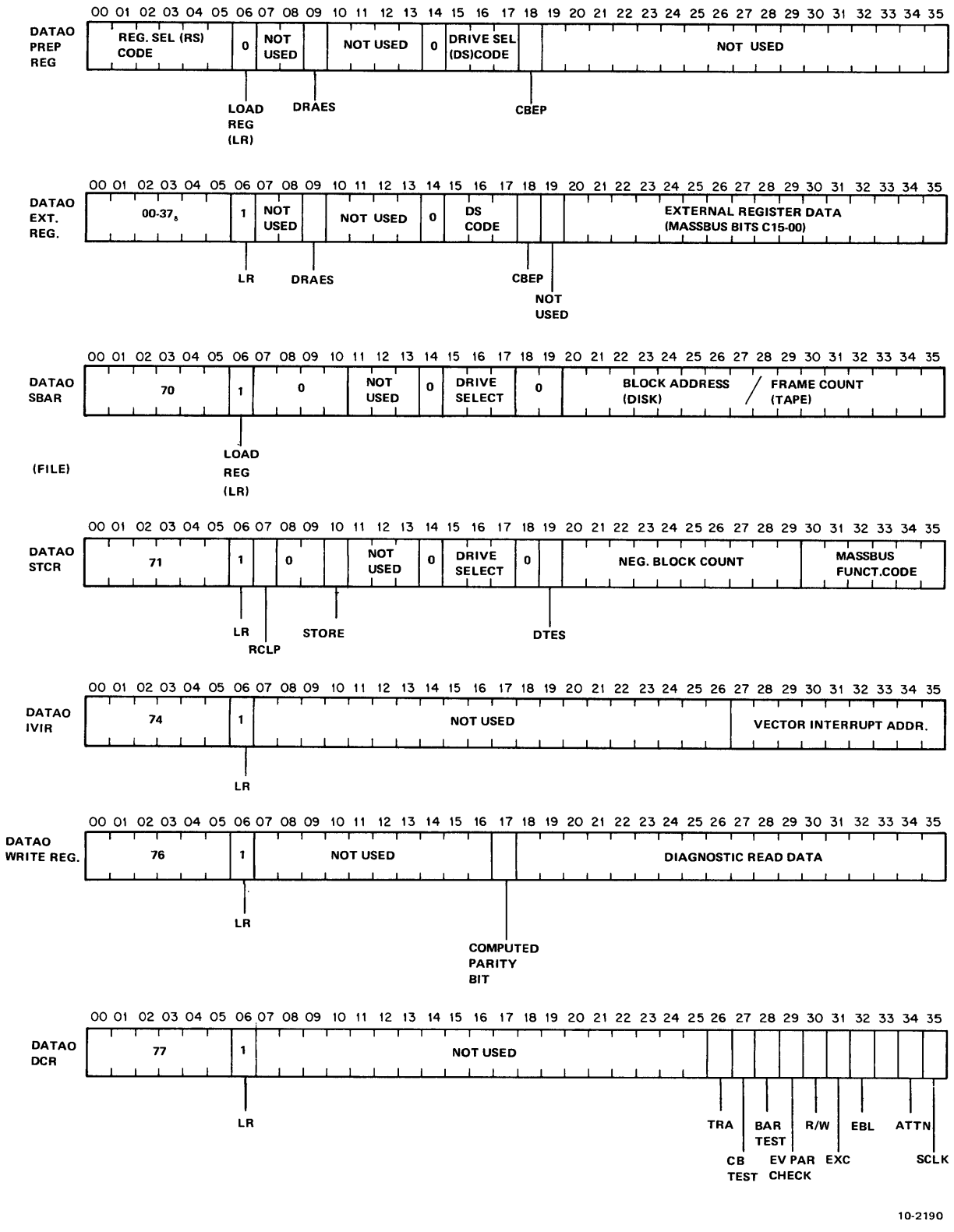


Figure 2-19 DATAO Command Format

2.5.3.4 STCR (RS = 71) Data

a. RCLP [RCLP] (bit 07) – Causes the MBox channel associated with the RH20 to be initialized (command list pointer reset, data buffers and status cleared) when the data transfer is started.

b. STORE [SES] (bit 10) – Causes the MBox to store channel status at the end of the data transfer. (Channel status bits are described in Subsection 2.7.)

c. DRIVE SELECT [DS] (bits 15–17) – Specifies the drive number of the device that is to perform the data transfer.

d. DTES [DTES] (bit 19) – Prevents DATA BUS PAR ERR (CONI: bit 18) or DR EXC (CONI : bit 19), causes a class A drive error, from terminating the data transfer.

e. NEGATIVE BLOCK COUNT [NBC] (bits 20–29) – Specifies the number (in 2's complement) of data blocks (sectors or records) to be read or written during the data transfer. For magnetic tape drives, only one record at a time shall be specified.

f. MASSBUS FUNCTION CODE [MFC] (bits 30–35) – Six-bit command code for the data transfer command. (Refer to Table 2-6.) Codes $61–67_8$ are reserved for write commands. Codes 71–77 are reserved for read commands. Codes for non-data transfer commands (Seek, Rewind, etc.) should not be loaded in the STCR. These commands are loaded directly in the drive by addressing the drive's control register (RS = 00).

2.5.3.5 IVIR (RS = 74) Data

a. INTERRUPT VECTOR ADDRESS [IVR] (bits 27–35) – Nine-bit EPT interrupt address. The program may specify the currently unused location (word 3) in the controller's assigned EPT reset and status area.

2.5.3.6 WR (RS = 76) Data – Loaded by the program during diagnostic operations.

a. DIAGNOSTIC READ DATA [D] (bits 17–35) – Simulated read data. Bit 17 is the computed parity bit (computed by the program) for bits 18–35.

2.5.3.7 DCR (RS = 77) Data – The diagnostic control register is loaded only during diagnostic operations.

a. TRA [TT] (bit 26) – Simulates the TRA (transfer) signal normally asserted by a drive on the Massbus.

b. CB TEST [CBT] (bit 27) – Loops back the data in the Massbus buffer register (loaded by an external register write operation or command file transfer) via the Massbus transmitters and receivers so that the data may be read and verified by an external register read operation.

c. BAR TEST [BAR] (bit 28) – Inhibits the controller from reading the contents of the STCR during a command file transfer. Instead, the SBAR is read twice and its contents are left in the Massbus buffer register at the end of the command file transfer. The data may then be verified by an external register read operation.

d. EVEN PARITY CHECK [EPC] (bit 29) – Forces CBEP (DATAI, RS = $00–37_8$: bit 08) and RAE (CONI: bit 24) when normal (odd) parity is detected on the Massbus (control bus) data lines.

e. READ/WRITE [RW] (bit 30) – Set during a diagnostic read data transfer to loop back the simulated read data in the WR via the Massbus transmitters and receivers. Cleared during a diagnostic write operation.

f. EXC [E] (bit 31) – Simulates the EXC (exception) signal normally asserted by a drive on the Massbus.

g. EBL [EBL] (bit 32) – Simulates the EBL (end of block) signal normally asserted by a drive on the Massbus.

h. ATTN [A] (bit 34) – Simulates the ATTN (attention) signal normally asserted by a drive on the Massbus.

i. SCLK [C] (bit 35) – Simulates the SCLK (sync clock) signal normally asserted by a drive on the Massbus.

2.5.4 DATAI

The DATAI command is used to read both internal and external registers. The register that is read is determined by the address and control data stored in the preparation register by a previous DATAO. Bit format for the DATAI command is shown in Figure 2-20.

a. REGISTER SELECT [RS] (bits 00–05) – The register select code stored in the preparation register.

b. LOAD REGISTER [LR] (bit 06) – Indicates the state of the LR bit stored in the preparation register. Equals 0 when the previous DATAO loaded only the preparation register. Equals 1 when the DATAO wrote the specified register.

2.5.4.1 External Register (RS = $00–37_8$) Data

a. CONTROL BUS PAR ERR [CBPE] (bit 08) – Indicates that bad parity was detected on the Massbus (control bus) data lines during the DATAI. Sets RAE (CONI: bit 24).

b. DRAES [DRAES] (bit 09) – Indicates that register access errors have been disabled. Set by DATAO: bit 09 = 1.

c. TRANSFER RECEIVED [TRA] – Indicates that a drive responded (asserted transfer on the Massbus) during the DATAI.

d. DRIVE SELECT [DS] (bits 15–17) – The drive select code stored in the preparation register by a DATAO.

e. CONTROL BUS PARITY BIT [CPA] – Indicates the state of the parity bit received on the Massbus (control bus).

f. EXTERNAL REGISTER DATA [ERD] (bits 20–35) – The contents of the register addressed by the DATAI.

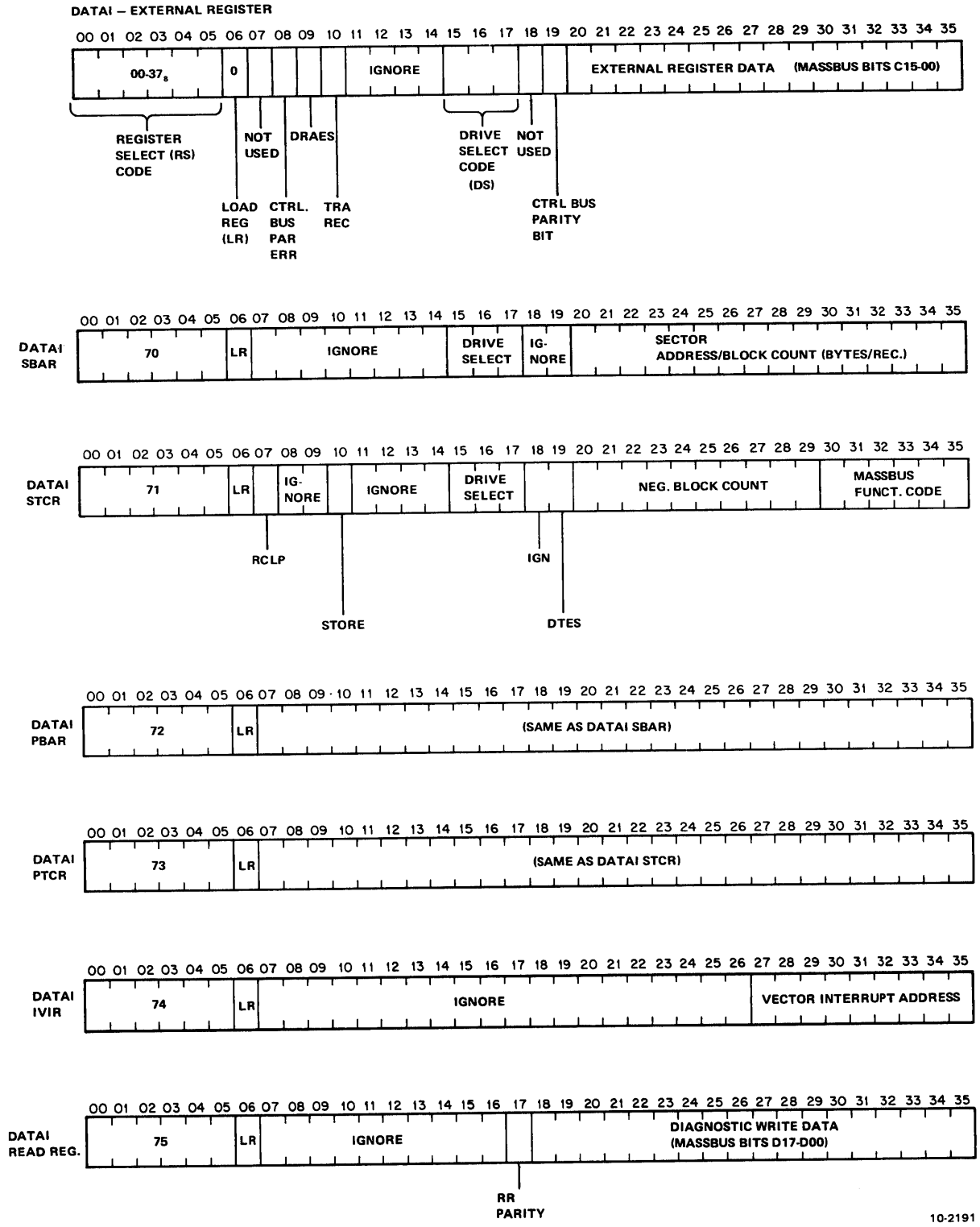


Figure 2-20 DATAI Command Format

2.5.4.2 SBAR (RS = 70) and PBAR (RS = 72) Data

a. DRIVE SELECT [DS] (bits 15–17) – The drive select code stored in the command file by the DATAO: bits 15–17.

b. BLOCK ADDRESS/FRAME COUNT [DBA] (bits 20–35) – The starting block address (disks) or frame count (tapes) stored in the command file by the DATAO: bit 20–35.

2.5.4.3 STCR (RS = 71) and PTCR (RS = 73) Data

a. RCLP [RCLP] (bit 07) – The channel initialization control bit loaded in the command file by a DATAO: bit 07.

b. STORE [SES] (bit 10) – The store channel status control bit loaded in the command file by a DATAO: bit 10.

c. DRIVE SELECT [DS] (bit 15–17) – The drive select code stored in the command file location by a DATAO: bits 15–17.

d. DTES [DTES] (bit 19) – The disable transfer error stop bit loaded in the command file by the DATAO: bit 19.

e. NEGATIVE BLOCK COUNT [NBC] (bits 20–29) – The data block count loaded in the command file by a DATAO: bits 20–29.

f. MASSBUS FUNCTION CODE [MFC] (bits 30–35) – The command code loaded in the command file by a DATAO: bits 30–35.

2.5.4.4 IVIR (RS = 74) Data

a. INTERRUPT VECTOR ADDRESS [IVR] (bits 27–35) – The interrupt address loaded by a DATAO: bits 27–35.

2.5.4.5 RR (RS = 75) Data – Read by the program during diagnostic operations.

a. DIAGNOSTIC WRITE DATA [D] (bits 17–35) – Write data and parity (bit 17) looped back via the Massbus transmitters and receivers during a diagnostic write data transfer.

2.6 CHANNEL COMMANDS

To control MBox data transfers, an active channel fetches and executes the appropriate channel commands from a command list previously set up in memory by the program. A command list pointer in the channel determines the memory address from which the commands (control words) are fetched. The pointer can be reset to a preassigned address when the RH20 starts the channel at the beginning of a data transfer. This initial address is the address of word 0 in the channel's reset and status logout area (Subsection 2.2).

The three basic channel commands are:

a. DATA TRANSFER
b. JUMP
c. HALT

The DATA TRANSFER command initiates a data transfer. It also supplies a data buffer address for the transfer and a data transfer length (word count). When fetched, the DATA TRANSFER command increments the command list pointer.

The JUMP command loads a new address in the command list pointer, thus allowing a branch during command list execution. For example, the program normally loads a JUMP in location 0 of the channel's reset and logout area to point to the first location in the command list.

The HALT command causes channel operations to stop; that is, the next sequential control word in the command list is not fetched. (A DATA TRANSFER CCW can also be made to halt channel operation.) The HALT command also loads a new address in the command list pointer, thus providing a starting address for the next channel command list if the RH20 does not reset the channel at the start of the next data transfer.

Channel command word format is shown in Figure 2-21.

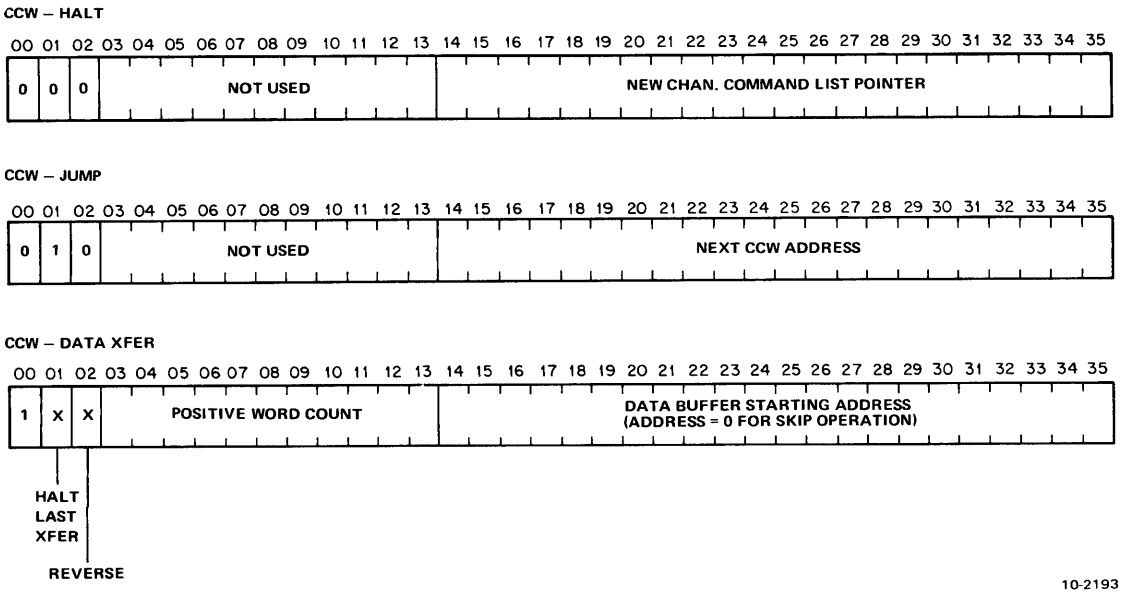


Figure 2-21 Channel Command Word Format

a. OP CODE (bits 00–02) – Specifies the function of the command. Bit 01 is further defined as HALT LAST XFER; bit 02 as REVERSE. (Refer to codes 4–7_8.) The HALT LAST XFER bit stops execution of the command list; the REVERSE bit causes the normally incremented data buffer address to be decremented during a data transfer.

   0_8 – Causes halt in command list execution (HALT command).

   2_8 – Causes branch in command list execution (JUMP command).

   4_8 – Causes a forward data transfer (device read or write) without halting (DATA TRANSFER command).

   5_8 – Causes a reverse data transfer (device read only) without halting (DATA TRANSFER command).

   6_8 – Causes a forward data transfer (device read or write) and a halt ("LAST" DATA TRANSFER command).

   7_8 – Causes a reverse data transfer (device read only) and a halt ("LAST" DATA TRANSFER command).

b. WORD COUNT field (bits 03–13) – Applicable only to DATA TRANSFER commands. A positive word count that specifies the number of 36-bit words to be transferred. The word count is stored in the channel and down-counted during the data transfer. When the word count goes to zero, the next command is fetched if bit 01 = 0. If bit 01 = 1, the channel halts.

c. ADDRESS field (bits 14–35) – The 22-bit physical address of the first data buffer location in memory. If bit 02 = 0, the address is incremented each time a word is moved to/from memory. If bit 02 = 1, the address is decremented during the data transfer to allow read reverse operations on magnetic tape systems. (The program must set bit 02 to zero for all device write operations.) If the ADDRESS field equals 0, it causes a skip operation. A skip during a device read operation causes the channel not to move data to memory. A skip during a device write operation causes the channel to supply full words of block fill data from EPT locations $60–63_8$ to write the remainder of a drive's data block. The skip function allows "gather-read" and "scatter-write" operations when chaining DATA TRANSFER commands in the command list.

2.7 CHANNEL STATUS

A channel, upon detecting an error or after receiving CBUS STORE from the RH20, will store two status words in words 1 and 2 of the channel's assigned reset and status logout area (Subsection 2.2). Status word bit format is shown in Figure 2-22.

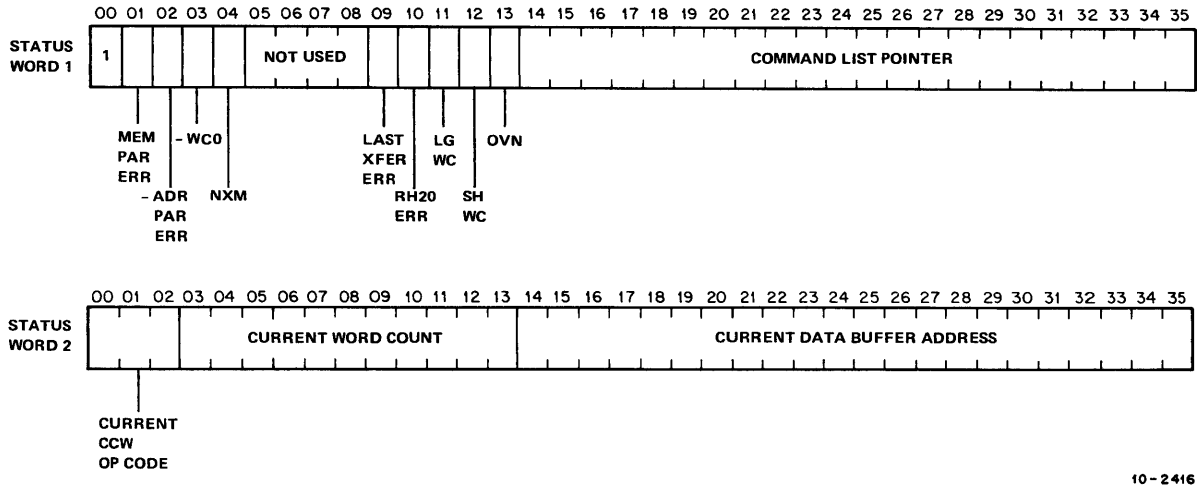


Figure 2-22 Channel Status Word Format

2.7.1 Status Word 1

a. MEM PAR ERROR (bit 01) – Indicates that a memory parity error was detected during a control word fetch.

b. NOT ADR PAR ERROR (bit 02) – Indicates that incorrect memory address (and request) line parity was not detected during a control word fetch.

c. NOT WORD COUNT = 0 (bit 03) – Indicates that the channel word count did not equal 0 when the status words were stored in the EPT.

d. NXM (bit 04) – Indicates the channel referenced a nonexistent memory address (no memory response).

e. LAST TRANSFER ERROR (bit 09) – Indicates that an error was detected after the RH20 terminated a data transfer. The channel will abort the next data transfer (if any) initiated by the RH20.

f. RH20 ERROR (bit 10) – Indicates that the RH20 attempted to start the channel and the channel was not prepared to start. Caused by a hardware malfunction.

g. LONG WORD COUNT ERROR (bit 11) – Indicates that the RH20 completed a data transfer and the channel did not reach the final word count specified in the CCW.

h. SHORT WORD COUNT ERROR (bit 12) – Indicates that the channel transferred all the data specified by the CCW and the RH20 was not through transferring data.

i. OVERRUN (bit 13) – Indicates that the RH20 sent data to the channel (device read) and the channel data buffers were full or that the RH20 requested data from the channel (device write) and the channel data buffers were empty.

j. COMMAND LIST POINTER (bits 14–35) – Indicates the address of the current CCW + 1.

2.7.2 Status Word 2

a. OP CODE (bits 00–02) – The operation (function) code of the current CCW.

b. CURRENT WORD COUNT (bits 03–13) – Indicates the current channel word count. Should be 0 at the end of a normal data transfer.

c. CURRENT DATA BUFFER ADDRESS (bits 14–35) – Indicates the address of the last data word transferred to/from memory.

2.8 PROGRAMMING THE RH20

The RH20 instruction set is described in Subsection 2.5.

2.8.1 Control Data Write/Read

The CONO is used to load miscellaneous control data; that is, load the PIA, clear errors, dismiss interrupts, etc. The CONI is used to read control bits set by the CONO plus RH20 error and interrupt status. All CONI bits, except CHAN RDY (bit 25), are cleared during system initialization or by a CONO: CLR MBC (bit 25) = 1. The CHAN RDY bit is set by system initialization.

2.8.2 Register Write/Read

The DATAO, by setting LR (bit 06) = 1, is used to write internal registers (RS 00–37_8) and external registers (RS = 70–77_8). By setting LR = 0, it is also used to load a register address (plus control data and a drive address if RS = 00–37_8) so that a register may be read by a subsequent DATAI. Repeated DATAI commands will read the same register; that is, the DATAI will always read the register specified by the last DATAO (LR = 0 or 1). A DATAO attempting to write a read-only internal register or an unspecified register (RS = 40–67_8) loads no data nor causes an error. Similarly, a DATAI reading a write-only internal register or an unspecified register reads no data nor causes an error.

When the RH20 detects an error (nonexistent drive or control bus parity) during the writing or reading of an external register, RAE (CONI: bit 24) will be set. This error generates an interrupt and inhibits the program from writing any more registers (internal or external) with a DATAO unless DRAES (DATAI: bit 09) was set by the DATAO that preceded or caused the error. (A DATAI is not inhibited and it can be used to read back the failing register and drive address.) RAE may be cleared by a CONO (bit 24) to dismiss the RH20 interrupt.

Drive errors resulting from the writing or reading of external registers assert ATTN (CONI: bit 28) and generate an RH20 interrupt if ATTN INT ENA (CONO: bit 30) is set. The RH20 interrupt may be dismissed by clearing ATTN INT ENA but ATTN can only be cleared by clearing the ATA bit in the appropriate drive(s). The program must first read the attention summary register (RS = 04, an external register address) to determine the interrupting drive.

ATTN is also asserted at the completion of non-data transfer commands that are loaded in the drive's control register (Subsection 2.4). For example, at the completion of a SEEK command issued to a moving-head disk, ATTN (together with the appropriate drive status bit) indicates that the heads are over the specified cylinder address. A read/write data transfer may then be initiated in the drive.

2.8.3 Write/Read Data Transfer

To start a data transfer, the program first sets up the command list in memory (Subsection 2.6). It then loads the SBAR (optional) and the STCR (mandatory) with a DATAO (RS = 70_8 and 71_8). The SBAR holds the starting track and sector address (disks) or frame count (tapes). The STCR holds the negative block count, the drive address, the drive command code (Subsection 2.4), and the RCLP, DTES, and STORE control bits. Only a read/write command code shall be loaded in the STCR. (Non-data transfer commands are written in the drive's control register directly by a DATAO.) Also, only one record shall be specified by the negative block count when transferring data to/from magnetic tape.

During normal operation, the command list word count should specify the same transfer length as that specified by the block count in the STCR; that is, the channel word count shall be equal to n times the block count, where n is equal to the block size of the drive. If the channel word count and RH20 block count do not correspond, transfer errors (LONG WC or SHORT WC) will be generated during the read/write.

If a data transfer is not already in progress, loading the STCR causes the contents of the SBAR (if loaded by the program) to be written in the drive's block address or frame count register (RS = 05_8) by the RH20. (The SBAR does not have to be loaded by the program when the drive's block address register, which is incremented during a read/write, has been left pointing to the desired surface address by the previous data transfer.) Immediately following the transfer of the SBAR contents to the selected drive, or following the load of the STCR if the SBAR had not been loaded by the program, the command code in the STCR is transferred to the drive's control register (RS = 00) to start the data transfer in the drive. At the same time the channel is started and its command list pointer is reset if RCLP = 1.

Once a data transfer is active, the STCR is flagged as empty (CONI: bit 29 = 0) and the program may reload the SBAR and STCR to specify the next transfer. The RH20 will then automatically address and load the drive register(s) to initiate the "backup" data transfer immediately after the current read/write operation ends. This allows data transfers to be initiated during the intersector gap of some devices and the next sector can be accessed by the "backup" command with minimum latency; that is, without having to wait a full disk revolution. To increase the probability of next sector read or write, the last control word in a command list should not be a HALT CCW. It should be a DATA XFER CCW with HALT LAST XFER (bit 01) = 1. Also, the first control word for the next transfer should also be a DATA XFER CCW; that is, the command list pointer should not be reset and the control word should be in the next sequential memory location following the last control word of the last transfer.

CMD DONE (CONI: bit 32) sets to generate an RH20 interrupt when a data transfer ends. If termination is due to a transfer error, an error flag will be set (CONI: bits 18–23, 26) together with CMD DONE. Errors detected in the channel are flagged by CHAN ERR (bit 22). Drive errors assert ATTN (bit 28) in addition to DR EXC (bit 19). A transfer error automatically causes channel status (Subsection 2.7) to be stored at the end of the transfer. Channel status is also stored if the program had set the STORE control bit in the STCR when specifying the read/write.

If the DTES bit is set in the STCR, data parity errors detected in the RH20 and class A errors detected in a drive are prevented from shutting down the transfer. (A class A drive error is defined as an error that does not automatically abort a read/write operation.) The DTES bit is set by the program during error recovery operations to allow as much data as possible to be retrieved from a failing disk or tape.

If a data transfer is shut down by a transfer error when the STCR is holding a "backup" read/write command, the "backup" command is not initiated until the transfer error condition has been cleared by the program (CONO: bit 26). If desired, the "backup" command may be deleted from the STCR (CONO: bit 29) before clearing the error.

SECTION 3
LOGIC DESCRIPTION

This section describes RH20 operation at the logic level. Refer to the Field Maintenance Print Set (M8555, M8556, and M8557 modules) and to the detailed block diagram shown in Figure 3-1. The RH20 circuitry is divided into three major categories:

a. Asynchronous Data Transfer Control Logic

b. Synchronous Data Transfer Control Logic

c. Diagnostic Data Transfer Control Logic.

3.1 ASYNCHRONOUS DATA TRANSFER CONTROL LOGIC

The asynchronous data transfer control logic interfaces the RH20 to the EBus and to the control bus portion of the Massbus. It contains the logic elements necessary to generate PI requests, execute EBus commands, and initiate command file transfers.

3.1.1 EBus Function Decoder and Address Comparators

The function decoder is a 4 to 10 line decoder that decodes the binary value of the EBus function lines (EBI7 F00–02) connected to its three low-order inputs. Depending on the encoded function code (0–5), one decoder output (0–5) is asserted at the beginning of each EBus operation. For example, a function code of 4 (PI SERVED) asserts output 4 (EBI1 PI SERVED ENA).

Four 4-bit comparator circuits are used to compare the EBus controller select lines with the hard-wired device code, the hard-wired physical address, and the interrupting PI channel number. Two comparators perform the device code check. The first asserts EBI1 DC COMP CRY when the four high-order select lines match the corresponding bits in the hard-wired device code address (i.e., EBI7 CS00–03 = 1011). The comparator is enabled to check for equality by the negation of function decoder output EBI1 PI SERVED ENA at the comparator's A = B input. PI SERVED ENA inhibits DC COMP CRY during a PI SERVED operation, one of the two EBus commands that does not address a controller by its device code.

Once asserted, DC COMP CRY enables the A = B input to a second comparator circuit, allowing it to check for a match between the other three select lines and the corresponding hard-wired address bits. If equal when DEMAND is asserted on the EBus (i.e., EBI7 CS04–06 = xxx, xxx = 000–111), the second comparator asserts DC COMP if function decoder output EBI1 ADR IN ENA is false. (EBI7 DEMAND and – EBI1 ADR IN ENA connect to the comparator's high order inputs.) ADR IN ENA inhibits DC COMP during the PI ADR IN command, the second of the two EBus instructions that do not address a controller by means of device code.

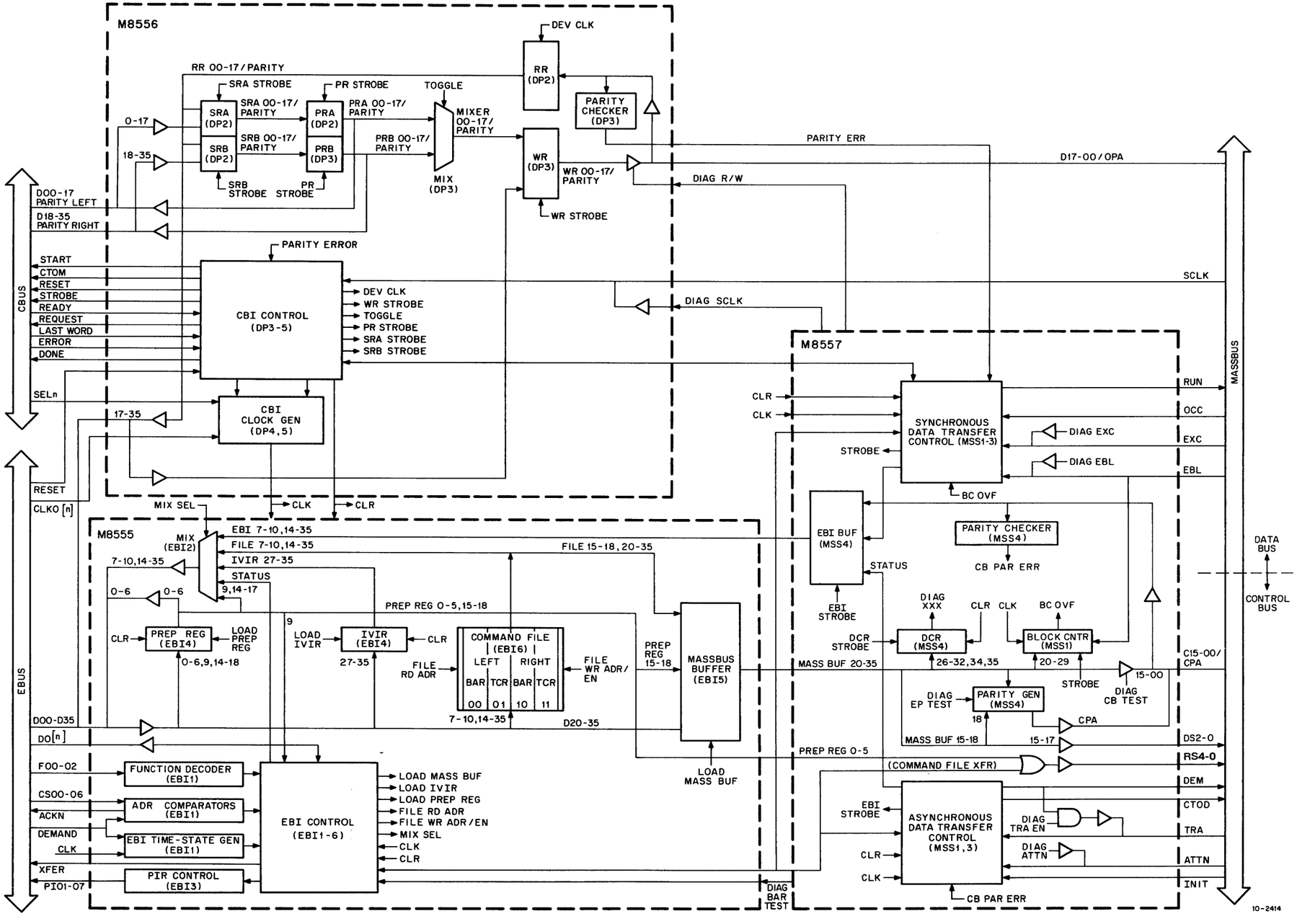


Figure 3-1 RH20 Detailed Block Diagram

With DEMAND asserted, DC COMP starts the execution of EBus commands addressing the RH20 by its device code (i.e., CONO, DATAO, CONI, DATAI). It asserts EBI1 ACKN, causing the ACKN signal to be transmitted on the EBus, and it is gated at the clock input to flip-flops EBI5 BAR TO MASSBUS and EBI5 TCR TO MASSBUS to prevent a command file transfer from starting during the EBus command. Also, DC COMP enables the first EBI time-state generator output, EBI1 T0, causing it to be asserted by the next controller clock. The active time-state generator then sequences RH20 operation during the command.

During the PI SERVED and PI ADR IN commands, a third comparator circuit asserts EBI1 INTR COMP when the controller select lines match the interrupting PI channel number (i.e., CS04–06 = xxx, xxx = 001–111). The PI channel (PIA) is specified by flip-flops EBI4 PIA33–35, set previously by the CONO instruction. The PIA flip-flops connect to the comparator's input together with the negation of EBI INTR COND. This signal is asserted, allowing INTR COMP to go true, only when conditions are met for initiating an interrupt request. Another condition for asserting INTR COMP is that the PIA cannot equal 0. (Setting the PIA to 0 in the RH20 allows the programmer to inhibit interrupt requests to the CPU.) This is ensured by the negation of EBI3 PIR 00, connected to the comparator's A = B input and asserted when the PIA flip-flops are cleared. PIR00 inhibits INTR COMP (and an RH20 response) when a PI SERVED command is polling devices that do interrupt on channel 0 (e.g., DTE20).

The fourth comparator circuit asserts EBI1 PHY COMP during the PI ADR IN command when the controller select lines match the hard-wired physical address (i.e., EBI7 CS00–03 = xxxx, xxxx = 0000–0111). Like the comparator that asserts INTR COMP, the circuit is enabled to check for equality by the negation of EBI3 PIR00 at its A = B input. This inhibits PHY COMP if the PIA = 0.

3.1.2 EBI Time-State Generator

Three D-type flip-flops (EBI1 T0-T2), cleared by DEMAND and connected as a shift register, serve as a time-state generator (and synchronizer) to sequence EBus commands that address the RH20 by its device code. At the start of CONO, DATAO, CONI and DATAI commands, device code comparator output DC COMP enables the first flip-flop, T0, allowing it to be set by the first controller clock following the assertion of DEMAND. T1 and T2 are then set, one after the other, by successive controller clocks. The time-state generator outputs are ANDed with the outputs from the function decoder (Subsection 3.1.1) to control RH20 operation. T0–T2 remain set until DEMAND is negated by the EBox at the end of the operation.

3.1.3 Control Data Load (CONO)

RH20 control information is loaded by the CONO command. A flow diagram for the CONO operation (as pertains to the RH20) is shown in Figure 3-2. A timing diagram is shown in Figure 3-3. Initially, the EBox asserts the control data for the RH20 (bit format shown in Figure 2-17) on the EBus data lines (EBUS D24–35), the function code for the CONO operation (0) on the function lines (EBUS F00–02), and the device code for the RH20 (540_8 – 574_8) on the controller select lines (EBUS CS00–06). The EBox then asserts EBUS DEMAND to signal that the CONO is to be executed.

In the RH20, the binary encoded value of the function lines causes EBI1 CONO ENA to be asserted by the function decoder (Subsection 3.1.1). When EBUS DEMAND is received, it allows EBI1 DC COMP to be generated. DC COMP is the output from the device code comparator (Subsection 3.1.1). It is asserted when the controller select lines match the hard-wired device code address.

DC COMP starts the CONO operation in the RH20 by asserting EBI1 ACKN, which causes EBUS ACKN to be transmitted on the EBus; and by enabling the EBI time-state generator, EBI1 T0–T2. The EBI time-state generator (Subsection 3.1.2) sequences the execution of EBus commands that address the RH20 by its device code.

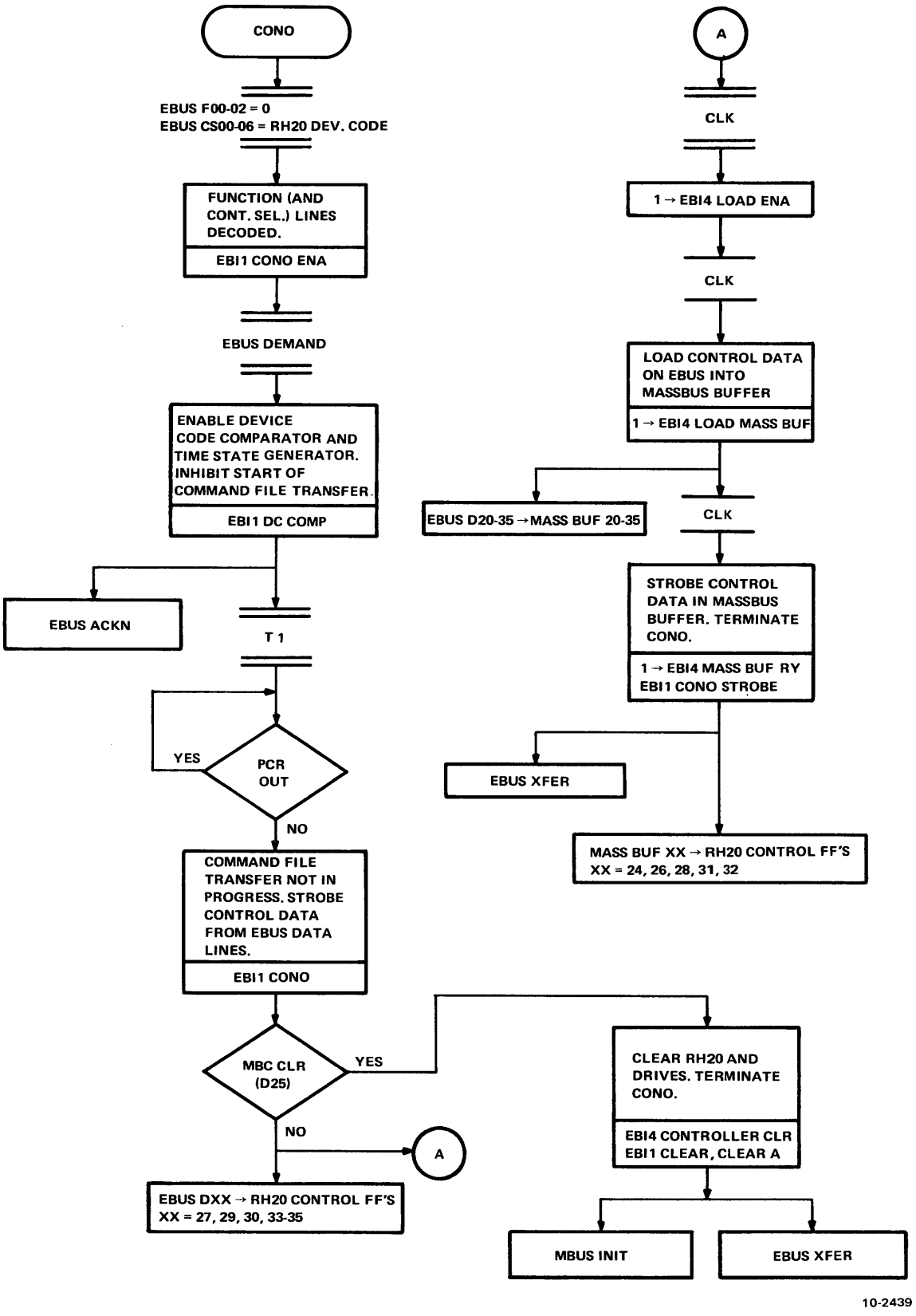


Figure 3-2 CONO Flow Diagram

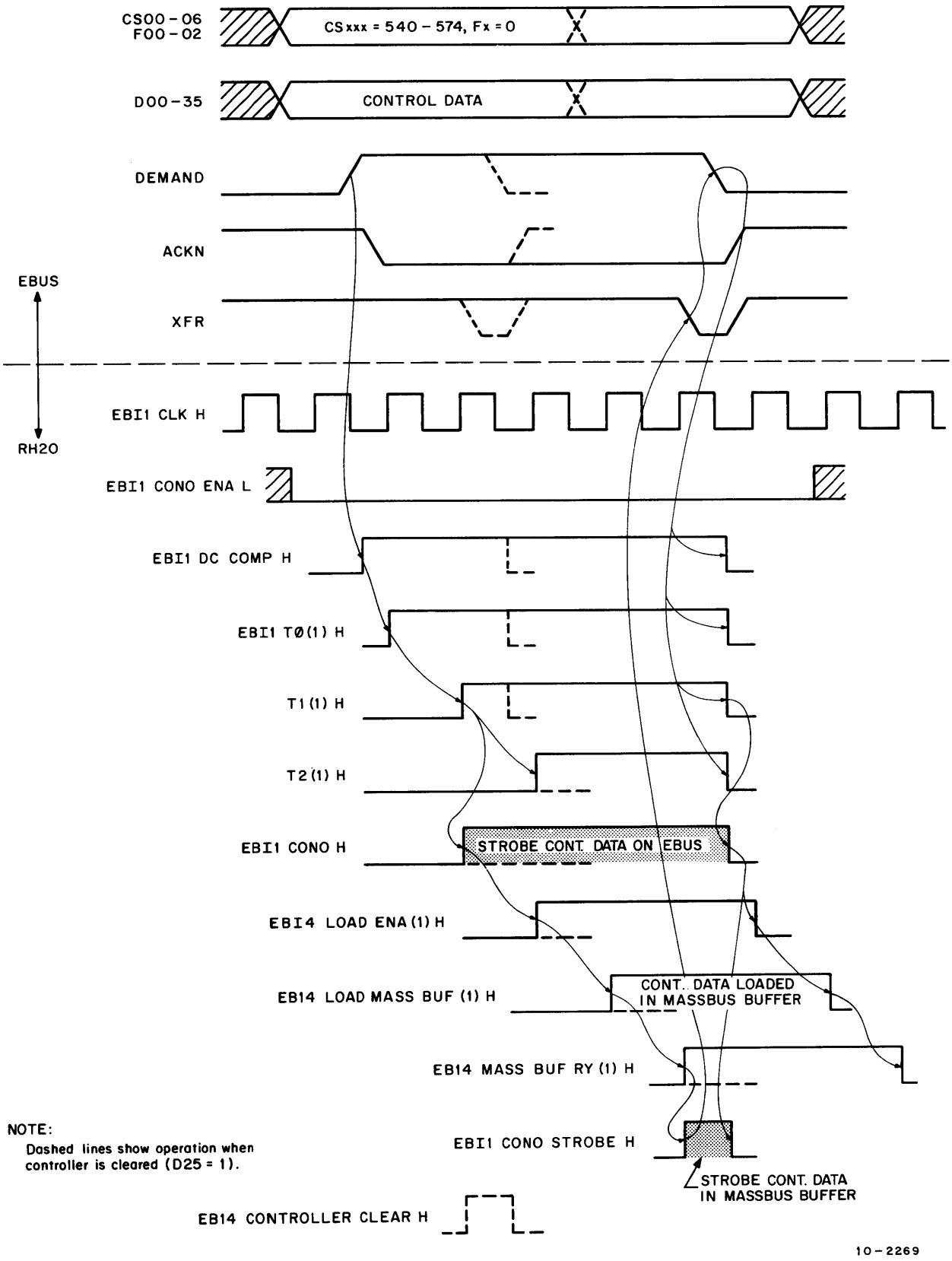


Figure 3-3 CONO Timing Diagram

DC COMP also inhibits the start of command file transfers by disabling the clock input to flip-flops EBI5 BAR TO MASSBUS and EBI5 TCR TO MASSBUS. Command file transfers (Subsection 3.1.9) are inhibited because they are made via the Massbus buffer register, and this register is used to hold control data (loaded from the EBus) during the CONO

With the EBI time-state generator active and CONO ENA true, EBI1 CONO is asserted at time T1 provided that a command file transfer is not taking place; that is, if EBI5 PCR OUT = 0. (Although DC COMP prevents a command file transfer from starting, a transfer may be in progress at time T1.) If EBI5 PCR OUT is true, the assertion of EBI1 CONO is delayed until the command file transfer ends. Once asserted, at the leading edge of T1 or later, EBI1 CONO does the following if the CONTROLLER CLEAR control bit has not been asserted on the data lines (D25 = 0).

a. Loads the PIA from EBus data lines D33–35 into flip-flops EBI4 PIA 33–35.
b. Loads enable flip-flops EBI4 MASSBUS ENA and EBI4 MBUS ATTN ENA from data lines D27 and D30.
c. Asserts EBI4 DELETE SREG to clear the secondary command file if data line D29 = 1.

If the CONTROLLER CLEAR control bit asserted (D25 = 1), EBI1 CONO generates EBI4 CONTROLLER CLR, which asserts EBI1 CLEAR and EBI1 CLEAR A. These three clear signals initialize the RH20. Also, CLEAR A terminates the CONO operation by asserting EBI1 XFER, which generates EBUS XFER. When the EBox receives XFER, it drops DEMAND, causing DC COMP and the EBI time-state generator outputs in the RH20 to be negated. DC COMP going false negates EBUS ACKN. EBI1 T1 going false negates EBI1 CONO, EBUS XFER, and the controller clear signals. This ends the CONO operation in the RH20.

If CONTROLLER CLEAR is not asserted, the CONO operation does not end and the control data on the EBus data lines is loaded in the Massbus buffer. Three flip-flops, set in succession, control the loading and strobing of the data. EBI1 CONO enables the first, synchronizing flip-flop EBI4 LOAD ENA, causing it to be set by the next controller clock. EBI4 LOAD MASS BUF (enabled by LOAD ENA) then sets one clock period later. LOAD MASS BUF clocks the Massbus buffer register flip-flops, EBI5 MASS BUF 20–35, to load the control data from the data lines. After another clock period, EBI4 MASS BUF RY sets (enabled to set by LOAD MASS BUF), asserting EBI1 CONO STROBE. With the Massbus buffer loaded, CONO STROBE does the following:

a. Asserts MSS3 RAE CLR to clear the register access error flag if MASS BUF 24 = 1.
b. Asserts MSS3 XFR ERR CLR to clear the transfer error flags if MASS BUF 26 = 1.
c. Clears MSS2 CMD DONE, the command done flag, if MASS BUF 32 = 1.
d. Sets MSS1 CONO RCLP ENA to reset the command list pointer if MASS BUF 28 = 1.
e. Sets MSS2 DONE TO MB ENA to stop a transfer if MASS BUF 31 = 1.
f. Asserts EBI1 XFER to transmit XFR on the EBus.

When the EBox receives XFR, it drops DEMAND. The negation of DEMAND then clears DC COMP and the EBI time-state generator outputs, which negates EBUS ACKN, EBI1 CONO, EBI1 CONO STROBE, and EBUS XFER. With EBI1 CONO false, LOAD EN, LOAD MASS BUF, and MASS BUF RY are cleared one after the other by successive controller clocks to end RH20 operation.

3.1.4 Control Status Read (CONI)

RH20 control and error status is read by the CONI command. Figure 3-4 illustrates sequence of operation for the RH20 during the CONI. Figure 3-5 shows RH20 timing. Initially, the EBox asserts the CONI function code (1) on the function lines (EBUS F00–02) and the RH20 device code (540_8–574_8) on the controller select lines (EBUS CS00–06). It then asserts EBUS DEMAND to initiate the CONI.

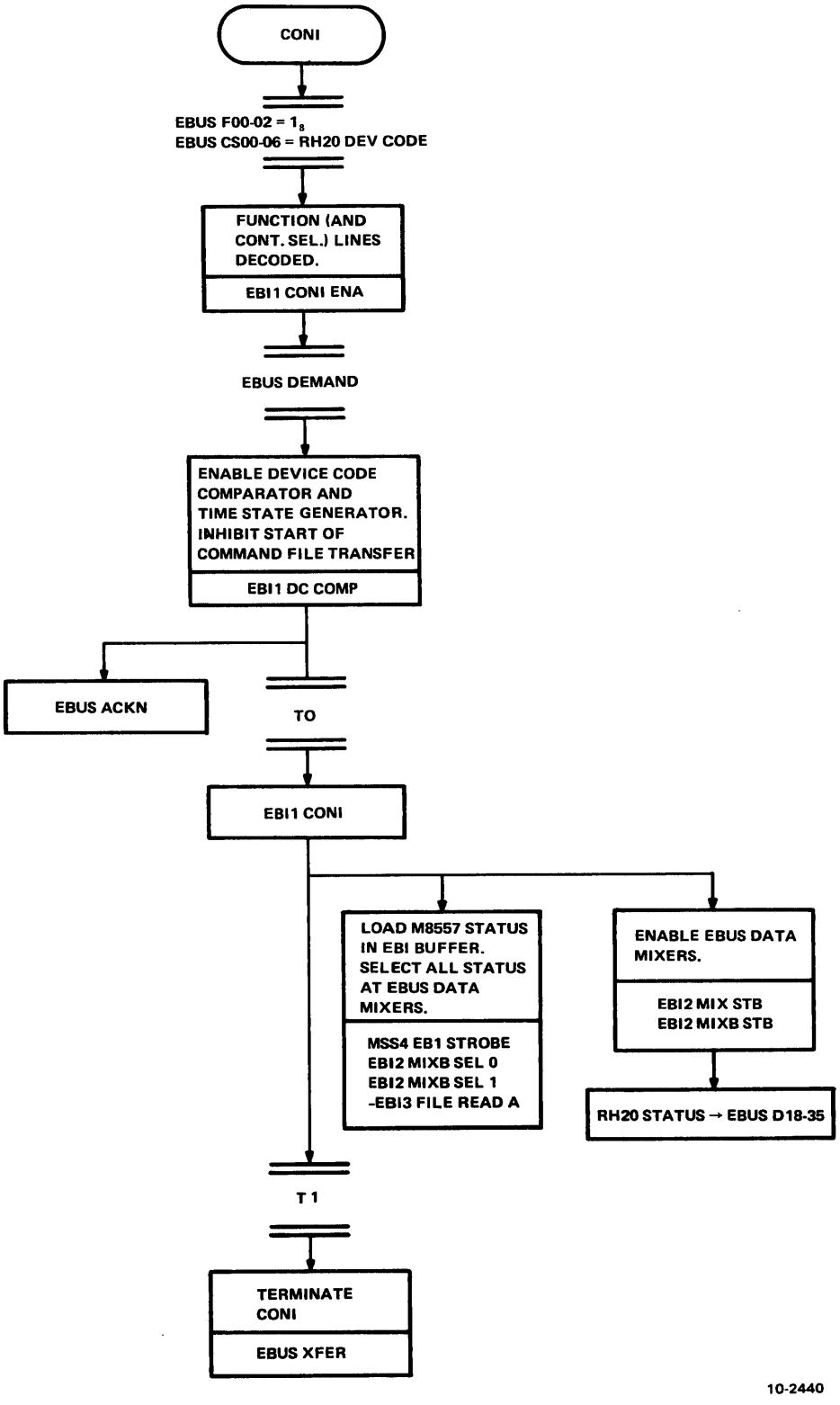


Figure 3-4 CONI Flow Diagram

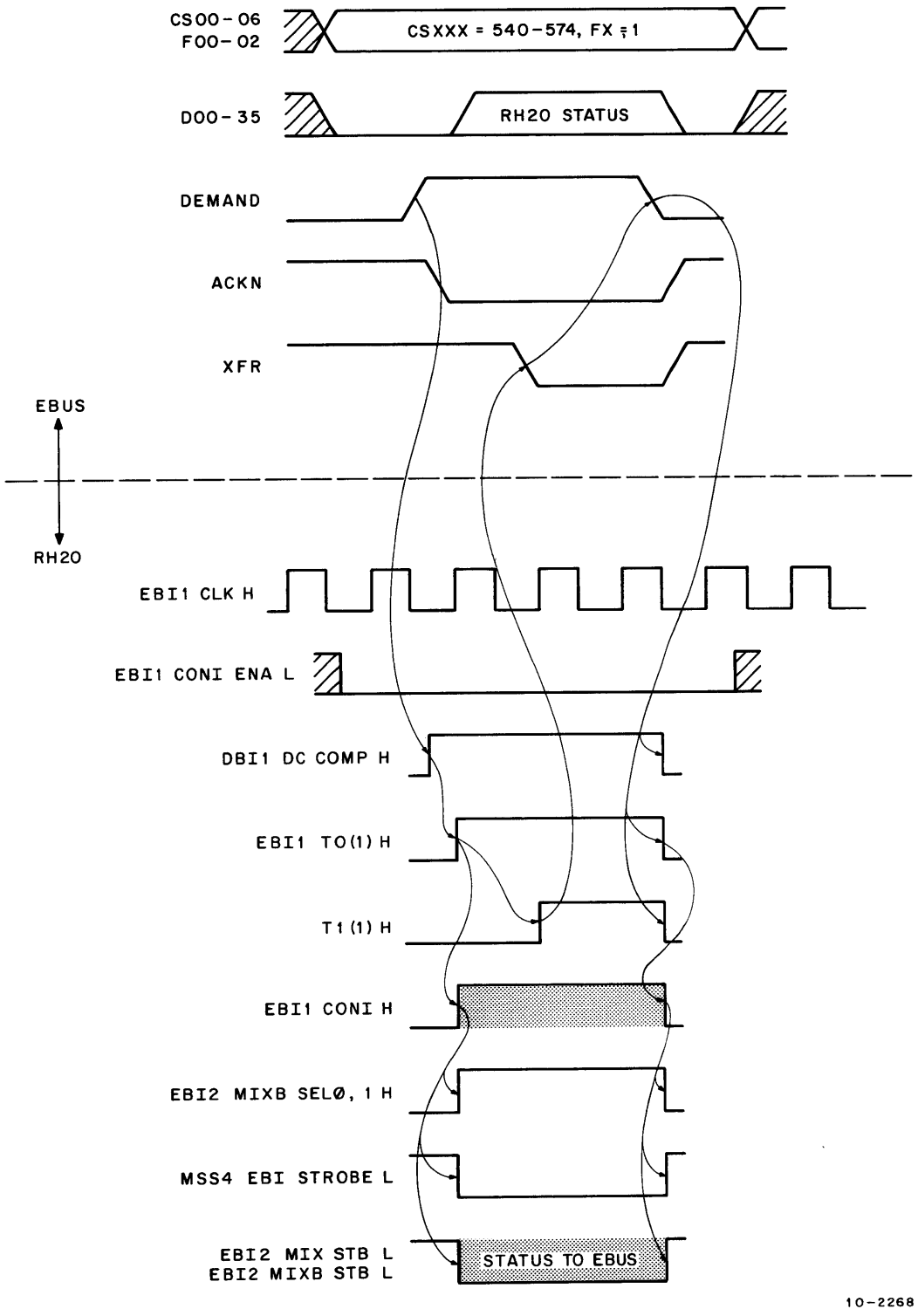


Figure 3-5 CONI Timing Diagram

In the RH20, the function line values assert function decoder output EBI1 CONI ENA. The DEMAND signal, when received, causes the device code comparator to assert EBI1 DC COMP. (The function decoder and device code comparator are discussed in Subsection 3.1.1.) DC COMP starts the execution of the CONI in the RH20 by generating EBI1 ACKN, which asserts ACKN on the EBus, and by enabling the EBI time-state generator (Subsection 3.1.2). With CONI ENA true, the first time-state generator output (T0) asserts EBI1 CONI. The CONI level does the following to select and gate RH20 status information onto the EBus data lines. (Status bit format is shown in Figure 2-18.)

a. Asserts MSS4 EBI STROBE to clock control status information (in the M8557 module) into the 10 low-order bit positions of the EBI buffer (MSS4 EBI 18–27). The EBI buffer outputs connect to the EBus data mixers.

b. Asserts EBI2 MIXB SEL0 and 1 to generate a select code of 3 at one group of EBus data mixers (EBI2 MIX 28–35). The rest of the data mixers (EBI2 MIX 18–27), which gate the status held in the EBI buffer, are selected by EBI3 FILE READ A. This signal is false during a CONI, generating a mixer select code of 0.

c. Asserts EBI2 MIX STB, which also asserts EBI2 MIXB STB, to enable the EBUS data mixers selecting status (EBI2 MIX 18–35). This causes the RH20 status information to be transmitted on the EBus data lines.

d. Asserts EBI1 XFER, when the second time-state generator output (T1) goes true, to transmit XFER on the EBus.

When XFER is received by the EBox, it strobes the status information from the EBus data lines and drops DEMAND. In the RH20, the negation of DEMAND clears DC COMP and the time-state generator outputs. DC COMP going false negates EBUS ACKN. T0 and T1 going false negate EBUS XFER and EBI1 CONI. The MIXB SEL levels, EBI STROBE, and MIX STB then go false, removing the controller status information from the EBus data lines and ending the CONI operation in the RH20.

3.1.5 Register Data Load (DATAO)

The DATAO command is used to load registers in the RH20 controller (defined as internal registers) and registers in the Massbus devices connected to the controllers (defined as external registers). It is also used to load a register address and other address and control information (in the preparation register) prior to reading a register with a DATAI command. Control bit format for the DATAO is shown in Figure 2-19.

A flow diagram showing RH20 operation during the DATAO is shown in Figure 3-6. Timing is shown in Figure 3-7. To start EBus operation, the EBox asserts the DATAO control data on the EBus data lines (EBUS D00–35), the DATAO function code (2) on the function lines (EBUS F00–02), and the RH20 device code (540_8–574_8) on the controller select lines (EBUS CS00–06). The EBox then asserts EBUS DEMAND.

In the RH20, the function line decoder (Subsection 3.3.3) asserts EBI1 DATAO ENA. The DEMAND signal, when received, causes EBI DC COMP to go true. DC COMP is the output of the device code comparator (Subsection 3.1.1). DC COMP asserts EBUS ACKN and it enables the EBI time-state generator (Subsection 3.1.2). DC COMP also inhibits the start of command file transfers by disabling the clock input to flip-flops EBI5 BAR TO MASSBUS and EBI5 TCR TO MASSBUS. Command file transfers (Subsection 3.1.9) are inhibited because file data is loaded in the Massbus buffer register and this register is used to hold control data during some DATAO commands; that is, when the DCR or an external register is addressed.

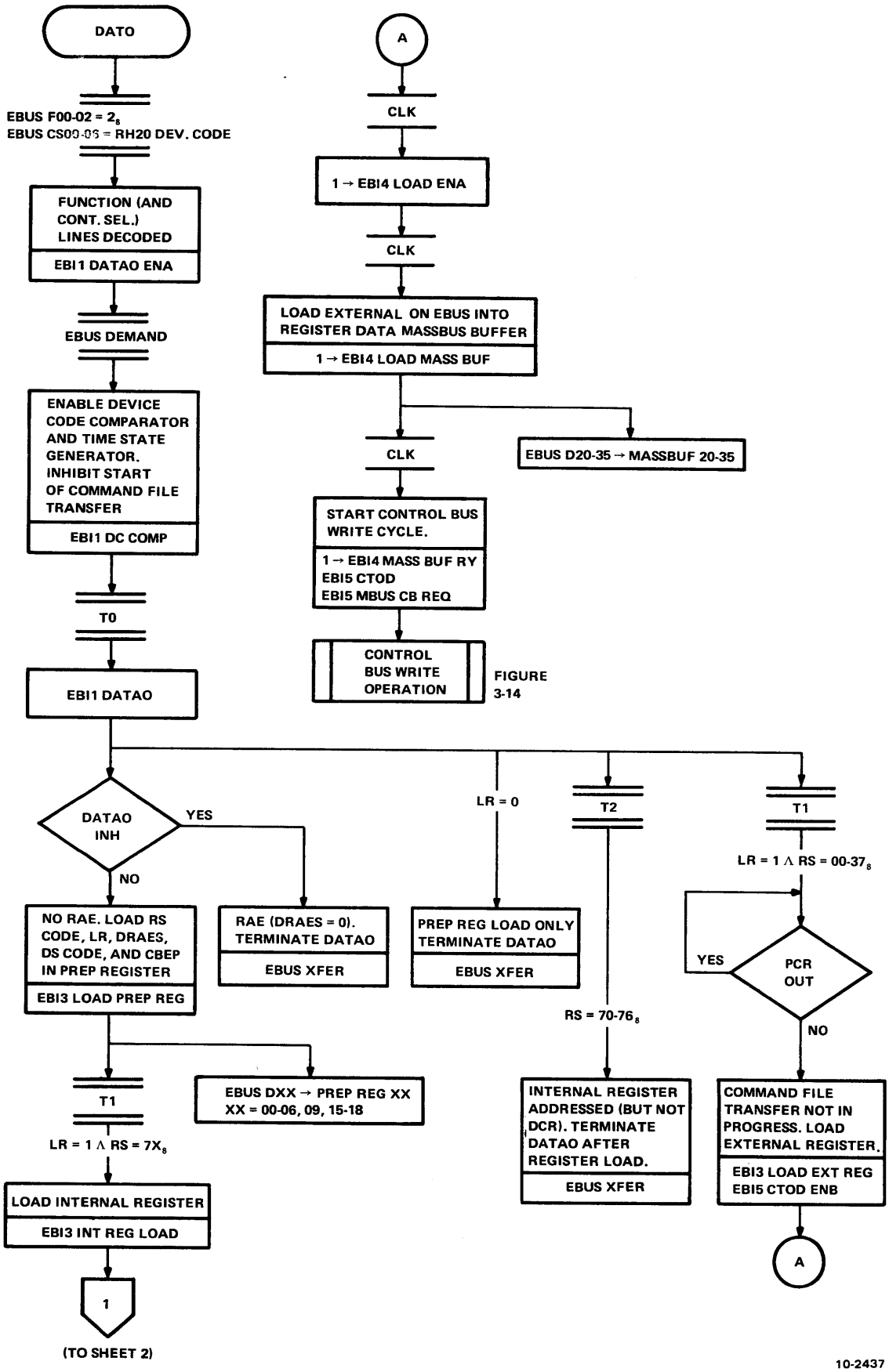


Figure 3-6 DATAO Flow Diagram (Sheet 1 of 2)

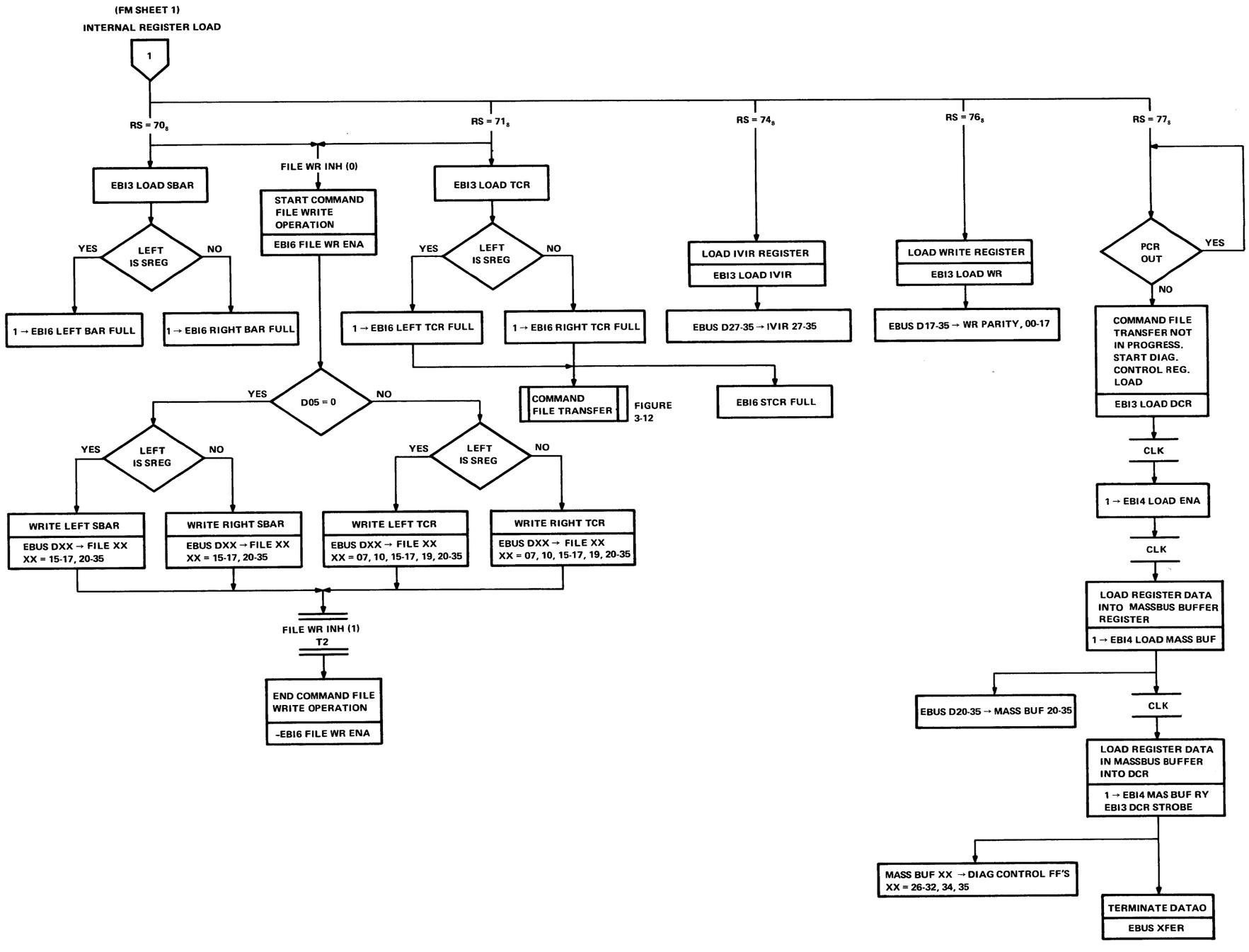


Figure 3-6 DATAO Flow Diagram
(Sheet 2 of 2)

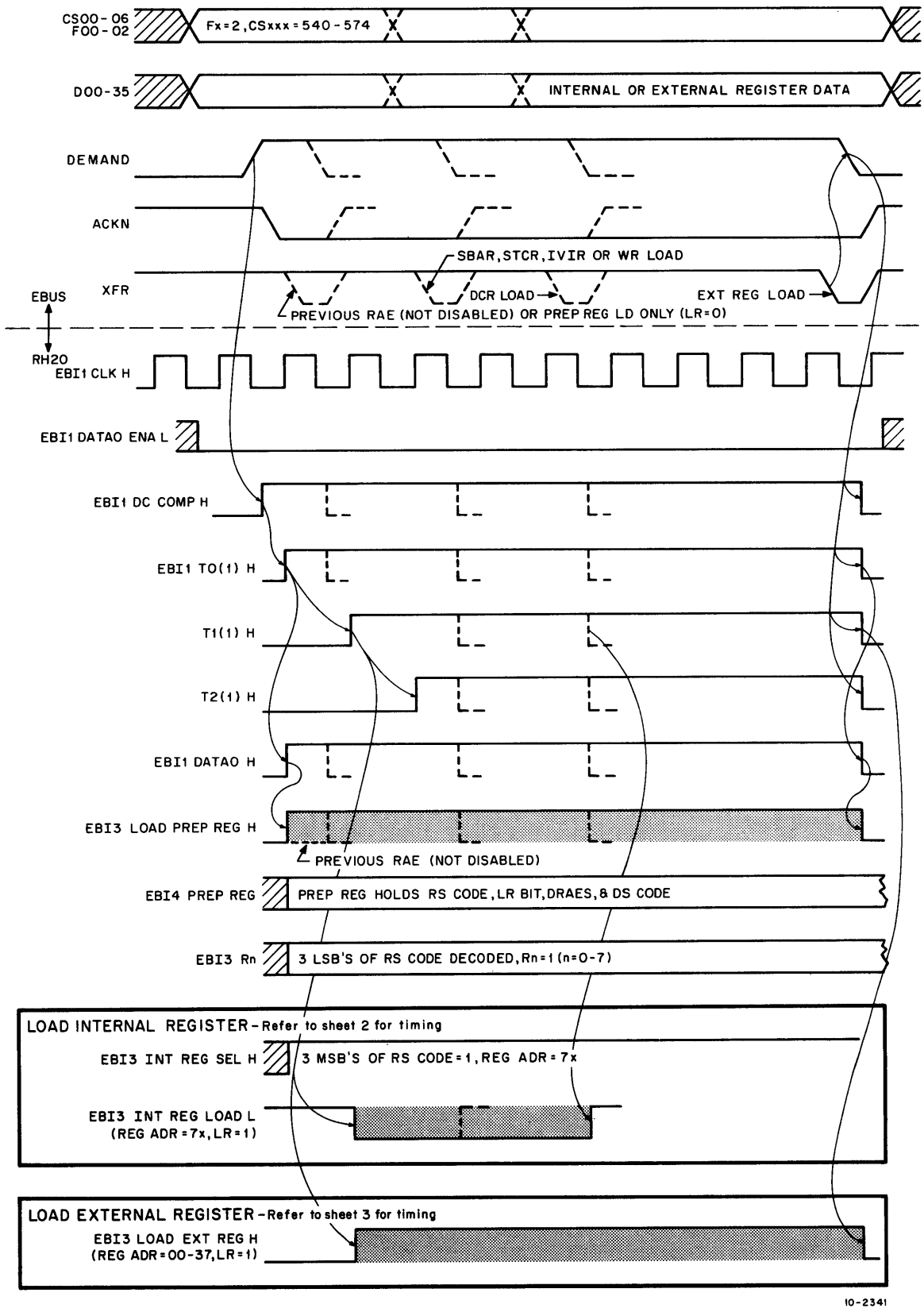


Figure 3-7 DATAO Timing Diagram (Sheet 1 of 3)

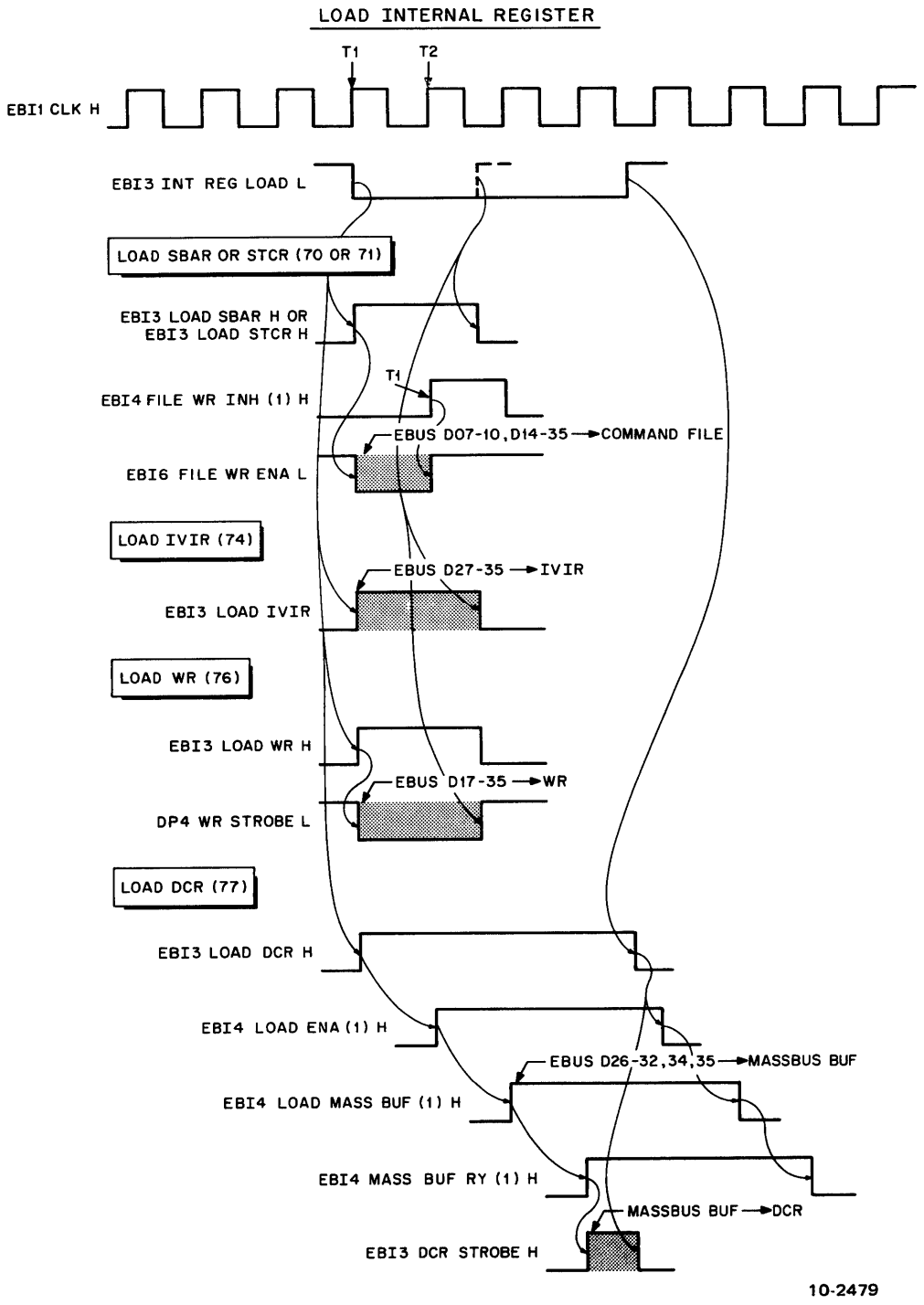


Figure 3-7 DATAO Timing Diagram (Sheet 2 of 3)

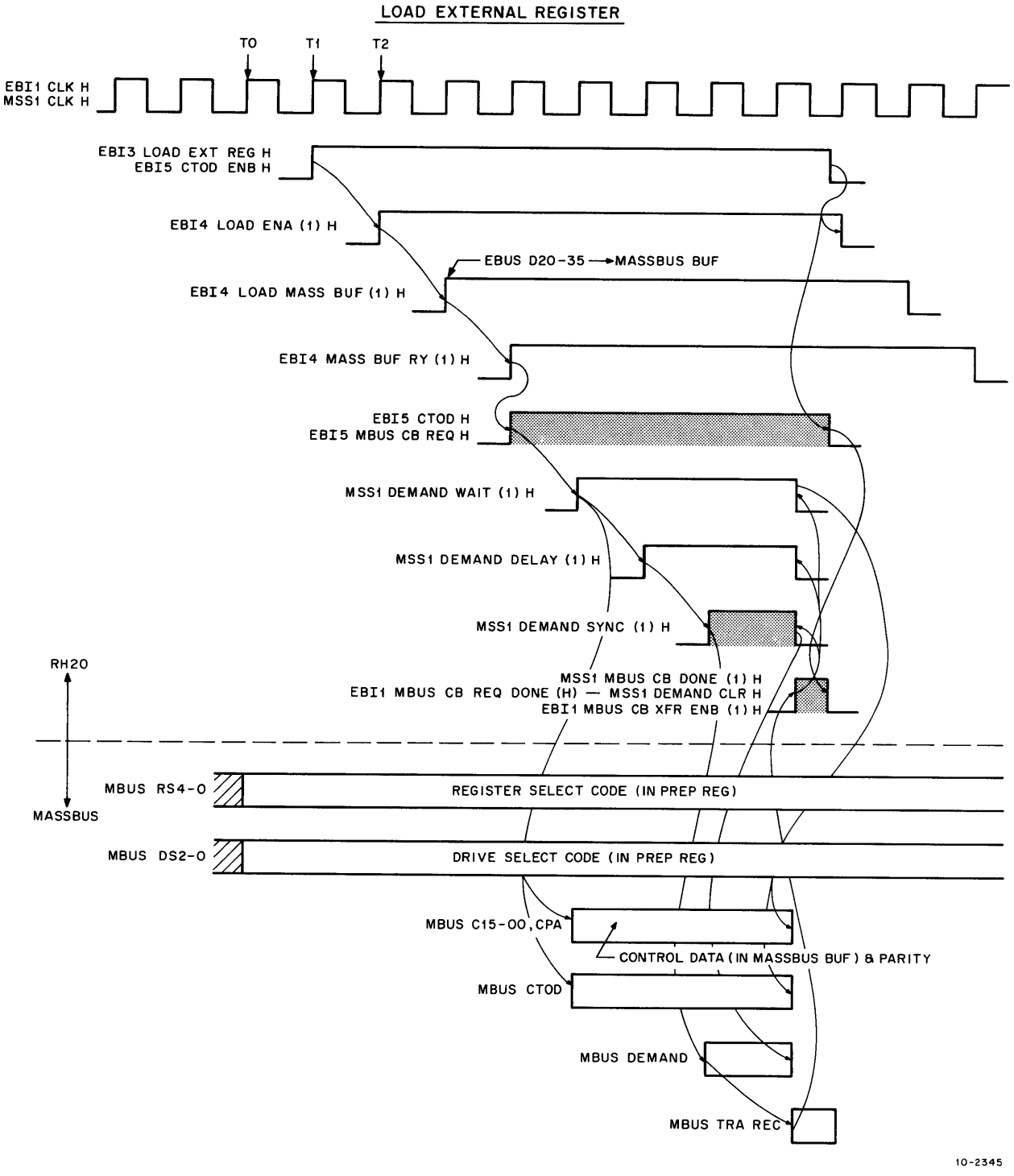


Figure 3-7 DATAO Timing Diagram (Sheet 3 of 3)

With DATAO EN true, EBI1 DATAO is asserted by the first time-state generator output (EBI1 T0). EBI1 DATAO does the following:

a. Asserts EBI3 LOAD PREP REG, if EBI1 DATAO INH is false, to clock the DATAO control data listed below into the preparation register.

   1. RS code (EBUS D00–05): loaded in EBI4 PREP REG 00–05.

   2. LR bit (EBUS D06): loaded in EBI4 PREP REG 06.

   3. DRAES bit (EBUS D09): loaded in EBI4 PREP REG 09.

   4. DS code (EBUS D15–17): loaded in EBI4 PREP REG 15–17.

   5. CBEP bit (EBUS D18): loaded in EBI4 PREP REG 18.

The RS code is the address of the internal or external register to be accessed. The LR (load register) bit is set to specify that the addressed register is to be loaded by the DATAO. The DRAES bit disables register access errors, occurring during the DATAO or a subsequent DATAI, from inhibiting future DATAO commands. The DS code is the drive address of the Massbus device containing the addressed external register. The CBEP bit, when set, causes even parity to be generated on the Massbus (control bus) when an external register is loaded. When not set, normal (odd) parity is written.

b. Asserts EBUS XFER, if EBI1 DATAO INH is true, to terminate the DATAO operation. DATAO INH is asserted if a register access error has been set (MSS3 RAE = 1) as a result of a previous DATAO, or DATAI, and if the error has not been disabled by the DRAES control bit (EBI4 PREP REG 09 = 0) loaded by a previous DATAO.

c. Asserts EBUS XFER to terminate the DATAO operation if the LR control bit is not asserted (EBUS D06 = 0). The LR bit is not asserted whenever the DATAO is to load only the preparation register (e.g., prior to a DATAI).

d. Asserts EBUS XFER to terminate the DATAO operation at time T2 if the register address is equal to $40–46_8$, $50–56_8$, $60–66_8$, and $70–76_8$ (EBUS D00 = 1, D03–05 $\neq$ 111). The assertion of XFER for this range of addresses terminates illegal references to undefined registers that have neither an external or an internal register address; and to internal registers that cannot be written (RS = 72, 73, 75). Its primary function, however, is to provide the normal termination when loading (legally) an internal register that can be written, other than the DCR. These registers (RS = 70, 71, 74, and 76) are loaded at the leading edge of time T1 and the DATAO can be terminated at time T2. The loading of the DCR (RS = 77) does not occur until after the leading edge of T2. Thus, a DATAO loading the DCR is not terminated at this time.

e. Asserts EBI3 LOAD EXT REG at time T01 to initiate an external register load sequence if the LR bit is set (EBI4 PREP REG 06 = 1) and if the RS code is equal to $00 - 37_8$ (EBI4 PREP REG 00 = 0), an external register address. Because the external register data is loaded in the Massbus buffer register and transferred over the Massbus, another condition for asserting LOAD EXT REG is that a command file transfer, which also uses the buffer and Massbus, cannot be in progress (EBI5 PCR OUT = 0). (Although the start of file transfers is inhibited by DC COMP, a transfer may be in progress at time T1.) If a transfer is active (PCR OUT = 1), the assertion of LOAD EXT REG is delayed until the transfer ends.

LOAD EXT REG initiates an external register load sequence by asserting EBI5 CTOD EN, which causes flip-flop EBI4 LOAD ENA to be set by the next controller clock. Flip-flop EBI4 LOAD MASS BUF (enabled by LOAD ENA) then sets one clock period later to load the external register data on the EBus data lines (EBUS D24–35) into the Massbus buffer register. After another controller clock period, flip-flop EBI4 MASS BUF RY (enabled by LOAD MASS BUF) sets to assert EBI5 MBUS CB REQ. This signal starts a Massbus cycle to transfer the external register data in the Massbus buffer to the Massbus device. The Massbus (control bus) cycle is described in Subsection 3.1.10. At the completion of the Massbus cycle, EBI1 MBUS CB XFR ENB asserts EBUS XFR to terminate the DATAO operation.

Whereas EBI1 DATAO asserts LOAD EXT REG to start the loading of an addressed external register, the previously discussed LOAD PREP REG level asserts EBI1 INT REG LOAD to load an addressed internal register. INT REG LOAD is asserted at time T1 if the LR bit is set (EBI4 PREP REG 06 = 1) and if the three most significant bits of the register address are true (EBI4 PRFP REG 00–02 = 111). The three asserted address bits, which specify an internal register address (RS = 7x), assert EBI3 INT REG SEL at the input to the INT REG LOAD level.

To complete the decoding of the internal register address, the three least significant address bits (EBI4 PREP REG 03–05) are connected to a binary-to-10-line decoder. One of eight decoder outputs (EBI3 R0–R7) is asserted, depending on the RS code. The decoder outputs are ANDed with the INT REG LOAD level to generate the appropriate load signal. For example, a DATAO addressing the write register (RS = 76) asserts decoder output R6. The R6 level, together with INT REG LOAD, then asserts EBI3 LOAD WR. Five internal registers can be loaded and one of five load signals are generated to do the following:

a. RS = 70 (SBAR) or RS = 71 (STCR) asserts EBI3 LOAD SBAR or EBI3 LOAD STCR to load the internal register data on the EBus data lines into the secondary command file.

b. RS = 74 (IVIR) asserts EBI3 LOAD IVIR to clock the internal register data on the EBus data lines into the IVIR.

c. RS = 76 (WR) asserts EBI3 LOAD WR to generate DP4 WR STROBE. The leading edge of WR STROBE clocks the internal register data on the EBus into the WR.

NOTE

As previously discussed, after the assertion of a load signal in operations (a), (b), or (c), EBUS XFER is asserted at time T2 by EBI1 DATAO to terminate the DATAO operation.

d. RS = 77 (DCR) asserts EBI3 LOAD DCR to start the transfer of internal register data (diagnostic data) from the EBus data lines into the DCR, the diagnostic control register. Similar to the transfer of external register data, the diagnostic data is first loaded in the Massbus buffer. Because a command file transfer also uses the buffer, the input to LOAD DCR (like the input to LOAD EXT REG) is conditioned by EBI5 PCR OUT. If a file transfer is active, LOAD DCR is inhibited until the transfer ends. When asserted, LOAD DCR sets EBI4 LOAD ENA. EBI4 LOAD MASS BUF and EBI4 BUF RY are then set, one after the other, by successive controller clocks. LOAD MASS BUF clocks the diagnostic data into the Massbus buffer register. MASS BUF RY, together with LOAD DCR, asserts EBI3 DCR STROBE, which clocks the diagnostic data in the buffer into the diagnostic control register. DCR STROBE also asserts EBUS XFER to terminate the DATAO operation.

Asserting EBUS XFER in the previously described operations terminates the DATAO operation in the RH20 as follows: when XFER is received by the EBox, it drops DEMAND. In the RH20, the trailing edge of DEMAND negates DC COMP, the time-state generator outputs, and CB XFR ENB if the DATAO had addressed an external register. DC COMP going false negates EBUS ACKN; T0 going false negates EBI1 DATAO. The clearing of CB XFR ENB and EBI1 DATAO negates EBUS XFER and the asserted load levels. If the Massbus buffer had been loaded by the DATAO, flip-flops LOAD ENA, LOAD MASS BUF, and MASS BUF RY are cleared during the next three controller clock periods, ending the operation.

3.1.6 Register Data Read (DATAI)

The DATAI is used to read registers in the RH20 (defined as internal registers) and registers in the Massbus devices connected to the RH20 (defined as external registers). The address and control information necessary to access a register is held in the preparation register, loaded previously by the DATAO command. Bit format for the register data read by the DATAI is shown in Figure 2-20. A flow diagram for the operation is shown in Figure 3-8. Timing is shown in Figure 3-9.

Initially, the EBox asserts the DATAI function code (3) on the EBus function lines (EBUS F00–02) and the device code for the RH20 ($540 - 574_8$) on the controller select lines (EBUS CS00–06). EBUS DEMAND is then asserted to start the operation.

In the RH20, the function line values assert function decoder output EBI1 DATAI EN. When DEMAND is received, DC COMP is asserted by the device code comparator. (The function decoder and device code comparator are described in Subsection 3.1.1.) DC COMP asserts EBUS ACKN and inhibits the start of a command file transfer (Subsection 3.1.9). It also enables the EBI time-state generator (Subsection 3.1.2). The next controller clock then asserts time-state generator output T0, which, together with DATAI EN, asserts EBI1 DATAI. The DATAI level does the following:

a. Asserts the value of the RS code and the LR bit held in the preparation register (EBI4 PREP REG 00–06) on the EBus data lines (EBUS D00–06).

b. Asserts the appropriate internal register read strobe if RS = $70–75_8$; that is, if EBI3 INT REG SEL is true (RS = 7x) and if one of decoder outputs EBI3 R0–R5 is asserted. The decoder, which is also used to generate internal register load signals during the DATAO (Subsection 3.1.5), asserts an output corresponding to the values of the three low-order RS bits. The read strobe that is generated from the decoder output is delayed if a command file transfer is active (EBI5 PCR OUT = 1). (Although DC COMP prevents the start of a transfer, one may be in progress when EBI1 DATAI is asserted.) The delay, until the transfer ends (PCR OUT = 0), is to prevent a file read by the DATAI (RS = $70–73_8$) from interfering with the file read by the command file transfer.

c. Asserts EBUS XFER if the RS code is for an internal register address other than four command file locations (i.e., EBI4 PREP REG 00, 03 = 1), thus providing the normal termination for the DATAI when reading the IVIR or the RR (RS = 74_8 or 75_8). The XFR signal is also asserted to terminate the DATAI if it is illegally referencing the WR or DCR (RS = 76_8 or 77_8), which are write-only registers, and when illegally referencing undefined registers (RS = $40–67_8$) in the address range $44–47_8$, $54–57_8$, and $64–67_8$. (Undefined regis ters have neither an internal nor external register address.) References to the rest of the undefined registers are terminated by a time-out in the EBox. The time-out occurs because a XFER signal is not generated by the RH20.

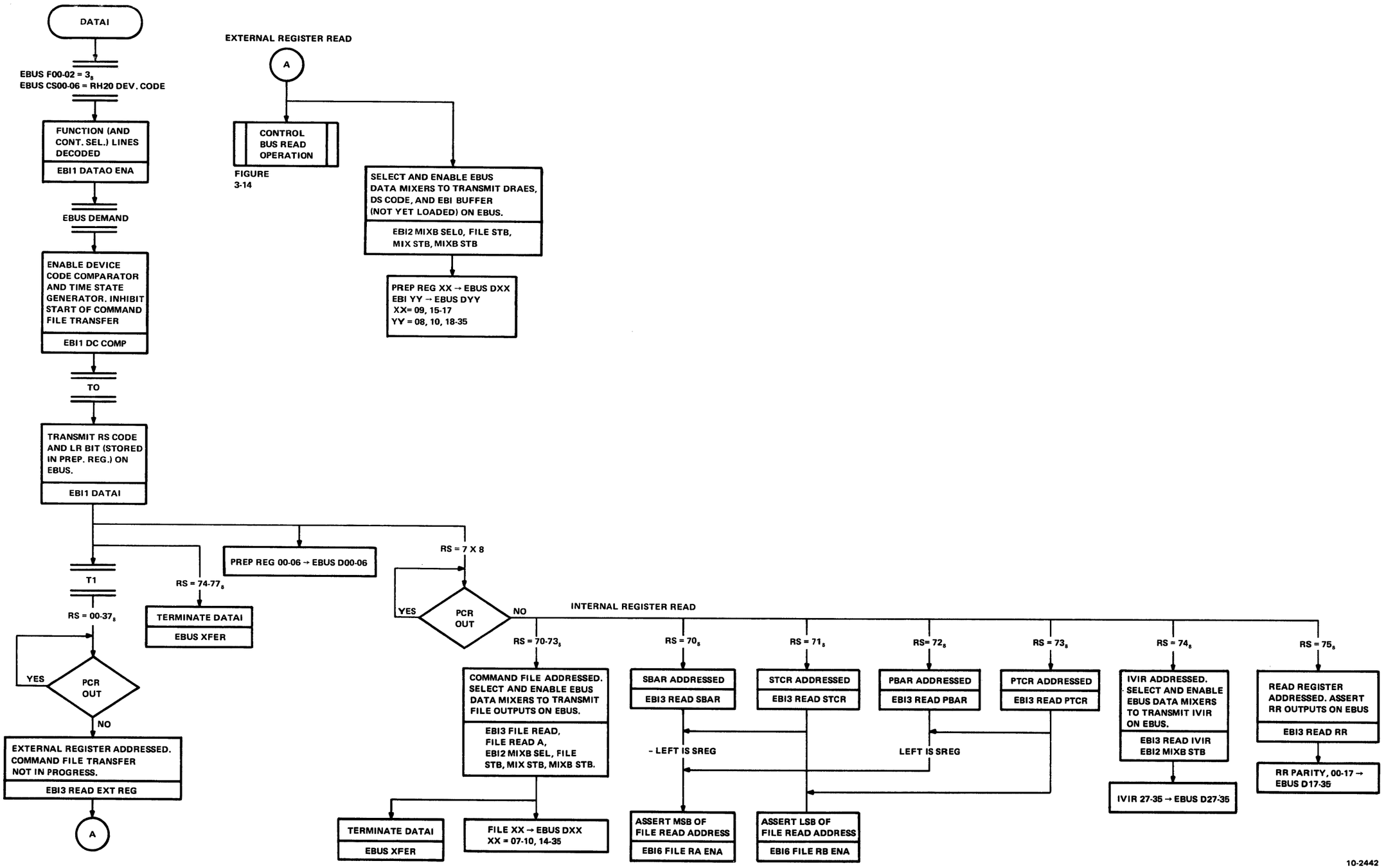


Figure 3-8 DATAI Flow Diagram

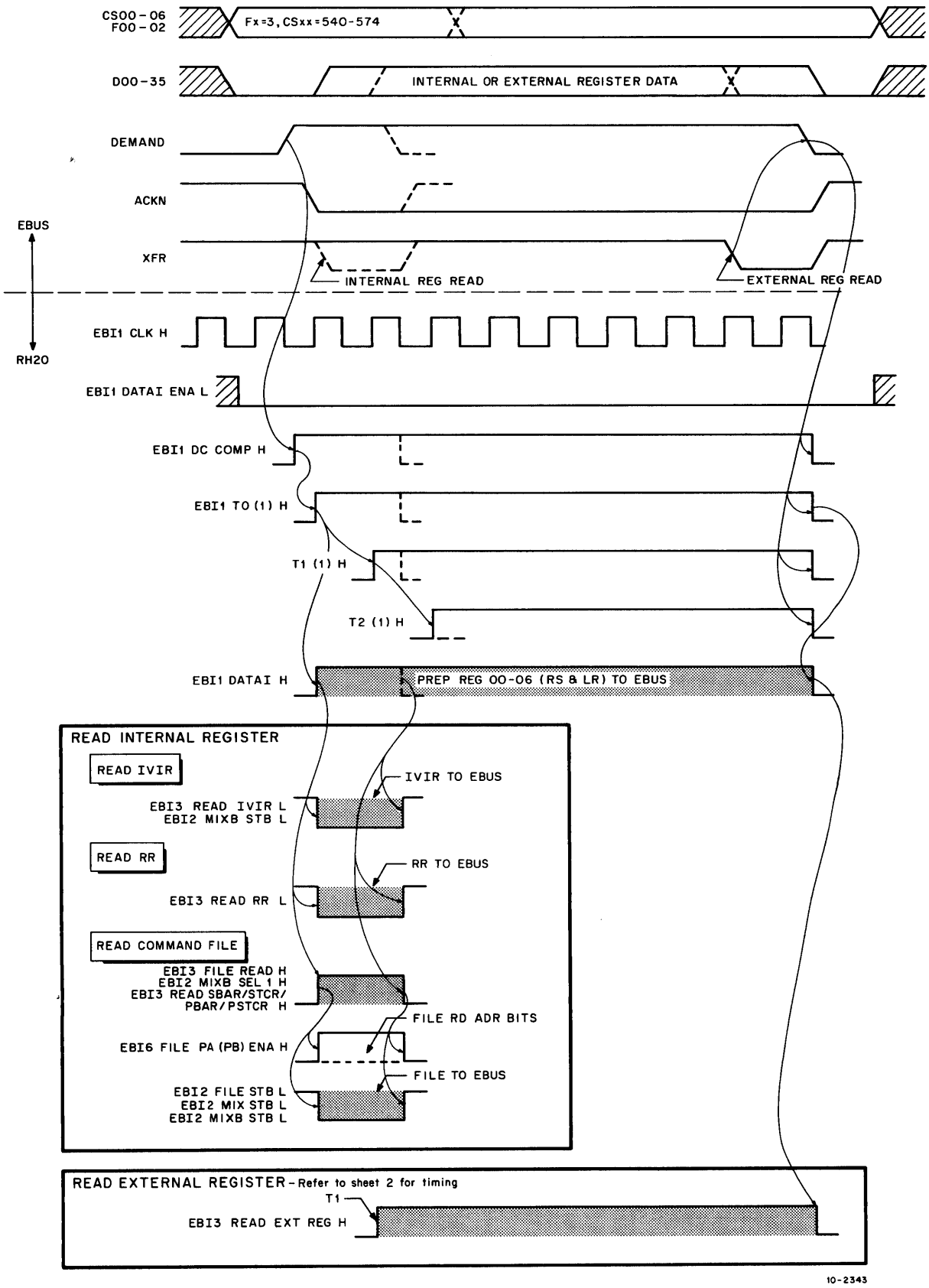


Figure 3-9 DATAI Timing Diagram (Sheet 1 of 2)

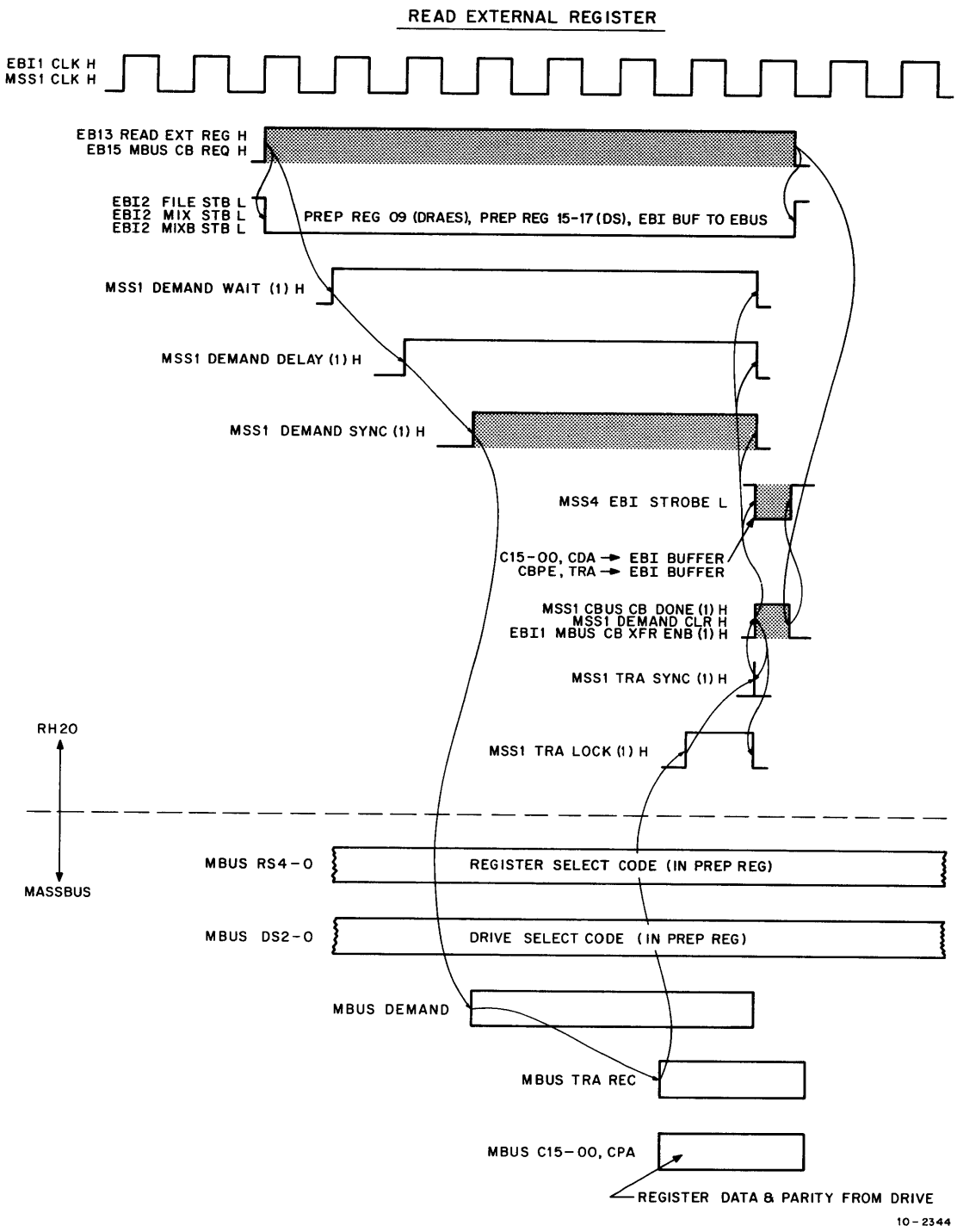


Figure 3-9 DATAI Timing Diagram (Sheet 2 of 2)

d. Asserts EBI3 READ EXT REG at time T1 to start an external register read sequence if RS = $00–37_8$ (EBI4 PREP REG 00 = 0), an external register address, and if a command file transfer is not in progress (EBI5 PCR OUT = 0). READ EXT REG is delayed when a file transfer is active (PCR OUT = 1) because both the file transfer and the external register operation transfer data over the Massbus (control bus). When asserted, at time T1 or later, READ EXT REG causes all EBus data mixer enable levels (EBI2 FILE STB, MIX STB, and MIXB STB) to be asserted, which transmits the DRAES bit status and drive select code held in the preparation register (EBI4 PREP REG 09, 15–17) on the EBus data lines (EBUS D09, 15–17). Also, the inputs to the EBI buffer flip-flops are selected and their outputs transmitted on the EBus. However, the EBI buffer is not loaded until the end of the external register read sequence, that is, until the end of the Massbus cycle. (The Massbus cycle is discussed in Subsection 3.3.10.) As for the DATAO accessing an external register, EBUS XFER is asserted by EBI1 MBUS CB XFR ENB at the completion of the Massbus cycle. The EBox strobes the register data and control status from the EBus data lines and drops DEMAND, ending the DATAI operation in the RH20.

When the internal register read strobe is generated from the RS code, as described in operation (b), information is gated onto the EBus data lines as follows:

1. RS = 74_8 (IVIR) generates EBI3 READ IVIR, which asserts EBI2 MIXB STB, to transmit the contents of the IVIR (EBI4 IVIR 27–35) on the EBus (EBUS D27–35). As occurs during the PI ADR IN operation (Subsection 3.1.8), READ IVIR selects and enables EBI2 MIX 27, the EBus data mixer corresponding to the IVIR's MSB, and the MIXB STB level enables EBI2 MIX 28–35, transmitting the other eight IVIR bits on the bus. This group of eight mixers are selected by EBI2 MIXB SEL 0 and 1, which are both false when reading the IVIR.

2. RS = 75_8 (RR) generates EBI3 RR to transmit the contents of the read register (DP2 RR PARITY, RR 00–17) on the EBus data lines (EBUS D17–35). The information is not gated through the EBus data mixers in the M8555 module; the register outputs connect directly to EBus transmitters on the M8556 board.

NOTE

As discussed in operation (c), EBUS XFER is asserted concurrent with the read strobe when accessing the IVIR and RR. When the EBox receives XFER, it strobes the internal register data from the EBus and drops DEMAND, terminating the DATAI operation in the RH20.

3. RS = $70–73_8$ (SBAR, STCR, PBAR, PTCR) generate the read strobes necessary to initiate and control a command file read operation. (The file read operation is discussed in Subsection 3.1.9.) In addition to the strobe generated for each RS code (i.e., EBI3 READ SBAR, EBI3 READ STCR, etc.), which controls file operation, two additional levels are generated to assert the EBus data mixer select and enable signals necessary to gate the file outputs onto the EBus data lines. EBI3 FILE READ A and EBI3 FILE READ (by asserting EBI2 MIXB STB) select the file outputs at all the EBus data mixers. In addition, FILE READ causes all the mixer enable levels (EBI2 FILE STB, MIX STB, and MIXB STB) to be asserted, transmitting the file outputs (EBI6 FILE 07–10, 14–35) on the EBus (EBUS D07–10, 14–35). FILE READ also asserts EBUS XFR, causing the EBox to strobe the data lines and negate DEMAND.

The negation of DEMAND terminates all DATAI operations in the RH20 by clearing DC COMP, the time-state generator outputs, and CB XFR ENB if an external register had been accessed. EBUS ACKN goes false with DC COMP, EBI1 DATAI with T0, and EBUS XFR with EBI1 DATAI or CB XFR ENB. The trailing edge of EBI1 DATAI also clears the asserted load levels ending the DATAI operation in the RH20.

3.1.7 PI Request Control

The PI request control consists of logic level EBI1 INTR COND, asserted when conditions are met for initiating an RH20 interrupt, and a 4 to 10-line decoder that asserts one of eight outputs, EBI3 PIR 00–07, whenever INTR COND (connected to the decoder's high-order input) goes true. The decoder output that is asserted depends on the binary value (0–7) of EBI4 PIA 33–35 (connected to the decoder's three low-order inputs). The PIA levels are loaded by the CONO command to specify an interrupt channel number. If the channel number is 1–7 when an interrupt condition occurs, the asserted decoder output, PIR 01–07, interrupts the CPU by asserting the corresponding PI request line on the EBus (EBUS PI 01–07). If the channel number is 0, indicating that the RH20 is not to interrupt the CPU, PIR 00 inhibits address comparator output EBI1 INTR COMP. This prevents a response to the PI SERVED command, as explained in Subsection 3.1.1.

INTR COND is asserted when a drive has asserted the attention line on the Massbus (MSS6 MASSBUS ATTN), a synchronous data transfer has terminated (MSS2 CMD DONE = 1), or if an asynchronous data transfer has caused a register access error (MSS3 RAE = 1) and the error has not been disabled by the program (EBI4 PREP REG 09 = 0). The register access error also prevents subsequent DATAO commands from loading a register (EBI1 DATAO INH = 1) until the CPU has serviced the interrupt and the program has either cleared or disabled the error condition.

3.1.8 PI Operation (PI SERVED and PI ADR IN)

Part of the control information loaded in the RH20 by a CONO instruction is a 3-bit priority interrupt channel number (PIA). It is loaded in flip-flops EBI4 PIA 33–35 by the CONO (Subsection 3.1.3). When conditions are met for initiating an interrupt, the RH20 uses the PIA (1–7) to assert one of seven PI request lines (EBUS PI01–07). The PI request logic is described in Subsection 3.1.7.

The EBox detects the assertion of a request line and resolves interrupt priority. (More than one controller may be interrupting and thus more than one request line may be true.) When ready to service the highest priority request (the request on the lowest numbered channel), the EBox executes a PI SERVED command to determine which controller has initiated the request.

To perform the PI SERVED command, the EBox asserts the PI channel number to be served on the three low-order controller select lines (EBUS CS04–06) and the PI SERVED function code (4) on the function lines (EBUS F00–02). It then asserts EBUS DEMAND to initiate the operation. Every interrupting controller, after decoding the function lines and receiving DEMAND, compares its PIA with the channel number encoded on the controller select lines. If a true comparison results (one or more controllers may detect a match), the controller responds by asserting the EBus data line corresponding to its physical address.

Figure 3-10 shows a flow diagram and Figure 3-11 shows timing for the PI SERVED operation in the RH20. Initially, the function lines assert function decoder output EBI1 PI SERVED ENABLE. Comparator output EBI1 INTR COMP is also asserted if the channel number on the select lines matches the PIA (PIA $\neq$ 0) and if the RH20 is interrupting (i.e., EBI1 INTR COND = 1). (The function decoder and the comparator circuit are described in Subsection 3.1.1.) When DEMAND is received, with PI SERVED ENA true, it generates EBI1 PI SERVED to assert the appropriate data line, provided the comparator's output is true. INTR COMP and PI SERVED are ANDed to assert EBUS Dn, where n is the physical number (0–7) of the RH20. The data line is not asserted when INTR COMP is false; that is, when the RH20 is not requesting an interrupt on the PI channel being served.

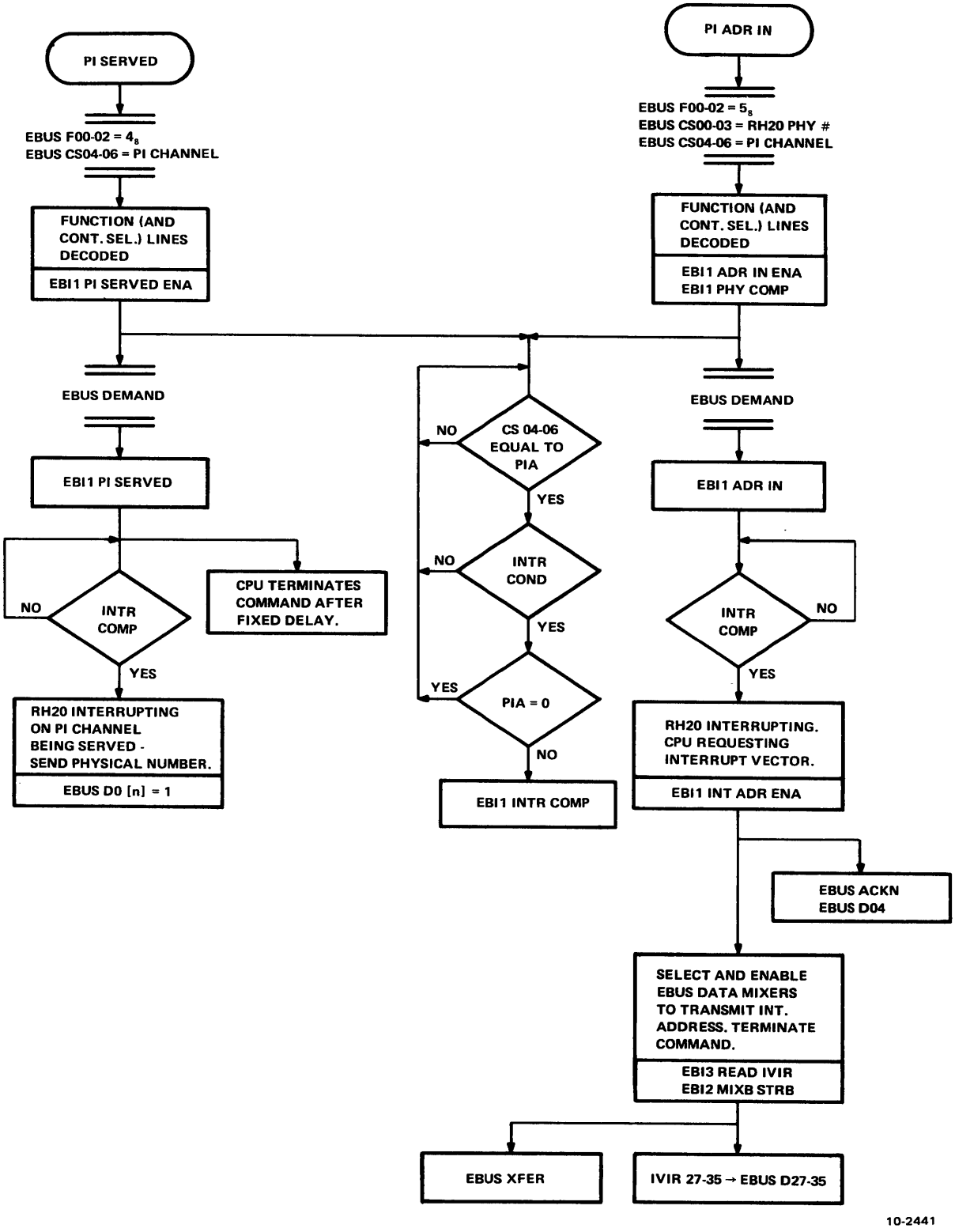


Figure 3-10 PI SERVED/PI ADR IN Flow Diagram

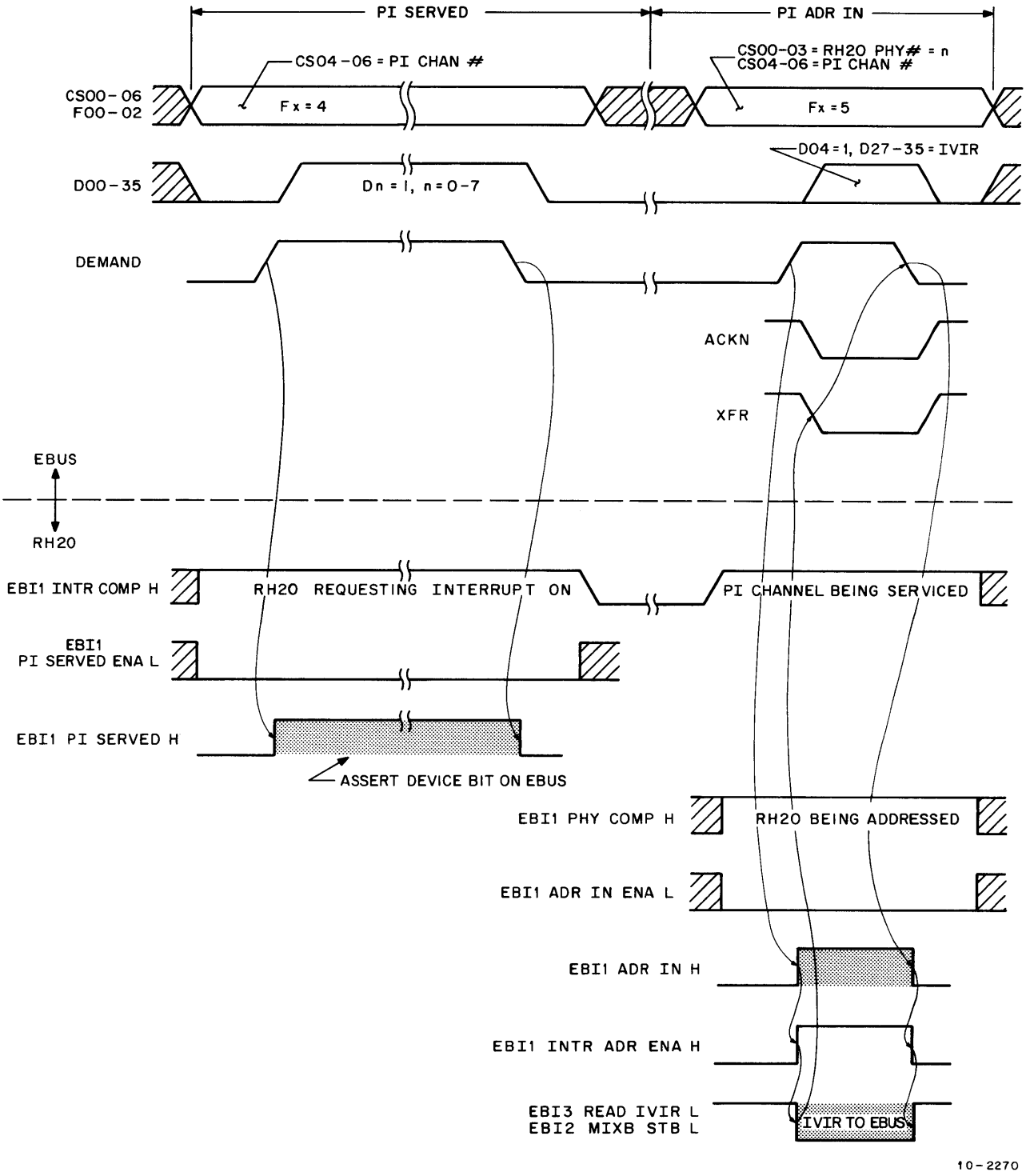


Figure 3-11 PI SERVED/PI ADR IN Timing Diagram

The PI SERVED operation is terminated when the EBox, after waiting at least 400 ns for all interrupting controllers to respond, strobes the EBus data lines and drops DEMAND. In the RH20, the trailing edge of DEMAND causes PI SERVED to go false, which negates the asserted data line (if the RH20 had responded to the command) and terminates the operation.

After a PI SERVED operation, the EBox services a responding controller by executing a PI ADR IN command to collect an API function word over the EBus data lines. (The controller having the lowest physical number is serviced if more than one had responded to the PI SERVED command.) The API response consists of an address space code, an interrupt function code, and an interrupt address (if any). Bit format is shown in Figure 2-5.

To perform the PI ADR IN command, the EBox places the controller's physical address on the four high-order controller select lines (CS00–03) and it aqain places the PI channel number on the three low-order select lines (EBUS CS 04–06). EBUS DEMAND is then asserted to start the operation. As for the PI SERVED operation, every interrupting controller responds to the PI ADR IN command by comparing the PI channel with its PIA. In addition, it compares the physical address on the select lines with its own hard-wired address. If both comparisons are true, the controller responds (only one should respond) by asserting EBUS ACKN and EBUS XFR and by transmitting the API function word on the EBus data lines. The API function word for the RH20 specifies a vector interrupt (function code = 2) to a preassigned EPT location (specified by the interrupt address) in the EPT (address code = 0).

Flow and timing for the PI ADR IN operation in the RH20 is shown in Figures 3-10 and 3-11. Function decoder output EBI1 ADR IN ENA will be true when DEMAND is received. The two levels assert EBI1 ADR IN, which is ANDed with comparator outputs INTR COMP and PHY COMP at the input to EBI1 INT ADR ENA. The INTR COMP level, also asserted during the PI SERVED command, is true if the PI channel number equals the PIA. The PHY COMP level, the output of another comparator (Subsection 3.1.1), is true when the physical address on the controller select lines matches the address of the RH20. With both INTR COMP and PHY COMP true, EBI1 INT ADR ENA is asserted, causing the RH20 to respond to the PI ADR IN command as follows:

a. INT ADR ENA asserts both EBUS ACKN and EBUS D04. The data line is asserted to specify an interrupt function of 2 (vector interrupt).

b. INT ADR ENA also asserts EBI3 READ IVIR to select and enable an EBUS data mixer (EBI2 MIX 27), causing it to transmit the IVIR's MSB (EBI4 IVIR 27) on the corresponding EBus data line (EBUS D27). The other eight bits of the IVIR register (EBI4 IVIR 28–35) are selected at the mixers (EBI2 MIX 28–35) by the negation of EBI2 MIXB SEL 0 and 1. Both levels are false during the PI ADR IN operation.

c. READ IVIR asserts EBI2 MIXB STB, which generates EBUS XFER. At the same time, MIXB STB enables EBI2 MIX 28–35, causing the rest of the IVIR contents (in addition to the MSB) to be transmitted on the EBus data lines (EBUS D27–35).

When the EBox receives XFER, it drops DEMAND. In the RH20, the trailing edge of DEMAND negates ADR IN. This terminates the operation if the RH20 had not been addressed by the command; that is, if INT ADR ENA had not been asserted. If the RH20 had been addressed, INT ADR ENA goes false with ADR IN, negating EBUS ACKN, EBUS DO4, and READ IVIR. The MIXB STRB level then goes false, which ends RH20 operation by negating EBUS XFER and by removing the contents of the IVIR register from the EBus data lines.

3.1.9 Command File Operation and Transfer

The command file serves as a data transfer command buffer between the EBox and the Massbus devices (drives) connected to the RH20. It is a fast memory with four locations that is loaded by the DATAO command and read when a synchronous data transfer (read/write operation) is to be started in a drive. The four file locations are listed in Table 3-1.

Table 3-1 Command File Locations

File Address	Name	Contents
00	Left BAR	Block Address
01	Left TCR	R/W Command
10	Right BAR	Block Address
11	Right TCR	R/W Command

To start a synchronous data transfer, the RH20 reads a TCR (transfer control register) file location, loaded previously by a DATAO. Reading the TCR starts the data channel in the MBox, sets various control functions, and causes the control register in the selected device to be written with the appropriate function code, either read or write. Prior to reading the TCR, a BAR (block address register) file location may also be read to transfer a starting surface address (or frame count) into the block address register in the selected device. The reading of a TCR or BAR file location, and the resulting transfer of control data, is defined as a command file transfer. The control information contained in a TCR or BAR is shown in Figure 2-16.

As indicated in Table 3-1, the file holds two sets of data transfer commands, each consisting of a block address (held by a BAR) and a read/write command (held by TCR). One half of the file, either the right or the left, is the secondary file. It is loaded by the program to start a data transfer operation. When the read or write operation is started, the secondary file is redefined as the primary file. The other half of the file then becomes the secondary file and it can be loaded to specify another data transfer operation while the RH20 is executinq the current one. When the current data transfer operation ends, the transfer specified by the secondary file is started automatically; that is, without program intervention. This provides for minimum latency between data transfer operations. For example, when reading or writing rotating magnetic memories (disks, disk packs, etc.), a data transfer to the next sector (on the same drive) may be initiated within the inter-sector gap. Without a secondary command file, the next sector might be missed and the data transfer could not occur until after a full disk revolution.

Program control of the command file is as follows:

a. Only the two secondary file locations, SBAR (RS = 70) and STCR (RS = 71), can be loaded by the DATAO.

b. Both the secondary file (RS = 70 and 71) and the primary file (RS = 72 and 73) can be read by the DATAI.

c. Normally, both the SBAR and STCR are loaded to specify a read/write operation. However, the SBAR does not have to be loaded when the second of two consecutive data transfers (to/from the same drive) is to start at the surface address following that read or written by the first. This is because the block address register in a drive is incremented during a data transfer and it is left pointing to the next surface address when the first transfer ends.

d. When loading both the SBAR and STCR to specify a read/write operation, the SBAR must be loaded first. This is because loading the STCR starts the operation immediately when there is no read or write already in progress. If the SBAR has not been loaded previously, it is not transferred to the drive.

e. Only read/write commands should be loaded in the STCR. A non-data transfer command (Seek, Rewind, etc.) will cause a hung controller. The non-data transfer commands should be loaded directly into the drive's control register (RS = 00).

Command file operation, in response to the program, is as follows:

a. Initially, at power-up and after a controller reset, the left file is the primary file. Thus, when SBAR and STCR are loaded for the first time, they are loaded in the right (secondary) file.

b. Loading the SBAR does not cause a command file transfer.

c. Loading the STCR for the first time should cause an immediate command file transfer. When the SBAR has been previously loaded (as required for the first read/write operation), it is the first file location read. The SBAR contents are transferred over the control bus portion of the Massbus and loaded into the block address register of the selected drive.

d. Whenever there is a command file transfer of the SBAR, it is immediately followed by a command file transfer of the STCR. This writes the appropriate function code in the control register of the selected drive to start the read/write operation.

e. Whenever there is a command file transfer of the STCR, starting a read/write operation, the primary and secondary files are redefined (e.g., secondary becomes primary and vice-versa). Thus, for the first read/write operation, the right file (initially the secondary file) becomes the primary and the left becomes the secondary. The primary file (PBAR and PTCR) then holds the data transfer commands for the read/write operation currently in progress.

f. With the files redefined and the primary file holding the data transfer commands for the last (and possibly the current) read/write operation, the program can again load the SBAR and STCR. (As mentioned previously, it is not always necessary to load SBAR.) If STCR is loaded after the last read/write operation has ended, it again causes an immediate command file transfer, starting another read/write operation as previously described. If SBAR and STCR are loaded while the last read/write operation is still in progress, the commands in the secondary file become "backup" commands and command file transfers do not take place until the current read/write operation ends.

g. When a read/write operation ends with the secondary file holding "back-up" commands, a command file transfer operation takes place automatically (hardware initiated) to start the next read/write operation. Again, the SBAR (if loaded) is transferred first, followed by the command file transfer of the STCR and the redefining of primary and secondary files. As operation continues, the program can keep the secondary file loaded with "back-up" commands, thus causing minimum latency and optimum use of the Massbus devices.

h. In the operations described previously, a command file transfer will not occur (or be delayed) if:

   1. A transfer error has been asserted as a result of a previous read/write operation. The error must be cleared by a CONO : BIT 26 = 1.

   2. A DATAO, CONO, DATAI or CONI is in progress.

3. The control bus portion of the Massbus is busy.

4. A drive is still active (occupied) as a result of a previous read/write operation.

5. The channel is not ready to service the RH20.

The 4 × 26 command file, EBI6 FILE 07–10, 14–35, consists of seven 4 × 4 fast memory elements with common address connections. (Two of the outputs from one element are unused.) The file may be both read and written at the same time and two sets of address inputs (read and write) and two enable (read and write) are provided on each element. Circuitry associated with the file includes the file address logic, control flip-flops to keep track of which file locations have been loaded and read, and synchronizing logic to control the start of Massbus (control bus) cycles based on the control flip-flop status.

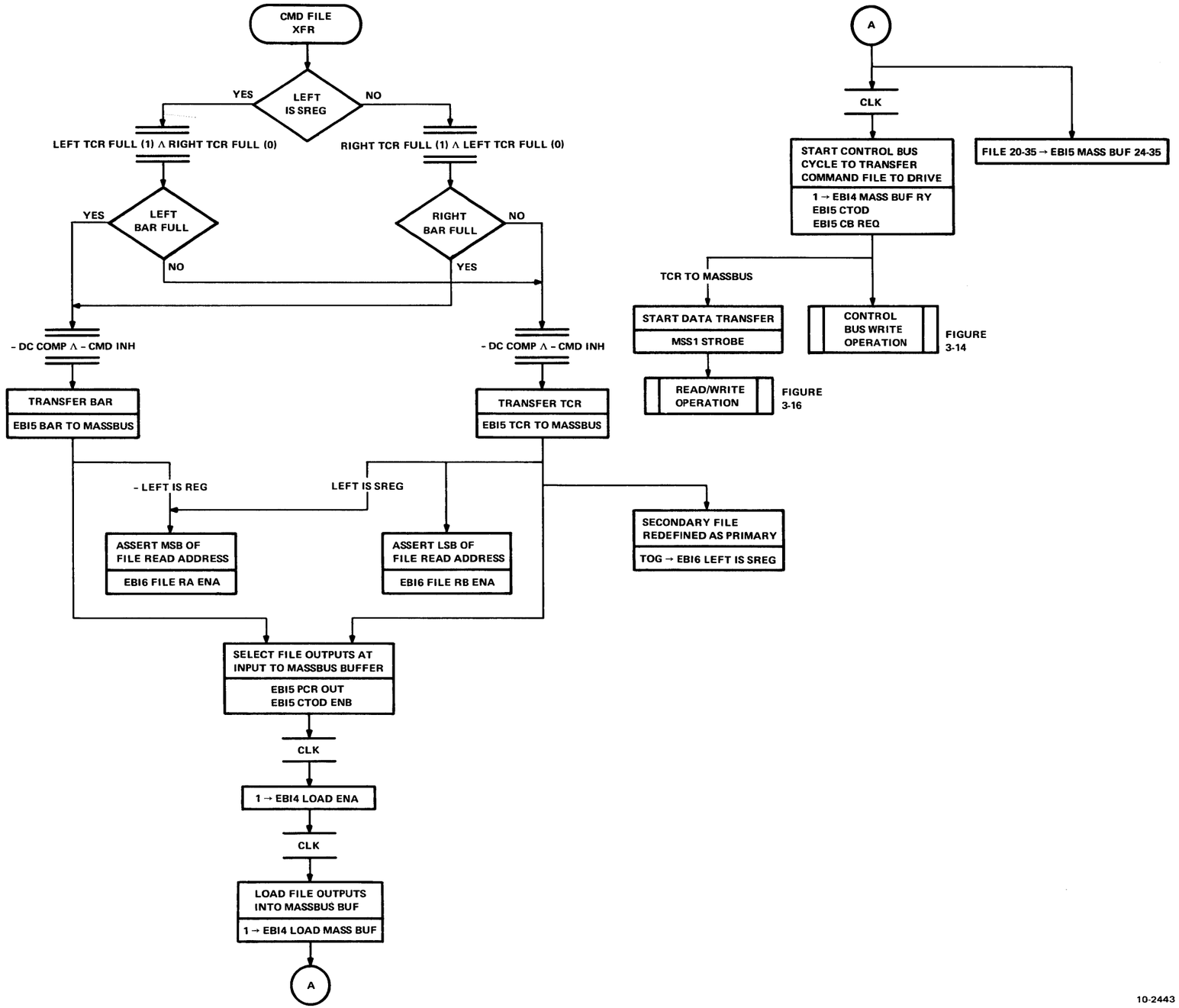
The data input to the file is the control data asserted on the EBus data lines (EBUS D07–10, 14–35) during the DATAO command (Subsection 3.1.5). The control data is written in the appropriate secondary file location when load level EBI3 LOAD SBAR or EBI3 LOAD STCR, asserted by the DATAO (RS = 70 or 71), asserts EBI6 FILE WR ENA at the file's WR EN inputs. The file location that is loaded depends on whether a BAR or a TCR location is addressed, as determined by EBUS D05 (LSB of the RS code), and whether the right or left file is the secondary file, as determined by control flip-flop EBI6 LEFT IS SREG. These two levels, the assertion of EBUS D05 and the negation of LEFT IS SREG, connect to the file's WR ADR inputs to generate the correct address. For example, if the DATAO loads the STCR (RS = 71), EBUS D05 will be true to assert the LSB of the write address. If the left file is the secondary file, flip-flop LEFT IS SREG will be set to negate the MSB of the write address. This causes the control data to be written in location 01, which is the address of the left TCR (Table 3-1).

Four control flip-flops are used to determine when a file location is full:

EBI6 LEFT BAR FUL
EBI6 LEFT TCR FULL
EBI6 RIGHT BAR FULL
EBI6 RIGHT TCR FULL

When a file location is loaded by the DATAO, the appropriate flip-flop is set by the leading edge of the asserted load level. For example, LOAD TCR sets the LEFT TCR FULL flip-flop when LEFT IS SREG is true. The flip-flop outputs are gated with LEFT IS SREG at the clock input to flip-flops EBI5 BAR TO MASSBUS and EBI5 TCR TO MASSBUS to control the start of command file transfers.

Figure 3-12, a flow diagram, shows RH20 operation during a command file transfer. Figure 3-13 illustrates timing. Assume that the RH20 has been initialized (EBI1 CLEAR = 1), which clears all control flip-flops. When the program loads SBAR, RIGHT BAR FULL sets but it cannot set BAR TO MASSBUS until RIGHT TCR FULL sets; that is, until the STCR is loaded. When the program has loaded both secondary files and assuming that MSS2 CMD INH is false, BAR TO MASSBUS sets to start a command file transfer as soon as EBI1 DC COMP goes false. DC COMP is asserted during all EBus commands that address the RH20 by device code. It goes false when the DATAO that has loaded the STCR ends. The normal function of DC COMP at the input to BAR TO MASSBUS (and TCR TO MASSBUS) is to prevent the start of command file transfers immediately after an EBus command has started. This is necessary because the EBus command may also use the Massbus buffer and Massbus.



10-2443

Figure 3-12 Command File Transfer Flow Diagram

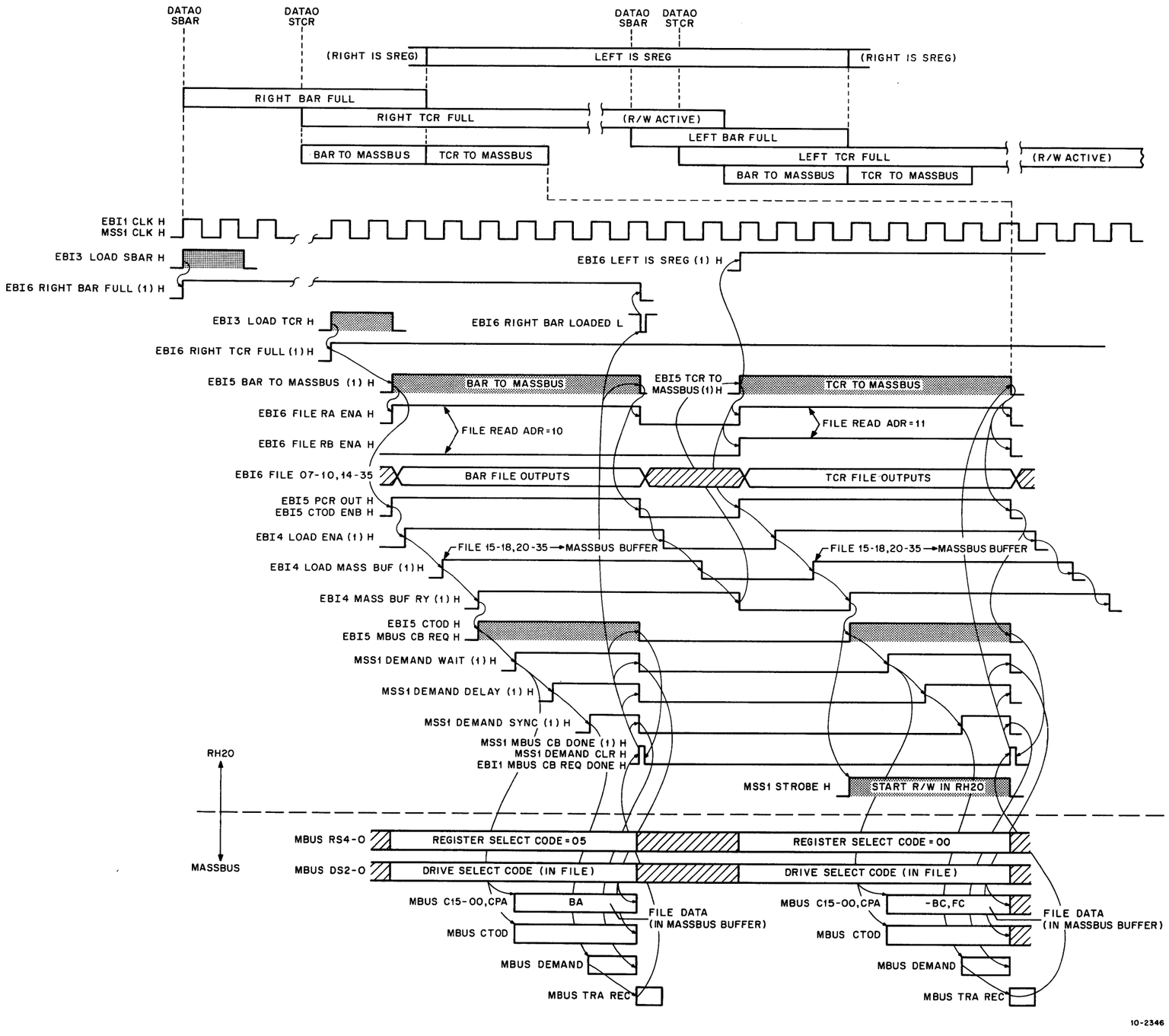


Figure 3-13 Command File Transfer Timing Diagram

At the end of the command file transfer, MSS1 CB DONE sets to assert EBI6 RIGHT BAR LOADED. This clears RIGHT BAR FULL to indicate that the SBAR contents have been transferred to a drive. (CB DONE also asserts EBI1 MBUS CB REQ DONE, clearing the BAR TO MASSBUS flip-flop.)

With the SBAR empty and the STCR still full, STCR TO MASSBUS will set to start a second command file transfer as soon as CMD INH, normally asserted by EBI4 MASS BUS RY at this time, goes false.

CMD INH is gated at the input to STCR TO MASSBUS and BAR TO MASSBUS to do the following:

a. Asserted by EBI5 MBUS CB REQ to cut off the clock input to the flip-flops when the command file transfer has been started. This makes it impossible to start another command file transfer when one is already in progress.

b. Asserted by MSS2 CHANNEL READY to prevent a transfer until the channel is ready to service the RH20.

c. Asserted by MSS2 XFR ERR to prevent a transfer when a previous transfer has caused a transfer error.

d. Asserted by MSS5 OCC REC to prevent a transfer until a previously initiated read/write operation in a drive has ended.

e. Asserted by EBI4 MASS BUF RY to delay the start of the next transfer, thus preventing a race at the input to EBI5 CTOD and the premature start of the next Massbus (control bus) cycle.

Assuming that CMD INH does not remain asserted, STCR TO MASSBUS sets allowing the second command file transfer to take place. STCR TO MASSBUS also toggles the LEFT IS SREG flip-flop, which redefines the file. LEFT IS SREG is now set, cutting off the input to TCR TO MASSBUS, and switching control of both TCR TO MASSBUS and BAR TO MASSBUS to another set of input gates. (The second set of gates are conditioned by the control flip-flops associated with the left file.)

STCR TO MASSBUS is cleared when the second command file transfer ends; that is, when drive's control register has been written and CB REQ DONE has been asserted. RIGHT TCR FULL, however, is not cleared at the same time. RIGHT TCR FULL does not clear until EBI6 RIGHT CLEAR is asserted by MSS2 CMD DONE ENA at the end of the read/write operation that is started by the command file transfer. Thus, the RIGHT TCR FULL flip-flop acts as a busy indicator, remaining set while the read/write operation is in progress.

With a read/write operation active, the program may load "backup" commands in the secondary file (now the left file). When RIGHT TCR FULL is cleared, indicating that the first read/write operation has completed, two command file transfers are made from the left file as described previously for the right file. LEFT BAR FULL and LEFT STCR FULL set BAR TO MASSBUS to start the first transfer. Similar to the clearing of RIGHT BAR FULL, the LEFT BAR FULL flip-flop is cleared by EBI6 LEFT BAR LOADED when MBUS CB DONE is asserted at the end of the first transfer. STCR TO MASSBUS then sets to start the second transfer and toggle LEFT IS SREG, redefining the file. If the program had not loaded the SBAR (next read/write to next surface address on same drive), only STCR TO MASSBUS is set and the RH20 starts a read/write operation without loading the block address register in the drive. This is why the program must load SBAR before STCR. Otherwise, only the STCR is transferred to the drive, causing a read/write to the wrong surface address.

Similar to RIGHT TCR FULL, the LEFT TCR FULL flip-flop is cleared by EBI6 LEFT CLEAR when CMD DONE ENA is asserted at the end of the second read/write operation. As more data transfer commands are loaded in the file, operation continues with the secondary file alternately defined as left and right, and the FULL flip-flops controlling the start of the command file transfers under control of LEFT IS SREG.

Whenever BAR TO MASSBUS or TCR TO MASSBUS is set to start and control a command file transfer, the following occurs:

a. Together with LEFT IS SREG, BAR TO MASSBUS or TCR TO MASSBUS generates the appropriate file read address; that is, either the address for the secondary BAR or the address for the primary TCR. (The secondary TCR is redefined as the primary at start of the command file transfer.) For example, TCR TO MASSBUS asserts EBI6 FILE RB ENA (LSB of read address) and it asserts EBI6 FILE RA ENA (MSB of read address) when LEFT IS REG = 1. The resulting file read address of 11 causes the contents of the primary TCR (the right TCR, in this example) to be immediately asserted at the file's outputs. Outputs are asserted immediately because the file's RD EN inputs are always enabled (wired to ground).

b. BAR TO MASSBUS or TCR TO MASSBUS asserts EBI5 PCR OUT to select file outputs EBI6 FILE 15–18 and 20–35 at the input to the Massbus buffer register. EBI5 MASS BUF 15–17, which are mixer outputs, then assert the drive address held in FILE 15–17 on the Massbus DS lines (MBUS DS2–0). EBI5 MASS BUF 20–35, which are flip-flops, are not loaded at this time.

c. PCR OUT asserts EBI5 CTOD ENA, which causes flip-flops EBI4 LOAD ENA, EBI4 LOAD MASS BUF, and EBI4 MASS BUF RY to be set one after the other by successive controller clocks. LOAD MASS BUF clocks the Massbus buffer flip-flops, loading FILE 20–35. The information in the buffer, either the block address when SBAR is read or the block count and function code when PTCR is read, is asserted on the Massbus control data lines (MBUS C15–00).

d. MASS BUF RY, together with CTOD ENB, asserts EBI5 CTOD and FBI5 MBUS CB REQ to start a Massbus (control bus) cycle. The file information in the Massbus buffer, asserted on the Massbus control data lines, is then written in the addressed drive. The Massbus cycle is described in Subsection 3.1.10.

NOTE

Although the block count is asserted on the Massbus control data lines during the command file transfer of a TCR location, it is not loaded in the drive's control register. Only the function code is loaded; all other information is ignored.

e. If TCR TO MASSBUS has initiated the command file transfer, MASS BUF RY also asserts MSS2 STROBE to load the block count (MASS BUF 20–29) into the RH20 block counter, the RCLP control bit (FILE 07) in MSS1 RCLP ENA, the STORE control bit (FILE 10) in MSS1 CBUS STORE ENA, and the DTES control bit (FILE 19) in MSS1 DTES. In addition, STROBE sets MSS1 READ (MASS BUF 32 = 1) or write (MASS BUF 32 = 0) and MSS1 START ENA to start the read/write operation in the RH20. The synchronous data transfer operation is described in Subsection 3.2.

If the RH20 hangs attempting to perform a command file transfer (e.g., if CMD INH remains asserted), the program may clear the hung condition and abort the read/write operation specified by the commands in the secondary file without initializing the RH20. A CONO with DELETE SCR (bit 29) = 1 asserts EBI4 DELETE SREG, which clears BAR TO MASSBUS and TCR TO MASSBUS. Also, DELETE SREG asserts RIGHT CLEAR or LEFT CLEAR, depending on the state of LEFT IS SREG. This removes the input to BAR TO MASSBUS and TCR TO MASSBUS by clearing the secondary files BAR FULL and STCR FULL flip-flops.

Two command file status bits are provided which can be read by the CONI. These are SCR FULL (bit 29), asserted by EBI6 STCR FULL, and PCR FULL (bit 31), asserted by EBI6 PTCR FULL. The appropriate status bit is asserted whenever the TCR location has been loaded (as determined from the TCR FULL flip-flops) in a secondary or primary file (as determined by LEFT IS SREG).

As stated previously, file operation allows simultaneous read and write operations. Also, the DATAO to write a file location can occur during a command file transfer. Thus, it is possible to load a SBAR file location at the same time that its contents are being read and transferred to a drive. (The race condition cannot occur when loading the STCR; the PTCR is read during a command file transfer, not the STCR.) Changing the SBAR contents during the command file transfer can cause a malfunction, but the race condition should never occur during normal programming of the RH20. The program should not issue a DATAO to load the SBAR until the SCR is empty (CONI: bit 29 = 0); that is, until after the command file transfer of the SBAR location has taken place.

3.1.10 Massbus (Control Bus) Cycle

The Massbus (control bus) read or write cycle transfers control data from or to the drives connected to a Massbus controller. In the RH20, the control bus write cycle takes place when an external register is addressed by the DATAO (Subsection 3.1.5) and when the control or block address register is written during the command file transfer (Subsection 3.1.9). A control bus read cycle occurs when an external register is accessed by the DATAI (Subsection 3.1.6).

During a control bus cycle, the RH20 controller asserts a drive address on the three Massbus drive select lines (MBUS DS2–0) and a register address on the five Massbus register select lines (MBUS RS4–0). If a write cycle is to take place, it also asserts the "controller to drive" line (MBUS CTOD) and it places the control data that is to be transferred on the control bus data lines (MBUS C15–00, CPA). The demand line (MBUS DEM) is then asserted by the controller to indicate that the transfer of data (to or from the drive) is to take place. Upon receiving the DEM signal, the addressed drive either loads the control data on the C lines into the specified register (if CTOD asserted) or it places the data in the specified register onto the C lines (if CTOD negated). The drive also asserts the Massbus transfer line (MBUS TRA) to indicate that the read or write of register data has been made. When the controller receives the TRA signal, it negates DEM and (if CTOD negated) strobes the register data from the C lines.

A flow diagram for RH20 operation during a control bus cycle is shown in Figure 3-14. Timing, as applies to the DATAO, DATAI, and command file transfer is shown in Figures 3-7, 3-9 and 3-13. All three operations start the control bus cycle by asserting EBI5 MBUS CB REQ. When the request level goes true, the DS and RS lines are already asserting the correct drive and register address.

The Massbus drivers for the RS lines normally transmit the register address held in the preparation register (EBI4 PREP REG 01–05), which is the correct register address for the DATAO and DATAI operations. When a command file transfer is started, MSS3 PCR OUT (together with EBI5 BAR TO MASSBUS and EBI5 TCR TO MASSBUS) force the appropriate RS code, overriding the preparation register inputs. That is, all drivers are disabled to address the drive's control register (RS = 00) during a command file transfer of the TCR. When transferring the BAR file, the drivers for RS0 and 2 are forced on, and the other disabled, to address the drive's block address register (RS = 05_8).

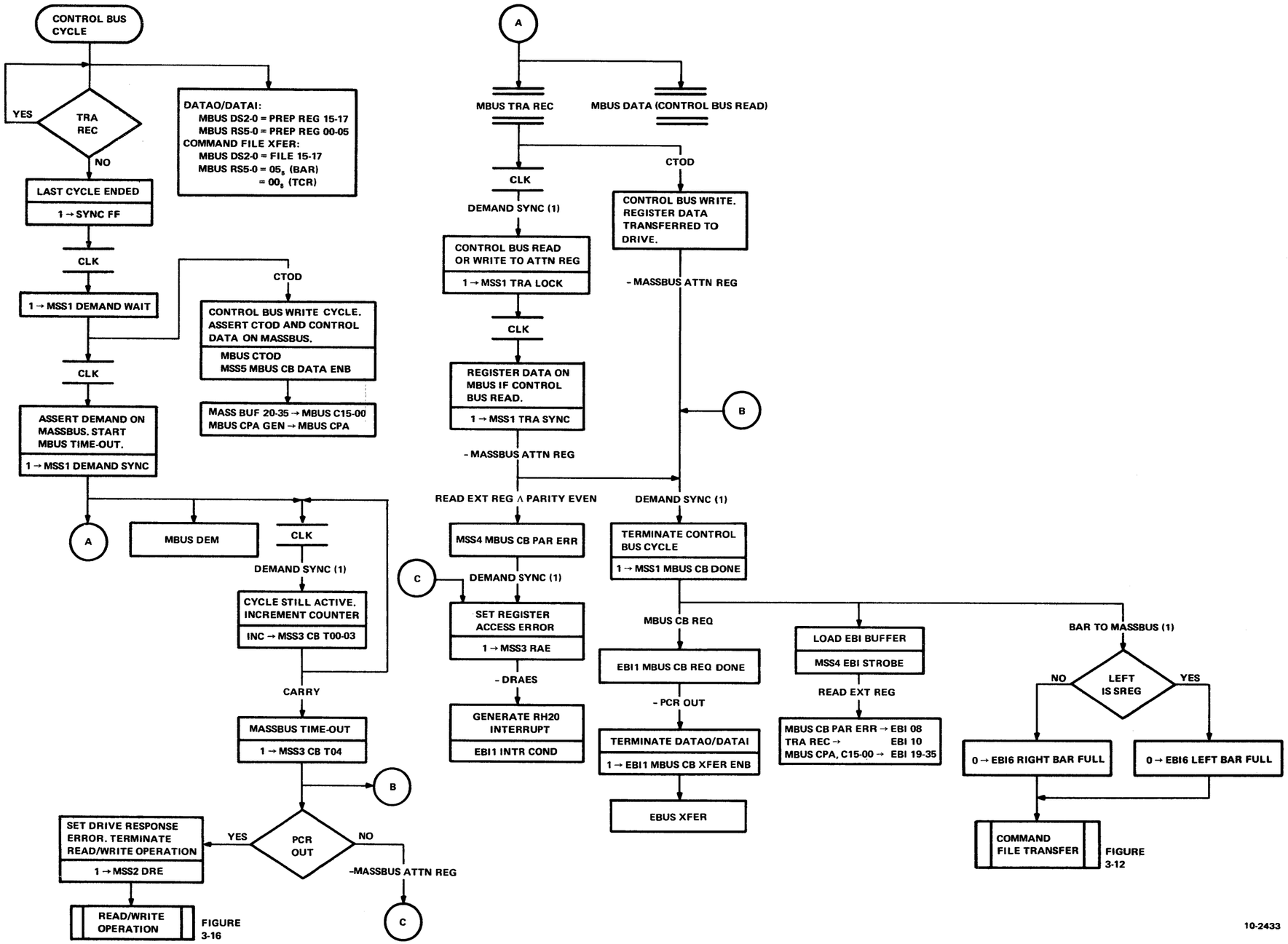


Figure 3-14 Control Bus Cycle Flow Diagram

The DS lines, like the RS lines, normally assert the address held in the preparation register (EBI4 PREP REG 15–17); that is, the drive address for the DATAO and DATAI operations. The preparation register outputs drive the Massbus DS drivers via EBI5 MASS BUF 15 - 17, the Massbus buffer mixer, when a command file transfer is not active. When a transfer is started (PCR OUT = 1), another set of inputs to the Massbus buffer mixer is selected, asserting the drive address held in the command file (EBI6 FILE 15–17) on the DS lines.

With the DS and RS lines selecting a drive and register in that drive, MBUS CB REQ starts a control bus cycle as follows:

a. MBUS CB REQ sets a synchronizing flip-flop if the bus dialogue resulting from the last control bus cycle has ended; that is, if TRA is no longer asserted on the bus (MSS5 TRA REC = 0).

b. Once asserted, the synchronizing flip-flop causes MSS1 DEMAND WAIT to be set by the next controller clock. If the control bus cycle is a write cycle (EBI5 CTOD = 1), the DEMAND WAIT flip-flop asserts MBUS CTOD on the bus and it asserts MSS5 CB DATA EN, A, B to enable the Massbus drivers for the control data lines, MBus C15–00 and CPA. Enabling the C line drivers transmits the control data in the Massbus buffer flip-flops (EBI5 MASS BUF 20–35) and the parity bit (MSS4 MBUS CPA GEN) on the bus. The parity generator circuit is described in Subsection 3.1.11.

c. DEMAND DELAY, enabled by DEMAND WAIT, is set by the next controller clock. The function of the DEMAND DELAY flip-flop is to introduce a deskewing delay before DEM is asserted on the bus.

d. With DEMAND DELAY set, flip-flop DEMAND SYNC is set by the next controller clock to assert MBUS DEM. The addressed drive should respond by asserting MBUS TRA.

When TRA is received by the RH20 (MSS5 TRA REC = 1) during a control bus write cycle (EBI5 CTOD = 1), indicating that the drive has strobed the control data from the C lines, MSS1 MBUS CB DONE is set to terminate the cycle provided that the attention summary register has not been addressed (MSS2 MASSBUS ATTN REG = 0). MBUS CB DONE asserts MSS1 DEMAND CLR, which clears the DEMAND flip-flops (and the synchronizing flip-flop), negating the DEM signal on the bus. MBUS CB DONE also asserts EBI1 MBUS CB REQ DONE. If a DATAO had started the cycle, this signal sets EBI1 MBUS CB XFR ENB to assert EBUS XFER and end the operation. If the cycle is the result of a command file transfer, MBUS CB REQ DONE ends file operation by clearing BAR TO MASSBUS or TCR TO MASSBUS.

When TRA is received during a control bus read cycle, indicating that the register data from the drive is on the C lines, MSS1 TRA LOCK and TRA SYNC are set, one after the other, by successive controller clocks. (These flip-flops do not set during a control bus write operation because DEMAND SYNC, cleared by TRA during the write, is ANDed at the input to TRA LOCK.) The function of TRA LOCK is to introduce a deskewing delay before the control data and parity on the C lines are strobed by the RH20. When TRA SYNC sets, one clock period later, it strobes the output of the parity checker, MSS4 CB PAR ERR, and it sets MBUS CB DONE if the attention summary register has not been addressed. (The parity checker and parity error indicators are described in Subsection 3.1.11.)

MBUS CB DONE clears the DEMAND flip-flops, as for the write operation, and it asserts MSS4 EBI STROBE. Both levels, MBUS CB DONE and EBI STROBE, strobe the register data on the C lines into the EBI buffer register ending the control bus cycle. As for a write cycle initiated by a DATAO, MBUS CB REQ DONE (asserted by MBUS CB DONE) sets MBUS CB XFR ENB. This asserts EBUS XFR, ending the DATAI that initiated the control bus read cycle.

As stated previously, MBUS CB DONE is not set by the TRA signal whenever the attention summary register is addressed. MSS2 MASSBUS ATTN REG, asserted when RS = 04, blocks TRA REC at the clock input during the control bus write cycle. It also blocks TRA SYNC during the read cycle. As a result, MBUS CB DONE is not set until MSS3 MBUS CB T04, the output from a time-out circuit, goes true 16 controller clock periods after the assertion of DEMAND SYNC. The reason for terminating the cycle after a predefined time interval is that all drives respond with a TRA signal when the attention summary register is referenced. If the first TRA that was received terminated the cycle, drives further down the bus (not yet asserting TRA) would not have time to transfer control data. The time-out provides enough time for all drives to respond.

Another function of the time-out is to terminate the cycle whenever a nonexistent drive is addressed. TRA is not received when a drive is not present, allowing MBUS CB T04 to set MBUS CB DONE, as when the attention summary register is addressed. To flag the condition, MBUS CB T04 also sets MSS3 RAE (register access error) when a DATAO or a DATAI (PCR OUT = 0) referenced the nonexistent drive and it sets MSS2 DRE (drive response error) when a command file transfer (PCR OUT = 1) made the reference.

The time-out circuit consists of a 4-stage binary counter (MSS3 CB T00–T03) that is enabled to increment at the controller clock frequency by DEMAND SYNC, when DEM is asserted on the bus. If DEM sync remains asserted; that is, if TRA is not received (nonexistent drive) or blocked (attention summary register addressed), the CARRY output is asserted (low) after 15 up-counts. The next controller clock, the 16th, clears the counter and the CARRY output is negated (high) to set flip-flop MSS3 MBUS CB T04. As explained previously, MBUS CB T04 sets MBUS CB DONE and sets an error flag if a nonexistent drive caused the time-out. Because MBUS CB DONE clears DEMAND SYNC, the counter enable level, the counter stops incrementing and (as required) will have a count of ZERO at the start of the next control bus cycle.

3.1.11 Massbus Control Data Parity Circuits – Parity is generated and checked on the Massbus control data lines (C lines) by four 9-bit (8 data inputs and a parity input) parity generator/checker circuits. Two circuits are cascaded to generate parity bit MSS4 MBUS CPA GEN prior to the start of a control bus write cycle, when the 16 Massbus buffer flip-flops (EBI5 MASS BUF 20–35) are loaded with the data subsequently transmitted on the C lines. When the C lines are asserted, the parity bit for the Massbus buffer contents is also transmitted on the bus. Odd parity is normally generated when a DATAO (MSS3 PCR OUT = 0) has initiated the control bus cycle. However, even parity can be generated if the CBEP control bit is set (EBI5 MASS BUF 18 = 1). This allows the diagnostic programmer to isolate failures in the data path and to verify the parity checking circuits in the drive. Odd parity is always generated when a command file transfer has initiated the control bus write cycle.

The other two parity generator/checker circuits (and an XOR gate) check the parity of the C lines (MSS7 C15–00, MSS6 MBUS CPA REC) during the control bus read cycle (EBI3 READ EXT REG = 1) if the attention summary register has not been addressed (MSS2 MASSBUS ATTN REG = 0). (Because no single drive generates all the C bits when the attention summary register is addressed, valid parity cannot be generated on the bus and the parity check must be disabled.) If incorrect (even) parity is detected after the lines have settled (MSS1 TRA SYNC = 1), MSS4 MBUS CB PAR ERR is asserted to set MSS3 RAE (register access error). The RAE flag (CONI: bit 24) generates an RH20 interrupt if the error has not been disabled by the program (DRAES = 0). MBUS CB PAR ERR is also stored in the EBI buffer (MSS4 EBI 08) allowing the error flag to be read by the DATAI (bit 08) along with the C line data. The program can force a parity error by setting the even parity test bit (MSS4 DIAG EP TEST) in the DCR. This causes the parity network to assert CB PAR ERR when the C lines have correct (odd) parity and allows the diagnostic programmer to verify the operation of RAE and to generate an interrupt.

3.2 SYNCHRONOUS DATA TRANSFER CONTROL LOGIC

The synchronous data transfer control logic interfaces the RH20 to the CBus and to the data bus portion of the Massbus. It contains the logic elements necessary to perform either a write data transfer operation, which collects data from an MBox channel and transfers the data to a Massbus device, or a read data transfer, which collects data from a Massbus device and transfers the data to an MBox channel.

3.2.1 CBI Clock Control

An RH20 Massbus controller (MBCn), by virtue of its physical number (n = 0–7), is linked to a specific MBox channel (0–7). MBC0 is serviced by MBox channel 0; MBC1 by channel 1, etc. Also, as explained in Subsection 2.1.3, the MBox continuously scans and selects the RH20 controllers in a system, and thus the channels in a system, by asserting the appropriate controller select line (CBUS SEL n) on the multiplexed CBus. CBUS SEL n defines the first of the four channel cycles (time slots) associated with each controller. The select cycle is followed by a request cycle, a wait cycle, and a data cycle as shown in Figure 2-13.

The duration and repetition rate of CBUS SEL n (i.e., the select cycle) is determined by and synchronized by the basic CP clock, which is distributed to each RH20 via the clock distribution module. The clock, CDS1 EBUS CLK 0n, is adjusted (deskewed) so that channel time 0 (T0) and RH20 clock DP4 CLK coincide. This synchronizes RH20 and channel operation.

DP4 CLK and its complement, DP4 CLK A, are generated directly by CDS1 EBUS CLK 0n. The clocks sequence the synchronous data transfer control logic and clock the following four flip-flops, connected in tandem, to generate a timing train started by the controller select level.

DP4 CHAN COMPARED
DP4 WAIT
DP4 MEM REC DATA
DP5 MEM SEND DATA

The flip-flop outputs correspond to the four channel cycles as shown in Figure 3-15.

3.2.2 Block Counter

Part of the information contained in a TCR command file location is a 10-bit negative block count. It specifies the length (in 2's complement) of the synchronous data transfer, that is, the number of sectors (or records) to be read or written by the RH20. The count is read from the command file into MASS BUF 20–29 and loaded into a binary counter by MSS1 STROBE during a command file transfer operation (Subsection 3.1.9). The command file transfer also starts a read or write operation in a Massbus device (drive) by writing the appropriate function code in the selected drive's control register. When the transfer of data starts, the binary counter is incremented by MSS7 EBL REC, the received end of block signal transmitted on the Massbus by the drive at the end of each sector (or record). At the end of the last block, the binary counter overflows and asserts MSS1 BC OFF. The overflow signal, signalling the end of the read/write operation, is used to terminate the RH20 and channel as described in Subsection 3.2.7.

The block counter is made up of three cascaded 4-bit synchronous binary counter elements. Synchronous operation provides a glitch-free CARRY output (the BC OVF output) during the initial load and during an up-count. Furthermore, BC OVF is asserted not when the count goes to all 1s, but on the following count. For example, when a block count of –1 (all 1s) is loaded to specify a one sector (record) transfer, BC OVF is not asserted until the first up-count or the first EBL signal, as required. BC OVF remains true for the duration of the EBL level.

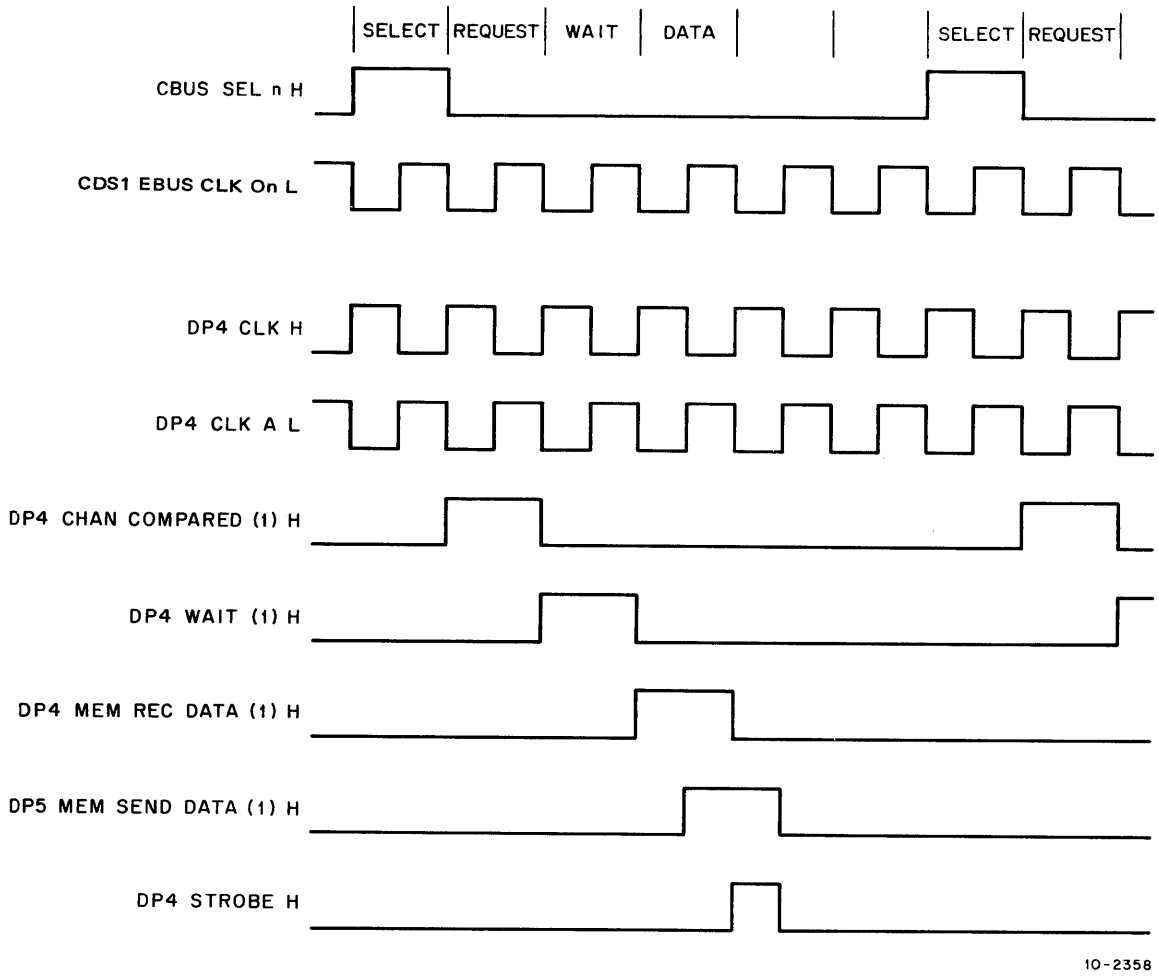


Figure 3-15 CBI Clock Control Timing Diagram

3.2.3 Data Buffers

Data words transferred to/from the CBus are 36 bits plus 2 parity bits, one parity bit for each 18 bits (right or left half-word) of data. Data words transferred to/from the Massbus are 18 bits plus parity.

During a write operation, CBus data words are loaded in the 38-bit secondary data register (DP2 SRA 00–17, PARITY and DP2 SRB 00–17, PARITY), transferred to the 38-bit primary data register (DP2 PRA 00–17, PARITY and DP3 PRB 00–17, PARITY), and then loaded via the data mixers (DP3 MIXER 00–17, PARITY) into the 19-bit write register (DP3 WR 00–17, PARITY) one half-word at a time for transfer to a drive over the Massbus. The parity bit transferred over the Massbus is not regenerated; that is, it is the parity bit (either right or left) loaded from the CBus.

During a read operation, Massbus data is loaded into the 19-bit read register (DP2 RR 00–17, PARITY) and alternately transferred to SRA and SRB. Whenever a full word is assembled in the SR, it is transferred to the PR and then transferred over the CBus to the MBox. As for the write operation, parity is not regenerated. The parity bits received on the Massbus are transmitted on the CBus.

3.2.4 Data Transfer Startup

A flow diagram for RH20 operation during a read/write data transfer is shown in Figure 3-16. During a command file transfer of a TCR location, and at the same time that EBI5 MBUS CB REQ is asserted to start the Massbus (control bus) cycle that writes a read or write function code in a drive's control register, MSS1 STROBE is asserted to start the read or write data transfer operation. The STROBE level does the following:

a. Sets MSS1 START ENA.

b. Sets MSS1 RCLP ENA if the command list pointer is to be reset (FILE 07 = 1).

c. Sets MSS1 WRITE if the data transfer is a write operation (MASS BUF 32 = 0) or MSS1 READ if it is a read operation (MASS BUF 32 = 1).

d. Sets MSS1 DTES if certain transfer errors are to be disabled; that is, prevented from terminating the subsequent data transfer (FILE 19 = 1).

e. Sets MSS1 CBUS STORE ENA if channel status is to be stored in the EPT at the termination of the data transfer (FILE 10 = 1).

NOTE
Status is stored unconditionally if a transfer error is detected during the transfer.

If MSS1 WRITE is set, DP4 WRITE is clocked on to assert the three Massbus data line driver enable levels (DP4 MASSBUS ENA, A, B). If MSS1 READ is set, DP4 READ is clocked on and MSS1 RUN is set to assert RUN on the Massbus. The addressed drive uses MBUS RUN to begin the search for the specified surface address (disk drives) or to start tape motion (tape drives).

CBUS START starts the MBox channel associated with the RH20. It is asserted during the controller's data cycle. With START ENA set, the first wait cycle that occurs (DP4 WAIT = 1) sets synchronizing flip-flop MSS2 START TO MB. This signal asserts the CBus driver input for the START level. The driver's BUS EN input is then asserted during the next channel cycle, a data cycle, by DP4 MEM REC DATA to transmit CBUS START to the MBox. The MEM REC DATA level is used to enable all CBus drivers that transmit control signals during the data cycle.

CBUS RESET is asserted at the same time as CBUS START when the command list pointer is to be reset; that is, when RCLP EN = 1. Similar to START ENA, the RCLP EN level sets a synchronizing flip-flop, MSS2 RCLP, at the start of the wait cycle. This asserts MSS2 RCLP TO MB, which generates CBUS RESET during the data cycle, coincident with CBUS START. The CBUS RESET signal can also be generated by the CONO command. (When the RH20 is cleared during a data transfer operation, the channel must also be reset.) A CONO (Subsection 3.1.3) with bit 28 = 1 sets MSS1 CONO RCLP ENA, which sets MSS2 CONO RCLP and asserts RCLP TO MB. As when initializing the channel at startup, RCLP to MB asserts CBUS RESET during the data cycle.

The START ENA, RCLP ENA, and the CONO RCLP ENA flip-flops, which are all set during the wait cycle, are cleared during the next request cycle. Thus, CBUS START and CBUS RESET (if asserted) are transmitted for only one data cycle interval.

CBUS CTOM (controller to memory) is asserted coincident with CBUS START (and CBUS RESET) if the data transfer is a read operation. Unlike the start and reset signals, CBUS CTOM is asserted for more than one data cycle. The negation of MSS1 WRITE asserts the input to the bus driver, causing CTOM to be generated every data cycle throughout the read data transfer.

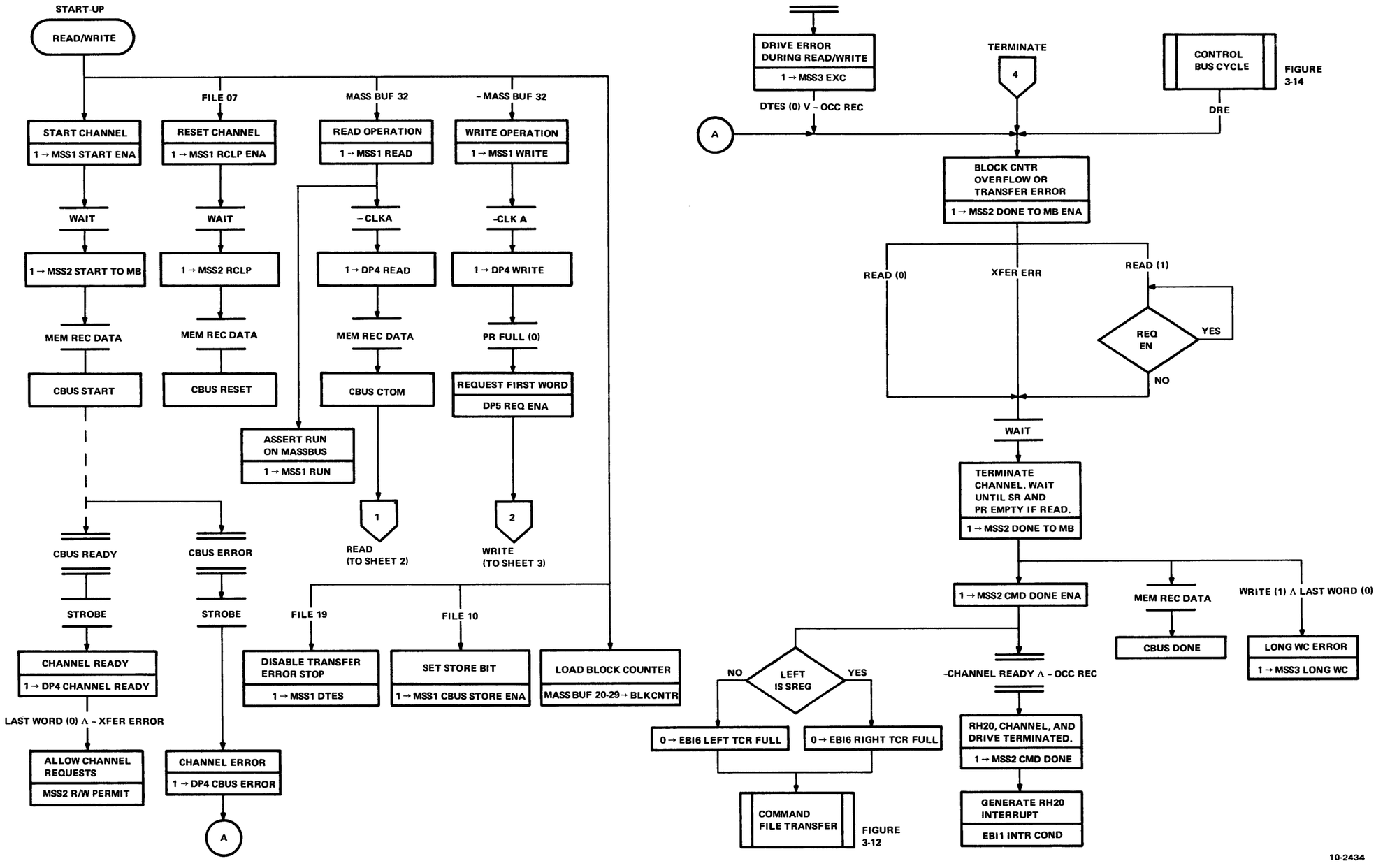
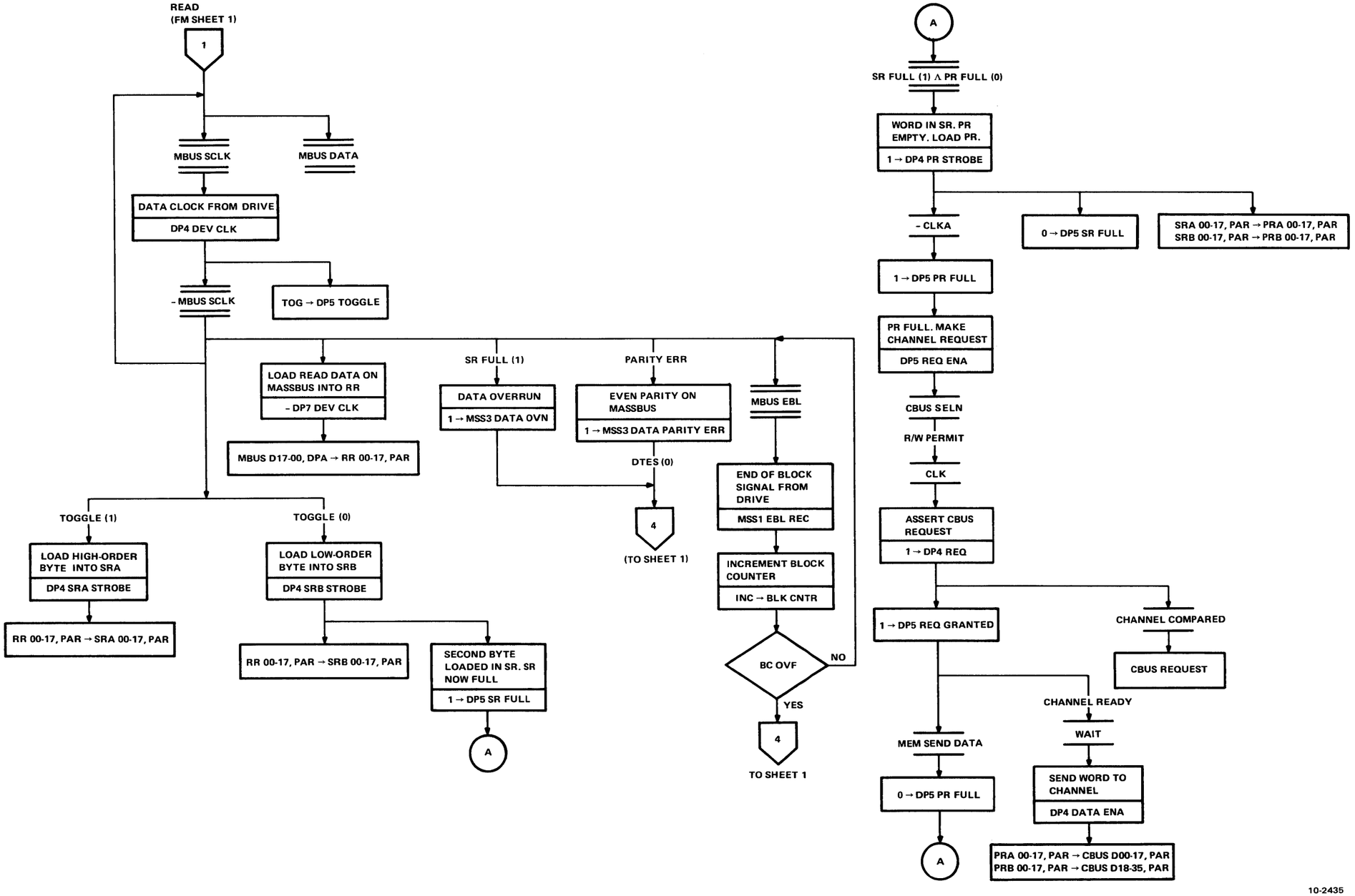
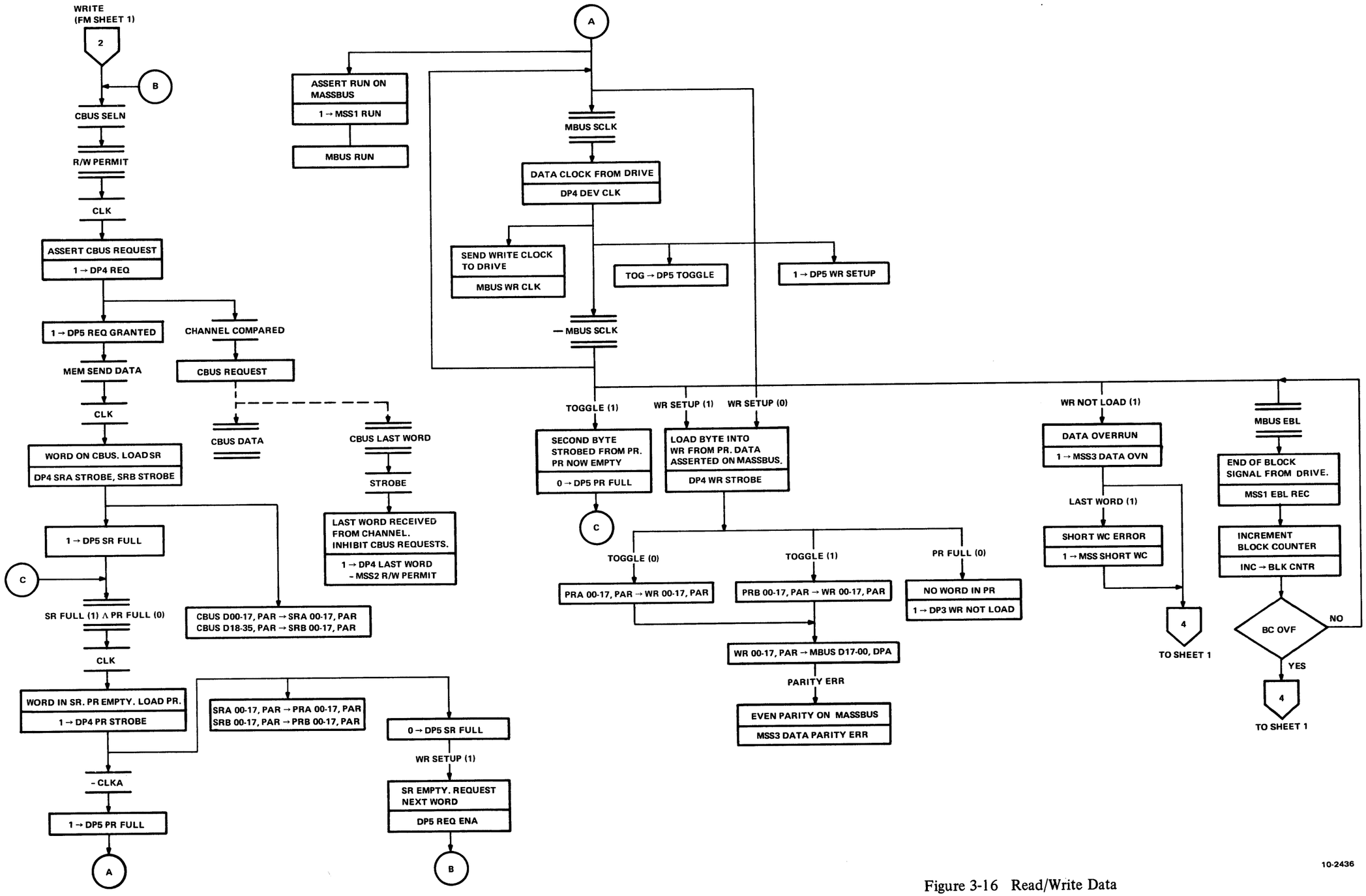


Figure 3-16 Read/Write Data Transfer Flow Diagram (Sheet 1 of 3)



10-2435

Figure 3-16 Read/Write Data Transfer Flow Diagram (Sheet 2 of 3)



10-2436

Figure 3-16 Read/Write Data Transfer Flow Diagram (Sheet 3 of 3)

3.2.5 Write Data Transfer

A timing diagram for the write data transfer is shown in Figure 3-17. After receiving CBUS START (Subsection 3.2.4), and using the address in the command list pointer associated with the RH20, the channel fetches channel control words from memory until the first data transfer control word is retrieved. (The command list pointer is reset, causing an initial control word fetch from word 0 in the controller's assigned EPT reset and status area, if CBUS RESET is asserted together with CBUS START.) After fetching a data transfer control word, the channel begins reading the specified number of data words from memory. When at least one data word is in the channel's data buffer, CBUS READY is asserted by the MBox during the controller's data cycle. The READY signal, once asserted, is negated only after CBUS DONE is asserted by the RH20 at the end of the data transfer and after the channel is prepared to start another data transfer operation.

In the RH20, CBUS READY causes DP4 CHANNEL READY to set. This flip-flop is clocked by DP4 STROBE, asserted every data cycle (Figure 3-15), and remains set as long as CBUS READY is asserted by MBox. CHANNEL READY asserts MSS2 R/W PERMIT, which enables the input to request flip-flop DP4 REQ during select cycles when DP5 REQ ENA is true. At this time, before any words have been collected from the channel, REQ ENA is true because the primary data register (PR) is empty; that is, DP5 PR FULL = 0. Thus, the next DP4 CLK sets the REQ flip-flop, which direct sets DP5 REQ GRANTED and asserts CBUS REQUEST during the request cycle.

During a write operation, CBUS REQ is a request for channel data, causing the channel to place a data word on the CBus data lines during the next data cycle (DP5 MEM SEND DATA = 1). At the trailing edge of the data cycle, with REQ GRANTED set, SR register clocks DP4 SRA STROBE and SRB STROBE are asserted to load the CBus data (the first word) into the SR. The SRB STROBE signal also sets DP5 SR FULL, indicating that the SR has been loaded.

Whenever the SR is loaded (SR FULL = 1) and the PR is empty (PR FULL = 0), as when loading the first word, DP4 PR STROBE is asserted almost immediately (by the next DP4 CLK) to transfer the SR contents to the PR. With the SR now empty and the PR holding a data word, PR STROBE clears SR FULL and sets PR FULL. Setting PR FULL for the first time, when DP5 WR SETUP = 0, asserts DP4 WR STROBE to load the high order byte of the first word (in the PRA) into the WR via the data mixers. (The data mixers are selected by DP5 TOGGLE, which is cleared at this time.) The WR outputs then transmit the first data byte and parity on the Massbus data lines (MBUS D17–00, DPA). PR FULL also asserts RUN on the Massbus and it negates the RQ EN level, preventing more words from being requested from the channel until the first sync clock (SCLK) is received on the Massbus.The SCLK signals are not asserted by a drive until MBUS RUN has been received and the drive is ready to transfer data.

When a SCLK is received (DP7 DEV CLK = 1) during a write operation (MSS1 READ = 0), the RH20 asserts a write clock on the Massbus. The drive then uses the leading of the write clock (MBUS WR CLK) to strobe the write data asserted by the RH20 from the data lines. Returning the sync clocks to a drive as a write data strobe (instead of using SCLK at the drive interface) eliminates the skew between data and data strobe otherwise introduced by the Massbus cable length.

The leading edge of the first SCLK in the RH20 causes a second data word to be requested from the channel. DP7 DEV CLK sets synchronizing flip-flop DP4 DEV CLK, which causes DP4 DEV LOCK and DP4 DEV SYNC to be set, one after the other, by DP4 CLK A. The DEV LOCK flip-flop sets DP5 TOGGLE and DP5 WR SETUP. (The WR SETUP flip-flop remains set for the rest of the data transfer.)

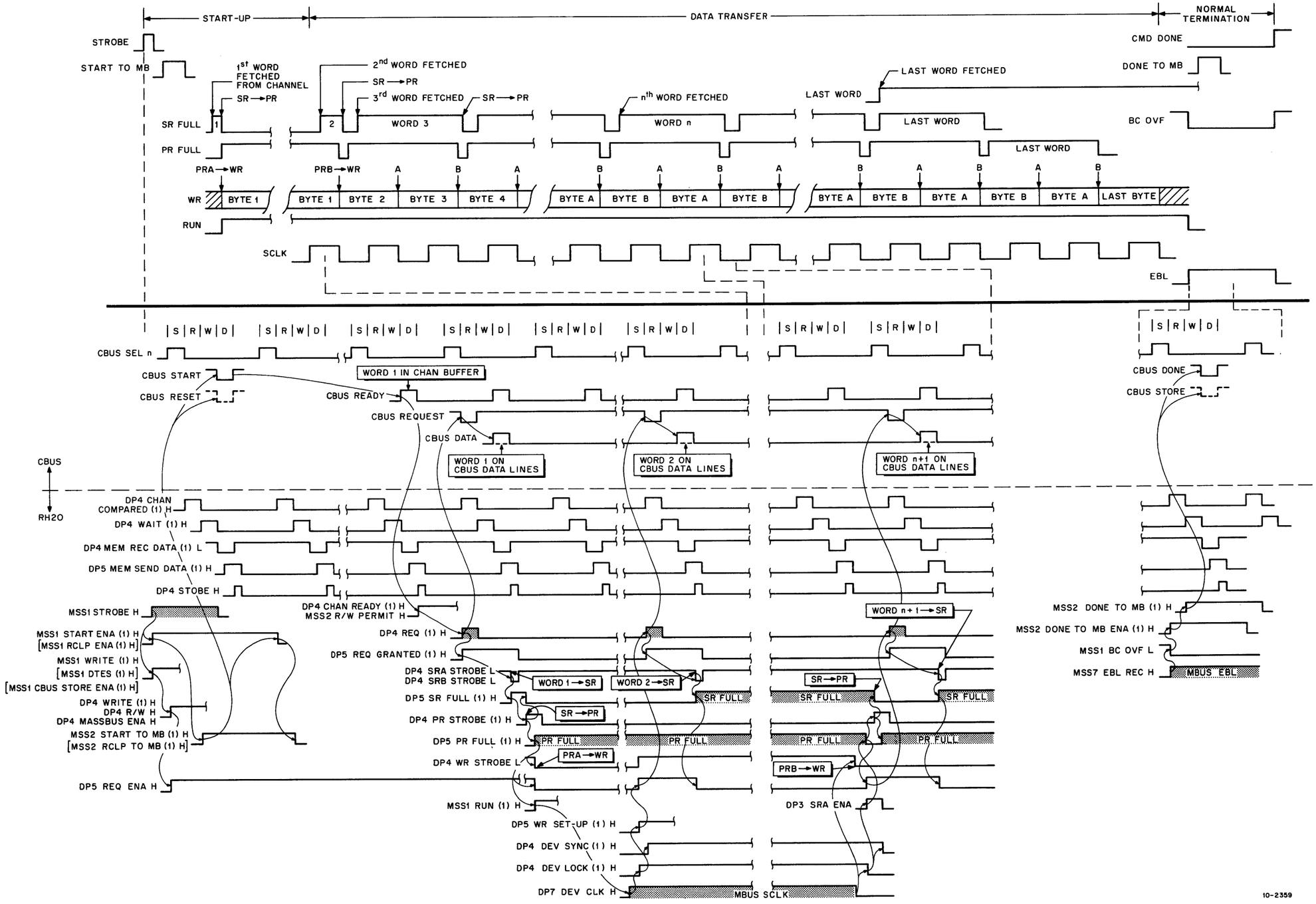


Figure 3-17 Write Data Transfer Timing Diagram

With the SR empty (SR FULL = 0), WR SETUP asserts RQ EN, which sets the REQ flip-flop and asserts CBUS REQUEST for a second time. The MBox then transmits the second data word on the CBus data lines and it is strobed into the SR as explained previously. Unlike the first word, the second word is not necessarily transferred from the SR to the PR immediately. For devices with medium to slow data rates (SCLK duration greater than the time required to fetch a word from the channel) the PR will still be full when the SR is loaded. (The trailing edge of SCLK empties the PR, as explained subsequently.) For faster devices, PR FULL could be cleared when SR FULL is set, which generates PR STROBE and the immediate transfer of SR to PR.

With WR SETUP set, the trailing edge of each SCLK (DP7 DEV CLK = 0) asserts WR STROBE to transfer a half-word from the PR to the WR via the data mixers. When the first SCLK is negated, the second or low-order byte of the first word is loaded in the WR from PRB and transmitted on the Massbus. PRB is selected at the mixers by the TOGGLE flip-flop, which is toggled at the leading edge of every SCLK and true at the trailing edge of the first. Once SCLK is negated, DEV LOCK and DEV SYNC are cleared one after the other by DP4 CLK A. With TOGGLE set, indicating that the second byte has just been transferred out of the PR, DP3 SRA ENA is asserted to clear PR FULL.

As stated previously, except for very fast devices, the SR will be holding data at the trailing edge of SCLK. (Figure 3-17 shows this type of operation.) Clearing PR FULL at the trailing edge of SCLK when SR FULL = 1 immediately generates PR STROBE to refill the PR with the second word. As for the transfer of the first word into the PR, the PR STROBE signal also clears SR FULL. With WR SETUP now set, SR FULL = 0 asserts RQ EN and a channel request is made to refill the SR within the next few channel cycles. For faster devices, when SR is not holding a word at the trailing edge of SCLK, a channel request will be pending (RQ EN = 1) and PR STROBE is delayed until the word is loaded in the SR and SR FULL is set.

The leading edge of the second SCLK clears TOGGLE and the trailing edge causes the first or high-order byte of the second word to be loaded in the WR. The third SCLK sets TOGGLE again and, like the first SCLK, loads the second byte of the word (in the PR) into the WR and clears PR FULL. The PR is then refilled (if or when SR is full), and a channel request is made when the SR goes empty to load another word from the CBus. Operation continues in this fashion with the RH20 requesting channel data, to keep SR and PR full, and alternately asserting the high- and low-order half-words on the Massbus until the channel has sent all the words specified by the command list. When sending the last word, the channel asserts CBUS LAST WORD coincident with the data. This signal, which is asserted once (during the data cycle) and only during the write data transfer, is stored in flip-flop MSS2 LAST WORD. Once asserted, the LAST WORD flip-flop negates R/W PERMIT, preventing further channel requests. The SR and PR are then emptied as the last words are loaded in the WR and transmitted on the Massbus.

After the WR has been loaded with the last data byte, PR FULL will equal 0 when the next WR load occurs at the trailing edge of SCLK. This sets WR NOT LOAD to indicate that all data supplied by the channel has been transferred to the drive. During normal operation, when the number of words is equivalent to the previously loaded block count (channel word count = n × BC, n = block size), WR NOT LOAD has no function. EBL is asserted by the drive, asserting BC OVF, and the transfer terminates normally as described in Subsection 3.2.7. When the number of words do not correspond to the block count, WR NOT LOAD is used to set a transfer error as described in Subsection 3.2.8.

3.2.6 Read Data Transfer

Timing for the read data transfer is shown in Figure 3-18. After receiving CBUS START (Subsection 3.2.4), the channel asserts CBUS READY during the controller's data cycle. It also starts fetching channel control words from memory, using the address in the command list pointer associated with the RH20. (The command list pointer is reset, causing an initial control word fetch from word 0 in the controller's assigned EPT reset and status area, if CBUS RESET is asserted at the same time as CBUS START.) The CBUS READY signal, once asserted, is negated only after CBUS DONE is asserted by the RH20 at the end of the data transfer and after the channel is prepared to start another data transfer operation.

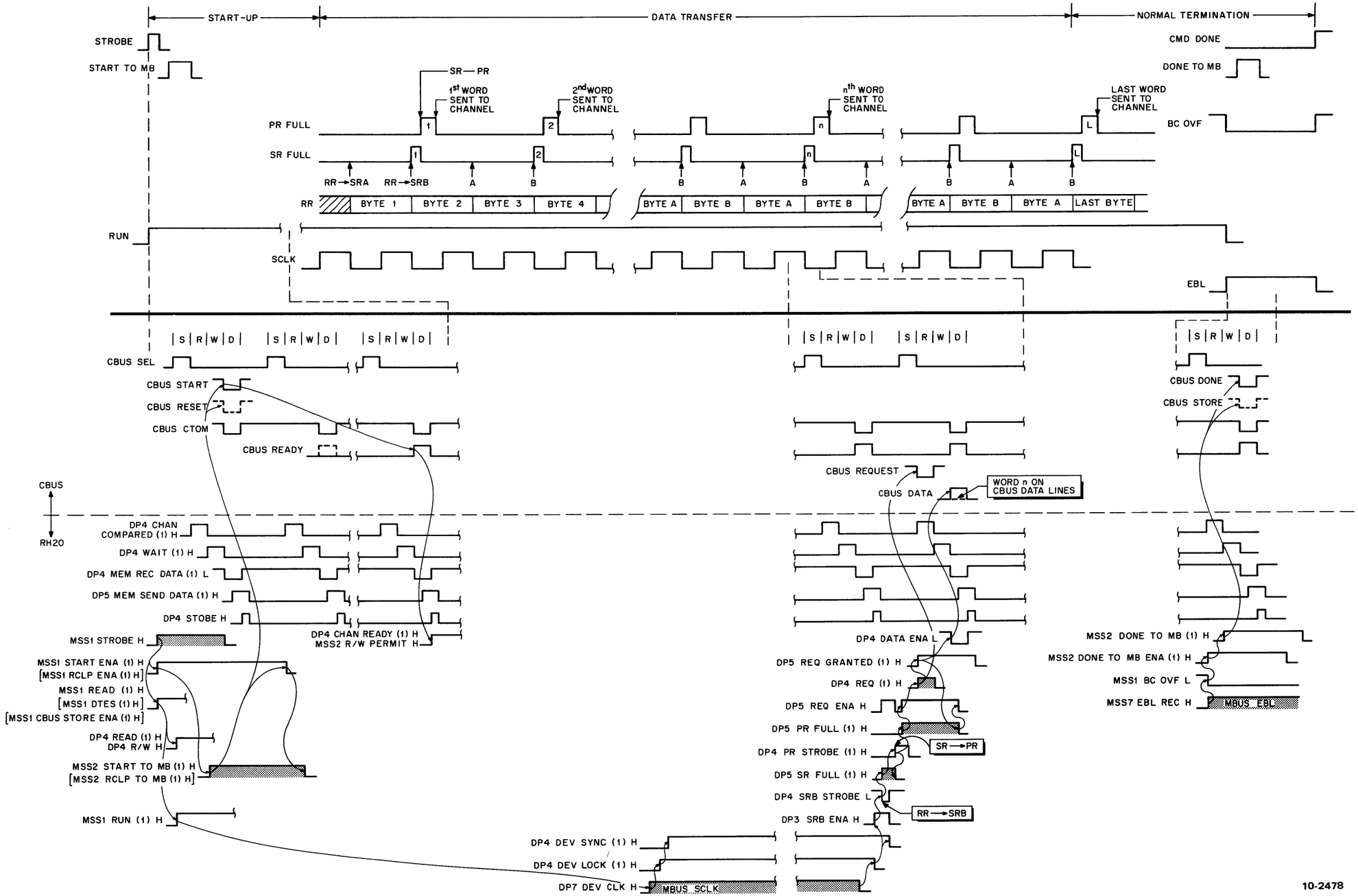


Figure 3-18 Read Data Transfer Timing Diagram

As for a write operation, CBUS READY causes DP4 CHANNEL READY to set in the RH20. CHANNEL READY asserts MSS2 R/W PERMIT, enabling the input to request flip-flop DP4 REQ during select cycles when DP5 REQ ENA is true. During a read operation, REQ EN is not asserted until the PR (or SR) is loaded with Massbus data.

With RUN asserted on the Massbus by MSS1 READ at the start of the data transfer (Subsection 3.2.4), sync clocks (SCLK) and data are asserted on the Massbus when the drive is ready to transfer data, that is, when the search for the addressed sector has completed (disk drives) or the recording medium is up to speed (tape drives). Data is asserted by the drive at the leading edge of SCLK and the data is strobed by the RH20 into the RR at the trailing edge (DP7 DEV CLK = 0). With the RR containing the first byte of data (high-order half of first word) and DP5 TOGGLE true, DP3 SRA ENA is asserted when the first SCLK synchronizing flip-flop DP4 DEV LOCK is cleared (DP4 DEV SYNC still true). (The TOGGLE flip-flop is set by the leading edge of the first SCLK and it is complemented by each SCLK thereafter.) SRA ENA generates DP4 SRA STROBE to clock the RR contents into the SRA. At the trailing edge of the next SCLK, TOGGLE is false and DP3 SRB ENA is asserted to generate DP4 SRB STROBE. This loads the second byte, the low-order half of the first word, into SRB. Because the SR now contains a full word of data, SRB STROBE sets DP4 SR FULL.

SR FULL causes DP4 PR STROBE to be generated when the PR is empty (DP5 PR FULL = 0). PR STROBE loads the PR from the SR, clears SR FULL, and sets PR FULL. The REQ EN level is also asserted and the request flip-flop, DP4 REQ, is set to assert CBUS REQUEST and direct set DP5 REQ GRANTED during the request cycle. When performing a read data transfer, CBUS REQUEST conditions the channel to accept a data word and the word in the PR is transmitted on the CBus data lines during the next data cycle by the DP4 DATA ENA levels. These levels are the outputs of four flip-flops conditioned by REQ GRANTED and the wait cycle (DP4 WAIT = 1), the cycle preceding the data cycle. While the data is asserted on the bus, PR FULL is cleared by time-state DP5 MEM SEND DATA to indicate that the data word has been sent to the channel. As the read data transfer proceeds and more data bytes and sync clocks are asserted on the Massbus, data continues to be clocked into the RR, alternately loaded into the SRA and SRB, and loaded into the PR when the SR contains a full word and the PR is empty. Whenever the PR is filled, the data words are transmitted on the CBus and stored in the data buffers associated with the active channel. When receiving data from Massbus devices with medium to slow data rates, the channel keeps the RH20 data buffers (SR and PR) empty most of the time. For faster devices, where the word rate approaches the repetition rate of channel cycles, data may back up in the SR and PR periodically as the RH20 waits for the next request cycle. Normal termination for a read operation is described in Subsection 3.2.7.

3.2.7 Normal Termination

Normal termination of a data transfer in the RH20 occurs at the end of a data block (sector or record) when the number of blocks specified in the RH20 block counter have been read or written. As explained in Subsection 3.2.2, MSS1 BC OVF is asserted when the end of block signal (EBL) is received from the Massbus device and the counter has overflowed. BC OVF sets MSS2 DONE TO MB ENA, which does the following:

a. Clears MSS1 RUN to negate MBUS RUN. Negating the RUN signal on the Massbus terminates the transfer of data in the drive.

b. Causes MSS2 DONE TO MB to be set by the next wait cycle (DP4 WAIT = 1) if the data transfer is a write operation or if the transfer is a read operation and all data has been sent to the channel. If the RH20 data buffers still hold read data, DP5 REQ ENA (SR FULL or PR FULL = 1) delays the assertion of DONE TO MB until the wait cycle following the transfer of the last word.

Once set, DONE TO MB does the following:

a. Asserts MSS1 CMD CLR to clear MSS1 DTES and MSS1 WRITE OR READ, whichever is set. Clearing MSS1 WRITE OR READ allows DP4 WRITE or READ to be clocked off.

b. Asserts CBUS DONE during the next data cycle (DP4 MEM REC DATA = 1) to signal the channel that the transfer of data is complete. After receiving CBUS DONE, and when finished with the current transfer, the MBox negates CBUS READY causing DP4 CHANNEL READY to be cleared in the RH20.

c. Asserts MSS2 STORE TO MB to assert CBUS STORE if MSS1 RCLP ENA had been set at startup (Subsection 3.2.4). The STORE signal, asserted at the same time as CBUS DONE, causes the channel to store status in words 1 and 2 of the channel's assigned reset and status area in the EPT.

d. Before generating CBUS DONE (and CBUS STORE), sets MSS2 CMD DONE ENA. This generates EBI6 RIGHT CLEAR or LEFT CLEAR, which clears the TCR FULL control flip-flop that is preventing the next command file transfer (Subsection 3.1.9), and it sets MSS2 CMD DONE when the channel negates CBUS READY and when the drive negates MBUS OCC (occupied). As stated previously, CHANNEL READY is negated when the MBox has completed the transfer. The OCC signal, asserted on the Massbus when the drive's control register is loaded with a read/write function code at startup, is negated when the drive has finished all operations associated with the read/write.

e. Clears DONE TO MB ENA during the next request cycle. With its enable level negated, DONE TO MB clears when clocked again during the wait cycle. Thus, the DONE (and STORE) signals are asserted on the CBUS one time only (during a data cycle).

Once CMD DONE has set (CONI: bit 32), indicating that the controller, channel, and drive have all completed the data transfer, it clears CMD DONE ENA and asserts EBI INTR COND to generate an RH20 interrupt. CMD DONE is cleared by the program (CONO = bit 32) when it is servicing the interrupt.

3.2.8 Transfer Error Logic

Whenever the RH20 detects an error associated with a synchronous data transfer, it asserts MSS2 XFER ERR (transfer error) to terminate the operation. XFER ERR negates R/W PERMIT, preventing further channel requests, and it sets MSS2 DONE TO MB ENA. Like a normal termination (Subsection 3.2.7), DONE TO MB ENA negates the RUN level on the Massbus and causes DONE TO MB to be set during the controller's wait cycle. (DONE TO MB is set unconditionally; the RH20 data buffers are not allowed to clear out during a read operation, as when terminating normally.) CBUS DONE (and CBUS STORE, if specified) are then sent to the channel and MSS2 CMD DONE is set to generate an RH20 interrupt when the channel goes inactive (CHANNEL READY = 0) and when the drive terminates the read/write (OCC = 0). With RUN negated, the drive will terminate the read/write at the end of the current block.

The following error conditions cause a transfer error.

a. CHANNEL ERROR – If an error is detected by the MBox during a data transfer, it asserts CBUS ERROR during the controller's data cycle. The error signal is stored in RH20 flip-flop DP4 CBUS ERR, which sets error flag MSS3 CHANNEL ERR. The error flag then asserts XFER ERR. Errors detected in the MBox are:

   1. Memory Parity Error – Incorrect control word parity detected during a memory cycle.

2. Address Parity Error – Incorrect control word address (and request) line parity detected during a memory cycle.

3. Nonexistent Memory – No memory response during a memory cycle.

4. Overrun Error – RH20 sending data faster than MBox can transfer data to memory (read data transfer) or RH20 requesting data faster than MBox can read the data from memory (write data transfer).

5. Last Transfer Error – MBox detected a parity error (data or address) or a nonexistent memory error after negating CBUS READY at the end of the last transfer. (Asserts CBUS ERROR at the start of the next transfer.)

6. Short Word Count Error – More words received from the RH20 than specified by the channel command list; that is, the channel word count equals 0, more data is in the channel's input buffers, and CBUS DONE has not been received from the RH20. Set during a read data transfer only.

7. Long Word Count Error – Less words received from (or requested by) the RH20 than specified by the channel command list; that is, the channel word count does not equal 0 and the input data buffers are empty (read data transfer) or the output data buffers are not empty (write data transfer).

8. RH20 Error – CBUS START received when CBUS READY is true. Caused by hardware malfunction.

NOTE

Channel errors can cause errors in the RH20 when the channel error occurs before the end of a sector or record. Refer to Table 3-2.

b. DRIVE RESPONSE ERROR – As explained in Subsection 3.1.10, MSS2 DRE is set to assert XFER ERR shortly after startup (after a control bus time-out) when a command file transfer attempts to load the read/write function code in a nonexistent drive. When CBUS DONE is received by the channel, it sets a long word count error and negates CBUS READY. This sets CMD DONE in the RH20, interrupting the CPU and flagging the error.

c. DATA PARITY ERROR – When even parity is detected on the Massbus (DP7 MASS D17–00, DPA), parity checker output DP3 PARITY ERR sets parity error flag MSS3 DATA PARITY ERR (CONI = bit 18) at the trailing edge of SCLK (MSS5 DEV CLK = 0). Parity is checked during both the read and write data transfers. During a write, the Massbus receivers are enabled to allow a parity check of the CBus data that is transmitted in the Massbus. XFER ERR is not set, however, the drive should detect the error, asserting ATTN and causing an RH20 interrupt. XFER ERR is only set during the read operation (MSS1 WRITE = 0), when the parity of the incoming drive data is incorrect and MSS1 DTES has not been set at startup (Subsection 3.2). The DTES bit, which is specified by the program when the command file is loaded by a DATAO, prevents termination of a read operation when attempting to retrieve all the data from a disk or tape having a bad spot. Normally, DTES is not set, allowing the first parity error that is detected to assert XFER ERR. Because the error is asserted during a data block (mid-sector or mid-record), the channel will detect a long word count error after receiving CBUS DONE. Also, the RH20 will set a DATA OVERRUN ERROR unless the parity error is detected during the transfer of the last two words in a data block. At the end of the current data block, CMD DONE sets to interrupt the CPU.

Table 3-2 RH20 Errors Caused By Channel Errors

Channel Error	RH20 Error Caused by Channel Error						
	CHAN ERR	DATA PAR ERR	DRE	DATA OVN	LONG WC	SHORT WC	EXC
MEM PAR ERR	x	x (WRITE)		x	x (WRITE)		x (WRITE)
ADR PAR ERR	x	x (WRITE)		x	x (WRITE)		x (WRITE)
NXM	x	x (WRITE)		x	x (WRITE)		x (WRITE)
RH20 ERR	x						
LONG WC	x						
SHORT WC	x	x (WRITE)		x			x (WRITE)
OVN	x	x (WRITE)		x	x (WRITE)		x (WRITE)
LAST XFER ERR	x	x (WRITE)		x	x		x (WRITE)

d. DATA OVERRUN ERROR – MSS3 DATA OVN is set to assert XFER ERR whenever a drive is requesting data faster than the RH20 can fill its buffers or whenever a drive is sending data faster than the RH20 can empty its buffers. The error flip-flop is clocked by the trailing edge of SCLK (MSS5 DEV CLK = 0), at the time that the RH20 either asserts data on the Massbus (write data transfer) or strobes data from the Massbus (read data transfer). During a write operation, DATA OVN is set when WR NOT LOAD is true at the trailing edge of SCLK. WR NOT LOAD is a flip-flop that is also set during a normal termination (Subsection 3.2.7), after the last word has been sent by the channel and by the trailing edge of the last SCLK on the data block. It indicates that the PR was empty when new data was to be clocked into the WR (from the PR). WR NOT LOAD does not set DATA OVN during a normal termination because EBL terminates the transfer and no more SCLKs are received from the drive. However, when an SCLK occurs after WR NOT LOAD has set, it indicates that the channel has stopped sending data, or is not supplying it fast enough, in mid-sector (or mid-record). DATA OVN then sets to flag the error. Once an overrun occurs and XFER ERR is asserted, the remainder of the sector (or record) being written is filled out with 0s. This is because CMD DONE ENA asserts CMD CLR, which clears the WRITE flip-flops and negates the MASSBUS ENA levels at the Massbus data line drivers. Because the parity line driver is also disabled, parity errors will be detected by the drive (DRIVE EXCEPTION ERROR) and the RH20 (DATA PARITY ERROR). At the end of the data block, CMD DONE sets to interrupt the CPU. During a read operation, DATA OVN is set when SR FULL is set at the trailing edge of SCLK. This indicates that both the SR and PR are full and that the channel has either stopped collecting data or it is not collecting data fast enough. When an overrun occurs and XFER ERR sets, drive data continues to be strobed into the RR, one byte overlaying the other, until the end of the data block. CMD DONE then sets to cause a CPU interrupt.

e. SHORT WORD COUNT ERR – MSS2 SHORT WC is set if an overrun in detected by the RH20 (MSS3 DATA OVN = 1) and the channel has asserted CBUS LAST WORD (MSS2 LAST WORD = 1). This error is asserted only during a write data transfer and it indicates that the number of words specified by the channel command list is not equal to a multiple of the drive's block size (channel word count = n × block size, n = 1, 2, 3, etc.). This error should not occur during normal operation because the program can include control words in the command list to fill the remainder of a data block and provide a normal termination. (A control word with ADR = 0 provides the additional word count and fills the data block with the data pattern in EPT locations $60–63_8$).

f. LONG WORD COUNT ERROR – MSS3 LONG WC is set during a write data transfer when the RH20 is terminating the transfer in the channel and drive (MSS2 DONE TO MB = 1) and the channel has not asserted CBUS LAST WORD (MSS2 LAST WORD = 0). This indicates either a programming error, where the block count loaded in the block counter has specified a smaller transfer than that specified by the channel command list, or it indicates an early termination, possibly due to another transfer error.

g. DRIVE EXCEPTION – MSS3 EXC is set if the EXC (exception) line is asserted on the Massbus, indicating that the drive performing the read/write has detected an error, and if EBL has also been asserted by the drive. EBL is asserted shortly after EXC when the error terminates the read/write operation in the drive. It is asserted at the end of the data block, as in normal operation, when the error does not terminate drive operation. If the drive does terminate immediately; that is, if OCC is negated on the Massbus, or if DTES = 0, MSS3 EXC asserts XFER ERR. This asserts CMD DONE to interrupt the CPU and flag the error. The drive also generates an interrupt by asserting ATTN at the termination of the read/write. When DTES is set, and the drive has not terminated because of the error, EXC does not set XFER ERR and the transfer continues until a normal termination occurs. As when disabling data parity errors, this allows the program to transfer as much data as possible during data recovery routines; DTES has no effect when the drive does not terminate normally, that is, when it is terminated by the error. As already stated, XFER ERR will set when OCC goes false.

NOTE

DATA PAR ERR, DRE, DATA OVN, LONG WC, and EXC can cause a LONG WC error in the channel.

3.3 DIAGNOSTIC DATA TRANSFER CONTROL LOGIC

The diagnostic data transfer control contains the logic elements necessary to test RH20 operation without transferring data to/from a Massbus device. Both asynchronous and synchronous data transfers can be tested when addressing a drive that is nonexistent or is powered down.

NOTE

A terminator must be installed in the Massbus port if the RH20 has no terminated Massbus cable connection.

3.3.1 Diagnostic Control Register

The diagnostic control register, or DCR, is loaded by a DATAO (Subsection 3.1.5). It consists of control flip-flops that are set and cleared by the diagnostic program to control and simulate Massbus operation.

Control flip-flops MSS4 DIAG SCLK, ATTN, EBL, and EXC assert Massbus drivers for bus signals that are normally received by the RH20, not transmitted. Because the Massbus receivers for these signals are always enabled, the control flip-flops simulate the input from a Massbus device. For example, the program can test the transfer error and interrupt logic by setting MSS4 DIAG EBL and EXC. Receiver outputs MSS7 EBL REC and EXC are then asserted, as if the signals had been generated by a drive flagging a transfer error, to set MSS3 EXC, assert XFER ERR, and cause an RH20 interrupt.

DIAG TRA EN is set to simulate the TRA signal transmitted by a drive during a control bus cycle. The control flip-flop does not assert a Massbus transmitter directly, as occurs for SCLK, EBL, etc. Instead, it causes MSS1 DEM SYNC to assert the transmitter and generate TRA. If TRA was generated directly by the control flip-flop, control bus time-outs would occur after issuing the DATAO to start a cycle and before issuing the DATAO to simulate the TRA signal.

Control flip-flop DIAG CB TEST asserts the MSS5 CB DATA ENA levels, thus enabling the Massbus (control bus) data line drivers. This allows data loaded in the Massbus buffer register by a DATAO to be read by a DATAI via the Massbus receivers.

DIAG EP TEST forces parity checker output MSS4 CB PAR ERR to be asserted whenever normal (odd) parity is detected on the Massbus (control bus) data lines. The control bit can be used to test parity networks and to force parity errors when reading a drive register. Forcing parity errors allows the program to test register access error circuitry and to generate RH20 interrupts.

DIAG BAR TEST allows the command file transfer of a BAR location to be verified. Normally, when a command file transfer is initiated by a DATAO, a BAR location and then a TCR location are loaded in the Massbus buffer in rapid succession. During diagnostic operation, the last file location loaded in the buffer can be read by a DATAI and its contents checked. Without a DIAG BAR TEST bit, only the TCR (the last location loaded) can be read. The test bit disables the TCR TO MASSBUS input to EBI6 FILE RA and RB and forces the BAR file address during the second file read. Thus, the BAR location is loaded in the buffer twice and its contents can be checked by the program.

DIAG R/W asserts the DP4 MASSBUS ENA levels to enable the Massbus (data bus) data line transmitters. When the control bit is set during a diagnostic read data transfer it allows data loaded in the WR to be transmitted on the data lines and looped back as simulated drive data via the data line receivers, which are always enabled.

3.3.2 Asynchronous Data Transfer

To test the writing and reading of an external register, the program issues a DATAO to set DIAG CB TEST and DIAG TRA ENA in the DCR (Subsection 3.3.1). It then issues a DATAO with an external register address and a drive address that selects a nonexistent (or powered down) device. As in normal operation, the external register data is loaded in the Massbus buffer and asserted on the Massbus (control bus) data lines. Also, a control bus write cycle is started and DEM is asserted on the Massbus. As previously explained, DIAG TRA ENA simulates a drive response by causing the DEMAND signal to transmit a TRA signal on the bus. The received TRA signal then ends the control bus cycle and terminates the DATAO. During normal operation, the external register data loaded in the Massbus buffer by the DATAO is transmitted on the bus only during the control bus cycle. However, with DIAG CB TEST set, the Massbus data transmitters are always enabled, asserting the receivers and providing a data loop. The program can then issue a DATAI (preparation register still holding external register address) to read the simulated read data just loaded by the DATAO. As in normal operation, a control bus read cycle is started and the read data is strobed into the EBI buffer when TRA is generated. As for the external register write operation in diagnostic mode, the simulated TRA signal is generated by DEM to end the control bus cycle and terminate the operation.

To test the command file transfer operation, the program must also set DIAG CB TEST and DIAG TRA ENA in the DCR. It then issues DATAO commands to load the command file. As in normal operation, loading the STCR causes a command file transfer of the SBAR (if loaded) and the STCR. Data is read from the file, loaded in the Massbus buffer, and the write control bus cycles that are initiated are terminated by the simulated TRA signals. The TCR is loaded in the Massbus buffer last and its contents can be read via the loopback path by loading the preparation with an external register address and issuing a DATAI. As explained previously, if the BAR file is to be checked, DIAG BAR TEST is set in the DCR prior to initiating the command file transfer. The BAR file is then loaded in the Massbus buffer last and can be read with the DATAI.

3.3.3 Synchronous Data Transfer

To perform a diagnostic write data transfer, the program sets up a command list in memory, sets DIAG TRA ENA in the DCR (Subsection 3.3.1), and loads the command file to start the operation. As in normal operation, the first data word is loaded in the SR and transferred to the PR. Also, the first half-word is loaded in the WR and transmitted on the Massbus data lines. The RH20 then hangs waiting for an SCLK. To step the data transfer, the program sets and then clears DIAG SCLK to simulate the input from a drive. This causes more words to be fetched from the channel and more data bytes to be transmitted on the bus. The write data is asserted by the Massbus receivers and it is loaded in the RR by the SCLK signal to provide a data loopback path. (The RR may be read by a DATAI.) The program terminates the operation by setting DIAG EBL at the appropriate time to simulate the end of the data block.

Similar to the write operation, the program starts a diagnostic read data transfer by setting up a command list and loading the command file with DIAG TRA ENA set. Again, the RH20 hangs waiting for an SCLK. To provide simulated read data, the program can load the WR with DIAG R/W set. The data is then asserted by the Massbus receivers and clocked into the RR when SCLK is generated by the program. Setting and clearing SCLK steps the read operation, causing the data that is clocked into the RR (specified by the contents of the WR) to be loaded in the SR and PR and then sent to the channel. As for the write, EBL terminates the operation.

APPENDIX A
ABBREVIATIONS AND MNEMONICS

ACKN - Acknowledge
ADR - Address
API - Automatic Priority Interrupt
ATA - Attention
ATTN - Attention
BAR - Block Address Register
BC - Block Counter
BUF - Buffer
C - Massbus (Control Bus) Data
CB - Control Bus
CBEP - Control Bus Even Parity
CBUS - Channel Bus
CHAN - Channel
CLK - Clock
CLR - Clear
CMD - Command
COMP - Compare
CONI - Conditions In
CONO - Conditions Out
CPA - Massbus (Control Bus Parity
CRY - Carry
CS - Controller Select
CTOD - Controller to Drive
CTOM - Controller to Memory
D - EBus Data/Massbus (Data Bus) Data
DATAI - Data In
DATAO - Data Out
DC - Device Code
DCR - Diagnostic Control Register
DEM - Demand
DEV - Device
DIAG - Diagnostic
DPA - Massbus (Data Bus) Parity
DRAES - Disable Register Access Error Stop
DRE - Drive Response Error
DS - Drive Select
DTES - Disable Transfer Error Stop
EBD - EBus Data
EBI - EBus Interface/EBus Interface Buffer
EBL - End of Block
EBus - Execution Bus
ECC - Error Correction Code
ENA - Enable
ENB - Enable
EP - Even Parity
EPT - Executive Process Table
ERR - Error
EXC - Exception
EXT - External
F - Function
FILE - Command File
GEN - Generator
GND - Ground
INH - Inhibit
INIT - Initialize
INT - Internal
INTR - Interrupt
I/O - Input/Output
IVIR - Interrupt Vector Index Register
LR - Load Register
MASS - Massbus
MASSBUS - Mass Storage (device) Bus
MB - MBox
MBC - Massbus Controller
MBUS - Massbus
MEM - Memory
MIX - Mixer
MIXB - Mixer B
MSS - Massbus Control
OCC - Occupied
OVN - Overrun
PBAR - Primary Block Address Register
PCR - Primary Control Register
PHY - Physical
PI - Priority Interrupt
PIA - Priority Interrupt Assignment
PIR - Priority Interrupt Request
PR/PRA//PRB - Primary (Buffer) Register
PREP - Preparation
PTCR - Primary Transfer Control Register
R - Register
RA - Low-Order File Read Address

RAE – Register Access Error
RB – High-Order File Read Address
RCLP – Reset Command List Pointer
REC – Receive
REG – Register
REQ – Request
RR – Read Register
RS – Register Select
R/W – Read/Write
RY – Ready
SBAR – Secondary Block Address Register
SCLK – Sync Clock
SCR – Secondary Control Register
SEL – Select
SR/SRA/SRB – Secondary (Buffer) Register
SREG – Secondary Register
STB – Strobe
STCR – Secondary Transfer Control Register
SYN – Sync
T – Time
TCR – Transfer Control Register
TRA – Transfer
WC – Word Count
WCLK – Write Clock
WR – Write Register
XFER – Transfer

INDEX

Reader's Comments

RH20 MASSBUS CONTROLLER UNIT DESCRIPTION
EK-RH20-UD-001

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? _____

_____

_____

_____

What features are most useful? _____

_____

_____

_____

What faults do you find with the manual? _____

_____

_____

_____

Does this manual satisfy the need you think it was intended to satisfy? _____

Does it satisfy *your* needs? _____ Why? _____

_____

_____

_____

Would you please indicate any factual errors you have found. _____

_____

_____

_____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

CUT OUT ON DOTTED LINE

— — — — — — — — — — — — Fold Here — — — — — — — — — — — —

— — — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

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

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
146 Main Street
Maynard, Massachusetts 01754