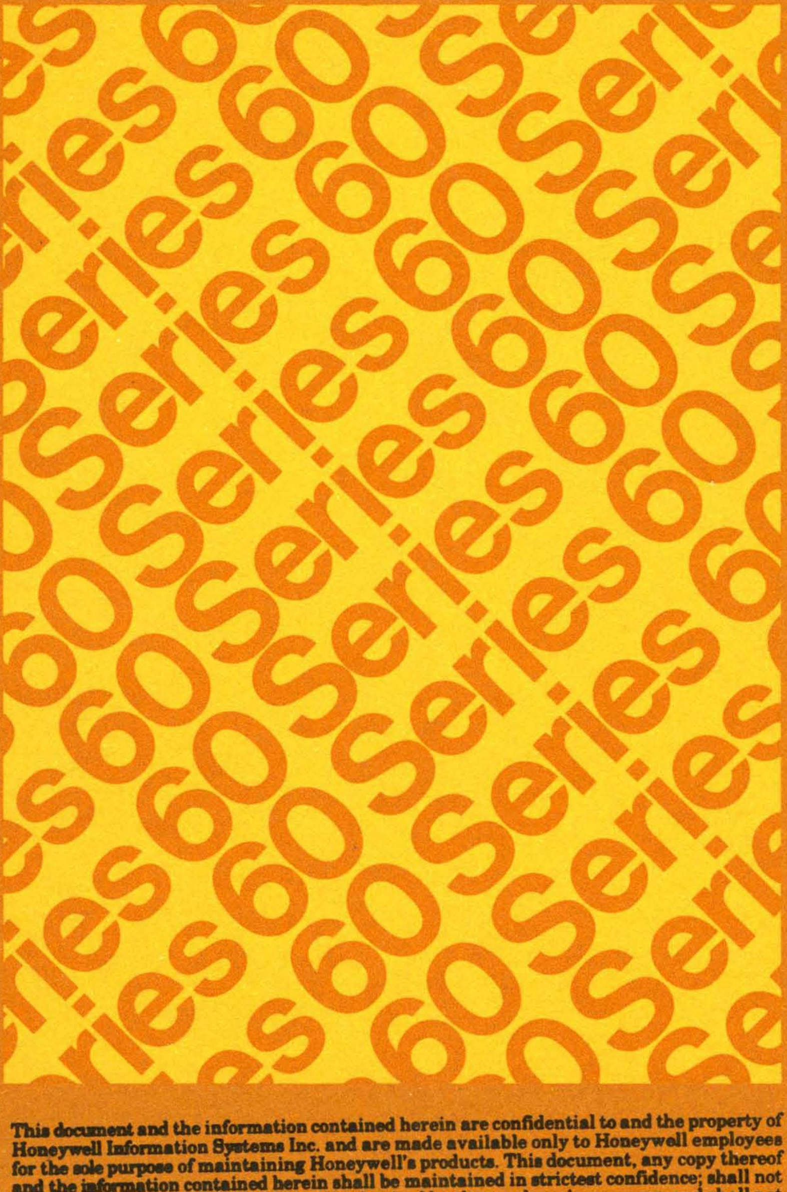
Honeywell



LEVEL 6

HARDWARE

TYPE MTC9101 MAGNETIC TAPE CONTROLLER MANUAL

TYPE MTC9101

MAGNETIC TAPE CONTROLLER MANUAL

Document No. 71010425-200   Order No. FM88, Rev. 1

This manual has been revised to the -200 level.
It supersedes all previous issues.

RECORD OF REVISIONS

REVISION	DATE	AUTHORITY	AFFECTED PAGES
-100	Dec. 1976	Interim Issue	--
-200	Aug. 1977	BLCO 70311	All

Hardware Publications, M&TO, Billerica, MA 01821

CONTENTS

CONTENTS

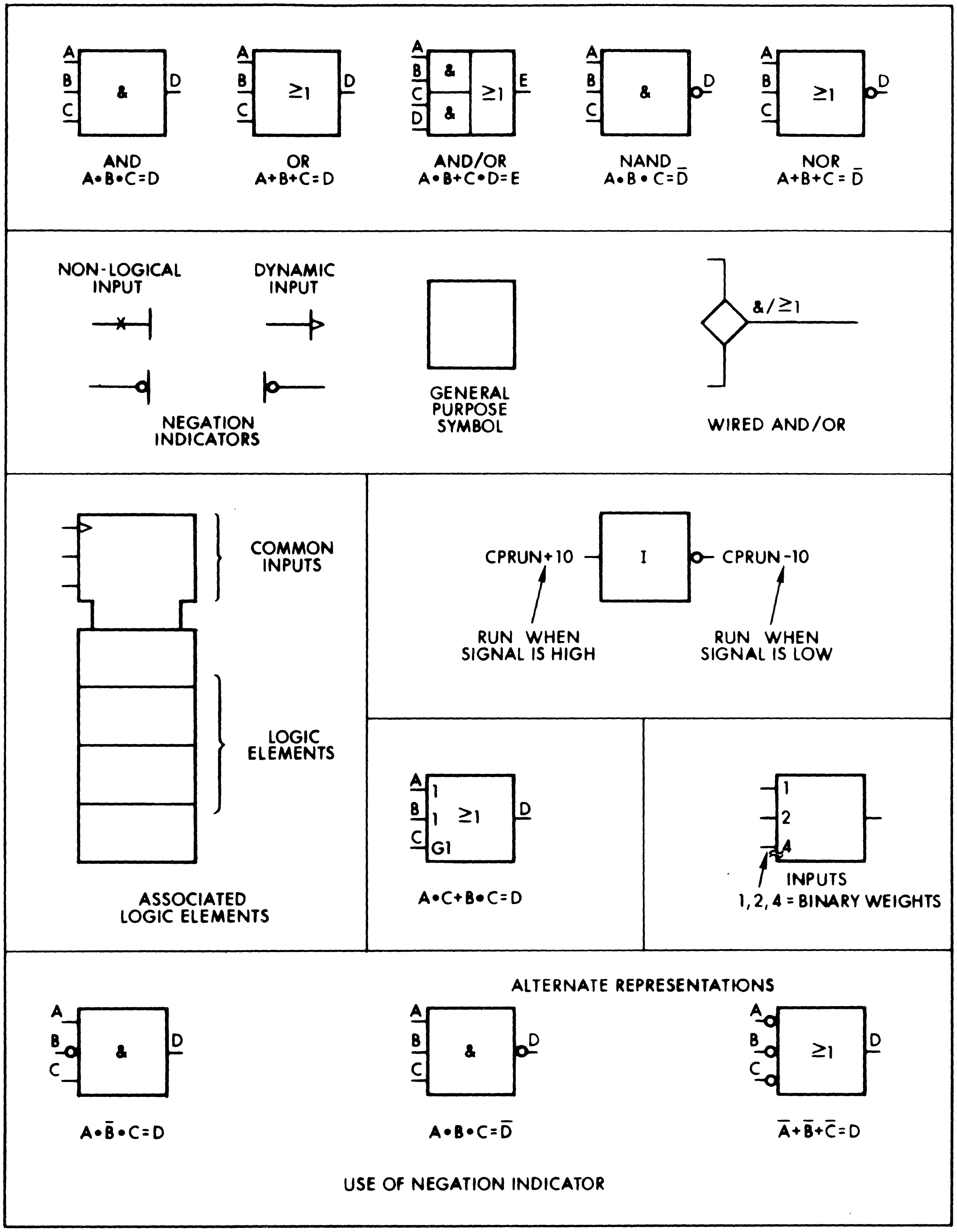
CONTENTS

Section Page

ILLUSTRATIONS

TABLES

LOGIC SYMBOLOGY



I
INTRODUCTION

This product manual describes the functionality and operation of the Type MTC9101 Magnetic Tape Controller (MTC). Programming considerations are included to aid in understanding the hardware and firmware* descriptions. Detailed programming information is contained in the Minicomputer Handbook (Order No. AS22) or the Peripherals Handbook (Order No. AT04).

Operational theory within this manual is designed to acquaint the user with the major functional hardware and firmware elements in order to aid in analyzing the MTC operation at the detail presented in the logic block diagrams (LBDs).

Detailed theory for the MTC hardware logic is related directly to the maintenance philosophy of board replacement rather than on-site board repair.

This product manual is comprised of this section and Sections II through IV as follows:

- Section II - Theory of Operation - Overview
  This section contains a brief description of the interrelationship between software and the MTC hardware and firmware, and supplies an operational overview of the functional areas.

---

*This manual supports firmware Rev. 24.

- Section III - Theory of Operation - Intermediate
  This section contains the intermediate theory of operation for each area discussed in Section II. Function names are included in many places to aid entry into LBDs supplied in the Type MTC9101 Magnetic Tape Controller Reference Manual (Order No. FM89).

- Section IV - Theory of Operation - Cycle Flow
  This section contains a description of the MTC micro-operations. An overview flow chart illustrates MTC cycle groupings with their entries and exits.

1.1 GENERAL DESCRIPTION

The Type MTC9101 Magnetic Tape Controller (subsequently referred to as the MTC or controller) is a solid state module (BF4MTC) which, in conjunction with seven or nine channel tape adapters, operates up to four daisy-chained tape devices in a Model 6/34, 6/36, or 6/40 configuration of the Series 60 Level 6 computer system. In subsystem configurations of three or fewer tape devices, the controller allows for the attachment of up to two low-speed devices/adapters. The devices and adapters for the MTC subsystem are listed in Table 1-1. The magnetic tape subsystem, whose attachment configuration is illustrated in Figure 1-1, uses 1/2-inch magnetic tape for storage and retrieval of Non-Return-to-Zero (NRZI) formatted data. Regardless of the number of tape devices connected to the subsystem, only one tape adapter can be utilized with a single MTC.

The MTC consists of dual in-line packages (DIPs) mounted on a multilayer Series 60 Level 6 printed wiring assembly module. The MTC provides eight 25-pin in-line connectors for attachment of adapters, and two 50-pin connectors for attachment to the Series 60 Level 6 Megabus.

1.2 FUNCTIONAL CHARACTERISTICS

The MTC is a microprogrammed peripheral device control unit which, together with the tape adapter, supports from one to four tape devices. Much of the microprocessor portion of the MTC is designed to facilitate its application as a control unit for various unit record and magnetic tape device adapters. The MTC performs the following general purpose control functions.

- Execution of the Series 60 Level 6 Megabus network sequences
- Command decoding
- Status and control register storage
- Data transfer multiplexing to/from device adapters
- Direction of the general flow of command execution.

Table 1-1 MTC Attachable Devices/Adapters

DEVICES			ADAPTERS	
DEVICE	TYPE NO.	DESCRIPTION	TYPE NO.	DESCRIPTION
Tape Drive (NRZI)	MTU9104 MTU9105	45 IPS, 9 Channel, 800 BPI 75 IPS, 9 Channel, 800 BPI	MTM9103	9-Channel NRZI Tape Adapter
	MTU9112 MTU9113	45 IPS, 7 Channel 800/556 BPI 75 IPS, 7 Channel, 800/556 BPI	MTM9101	7-Channel NRZI Tape Adapter
Card Reader (1)	CRU9101 CRU9102 CRU9103 CRU9104	300 CPM 300 CPM with Mark Sense 500 CPM 500 CPM with Mark Sense	CRM9101	Card Reader Adapter
Line Printer (2)	PRU9103 PRU9104 PRU9105 PRU9106	240 LPM, 96 Characters 300 LPM, 64 Characters 480 LPM, 96 Characters 600 LPM, 64 Characters	PRM9101	Printer Adapter
Serial Printer	PRU9101 PRU9102	64 Characters 96 Characters		

(1) CRF9101 51-column card option available
(2) PRF9102 vertical format unit option available

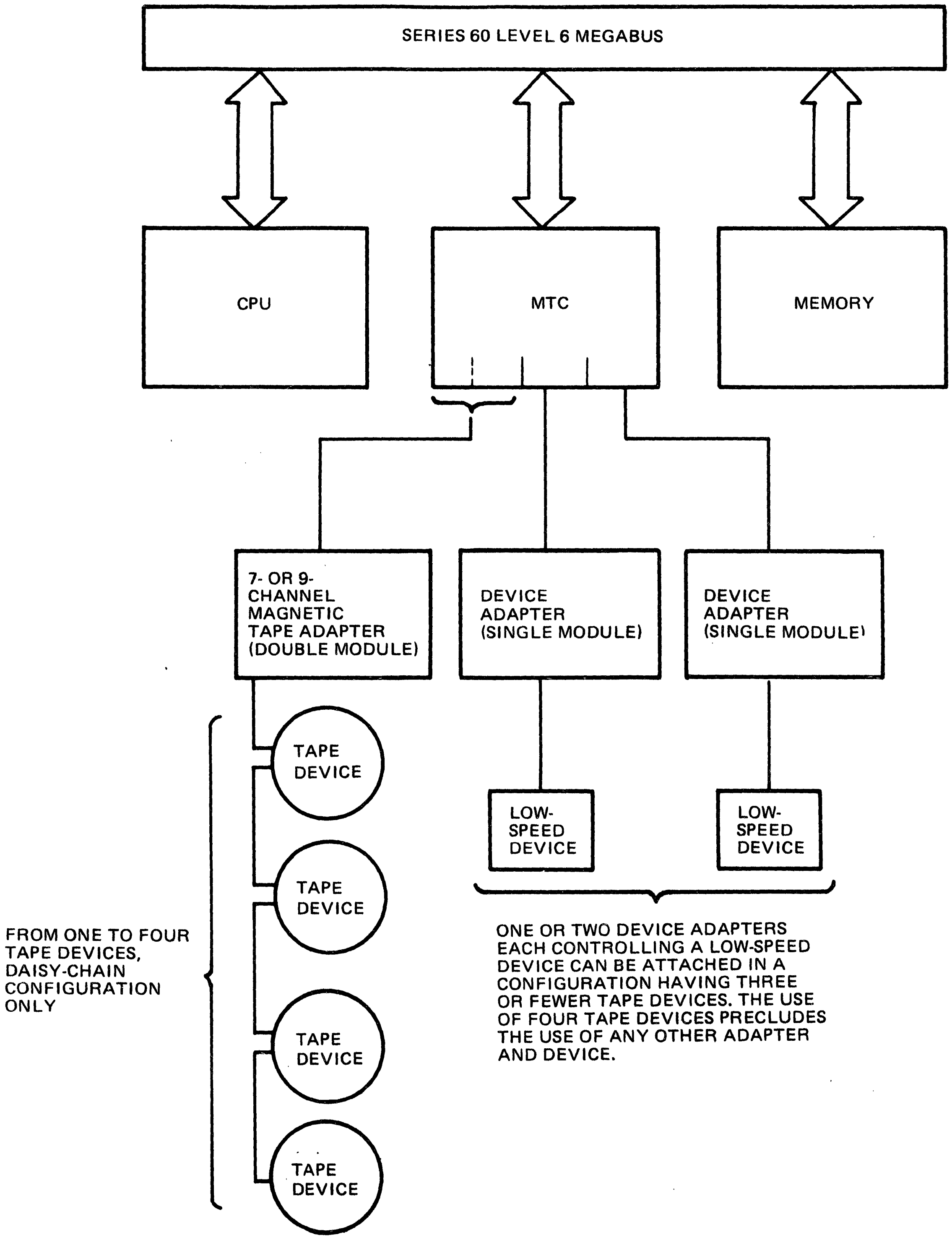
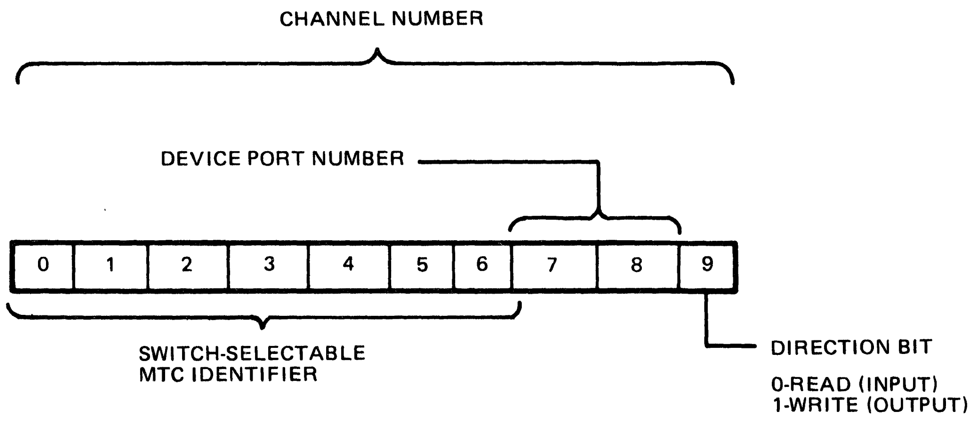

SERIES 60 LEVEL 6 MEGABUS
CPU
MTC
MEMORY
7- OR 9-CHANNEL MAGNETIC TAPE ADAPTER (DOUBLE MODULE)
DEVICE ADAPTER (SINGLE MODULE)
DEVICE ADAPTER (SINGLE MODULE)
TAPE DEVICE
TAPE DEVICE
TAPE DEVICE
TAPE DEVICE
LOW-SPEED DEVICE
LOW-SPEED DEVICE
FROM ONE TO FOUR TAPE DEVICES, DAISY-CHAIN CONFIGURATION ONLY
ONE OR TWO DEVICE ADAPTERS EACH CONTROLLING A LOW-SPEED DEVICE CAN BE ATTACHED IN A CONFIGURATION HAVING THREE OR FEWER TAPE DEVICES. THE USE OF FOUR TAPE DEVICES PRECLUDES THE USE OF ANY OTHER ADAPTER AND DEVICE.

Figure 1-1 MTC System Block Diagram

1.3 MTC SUBSYSTEM OPERATIONAL SUMMARY

Each device attached to the MTC by way of an adapter is addressable by software via channel numbers. A device has two channel numbers assigned, the numbers differing only in the low order bit position (called the direction bit). When an I/O Load (IOLD) command for the MTC channel is accepted, the direction bit of the channel number specifies whether the command designates an input or output data transfer. The direction bit of a subsequently accepted Output Task Word command is examined to verify that it agrees with the IOLD's direction bit. The following example indicates the composition of a channel number, where bits 0 through 6 are assigned at the time of system installation.



Software usability of the devices attached to the MTC is such that the devices are independent of one another. For example, operations on one device are, in general, independent of the activity of any other device except that the MTC can stall the initiation of a command sequence addressed to one device (channel number) while the it is servicing another device. This apparent device independence looks, to software, as if multiple levels of simultaneity exist.

The tape subsystem command sequencing results in any command addressed to a nonbusy channel being accepted, allowing the MTC to accept any command to a tape device while another tape device is executing a data transfer. The accepted command is not invoked until the present data transfer is completed. Stored commands are executed in the sequence in which they are received, but rewinds/unloads may be executed concurrently with data transfer. Because channels are serviced on a rotating priority basis, no one channel dominates adapter usage.

If an MTC has fewer than four devices attached, it responds only to the channel numbers associated with the installed devices. When a configuration of tape devices is three or fewer, up to two unit record adapters can be implemented by the MTC.

To allow for device access, the MTC contains a set of software-loadable scratch pad memory locations (refer to Section IV of this manual) assigned to each device which contain parameters and control information required for device operation. In addition to range and address locations, there is a configuration location which contains the mode of operation information, and a task word location which stores the command codes.

To perform a specific operation, the software first loads the configuration, then address and range into the scratch pad memory. The task word, which is loaded last, designates the operation to be performed. Upon receipt of the task word, execution of the command is initiated.

Commands addressed to a nonbusy device (channel) are always accepted, although their execution can be delayed as previously described. All commands addressed to a busy tape device are rejected (NAK response on the Megabus) with the exception of the Output Control Word command.

1.4 INTERFACES

Two interface networks are associated with the MTC. They are the Megabus/MTC interface and the MTC/device adapter interface (see Figure 1-2). For a detailed description of these interfaces, refer to Section III.

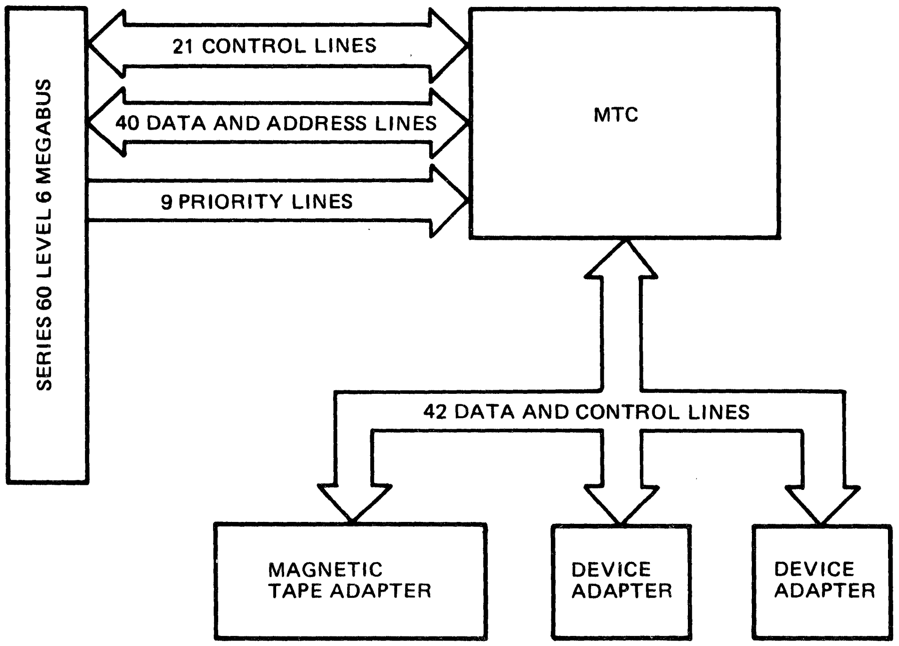


Figure 1-2 MTC Interfaces

1.5 REFERENCE DOCUMENTS

The information contained within the following documents supplements the contents of this manual.

Title	Document No.	Order No.
Model 34/36 System Manual	71010200	FL35
Model 34/36 CP Manual	71010201	FL36
Model 06 System Manual	71010210	FL37
Model 06 CP Manual	71010211	FL38
Type MTM9102 9-Channel NRZI Tape Adapter Manual	71010282	FN18
Type MTM9101 7-Channel NRZI Tape Adapter Manual	71010286	FN20
Type MTM9102 9-Channel NRZI Tape Adapter Reference Manual (Assembly No. 60130316-001)	71010283	FN19
Type MTM9102 9-Channel NRZI Tape Adapter Reference Manual (Assembly No. 60130975-001)	71010564	FP44
Type MTM9101 7-Channel NRZI Tape Adapter Reference Manual	71010287	FN21
Type PRM9101 Printer Adapter Manual	71010221	FL18
Type CRM9101 Card Reader Adapter Manual	71010222	FL13
Power Systems Manual	71010290	FL34
Level 6 Minicomputer Handbook	--	AS22
Peripherals Handbook	--	AT04
Type PRM9101 Printer Adapter Reference Manual (Assembly No. 60127843-001)	71010371	FL27
Type PRM9101 Printer Adapter Reference Manual (Assembly No. 60127843-003)	71010376	FM40

Type CRM9101 Card Reader Adapter Reference Manual (Assembly No. 60127840-003)	71010377	FM37
Type MTC9101 Magnetic Tape Controller Reference Manual	71010426	FM89

II THEORY OF OPERATION - OVERVIEW

The MTC, in conjunction with adapters and a variable configuration of devices (the maximum being four), is a firmware-operated peripheral device controller. The MTC allows for attachment of adapters which enable apparent simultaneous control of the attached devices. The MTC hardware, which implements data transfers and control sequences to and from the adapters, is divided into seven fundamental logic areas (see Figure 2-1):

- Microprogram Control Store
- Arithmetic Logic Unit and Accumulator
- Scratch Pad Memory
- Megabus Logic
- Adapter Logic
- Test Multiplexer
- Clock Logic.

These hardware components, in addition to firmware and software, provide for data transfer and efficient control of device operations. During the discussions in the remainder of this manual, only the tape adapter subsystem is considered unless otherwise noted.

2.1 SOFTWARE

The MTC operations are a direct result of an input or output command from the central processor. Table 2-1 lists the input, output, and diagnostic commands applicable to the MTC subsystem. These commands load address, range, and control

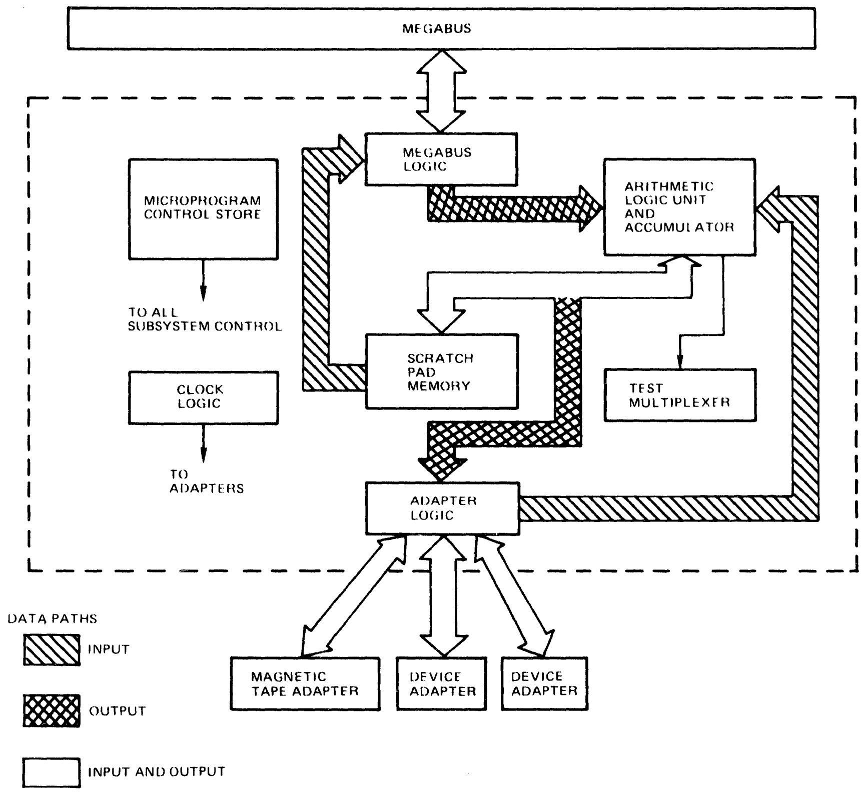


Figure 2-1 MTC Hardware Major Block Diagram

information into a device-specific segment of the scratch pad memory (SPM).

For output commands, the general format of the Megabus address and data lines is as shown in Figure 2-2, part A. The address lines reflect the channel number of the controller being addressed and a function code indicating the purpose of the information on the data lines. The data lines may have control information, address, or control words to be loaded into the MTC scratch pad memory.

For input commands, the general format of the Megabus address and data lines is as shown in Figure 2-2, part B. The address lines reflect the channel number of the controller being addressed and the appropriate function code. The data lines have the number of the channel initiating the command. In response to the input command, the MTC loads the bus address lines with the channel number of the unit that is to receive the information on the data lines.

Device operations are the result of device commands contained in the command code field of the task word. The task word is

Table 2-1 I/O Commands

TYPE COMMAND	FUNCTION CODE (HEX)	INSTRUCTION	ADAPTER APPLICATION
Output	01	Control Word	Magnetic Tape*, Card Reader, & Printer
	03	Interrupt Control	Magnetic Tape, Card Reader, & Printer
	07	Task Word	Magnetic Tape, Printer
	09	I/O Load Address	Magnetic Tape, Card Reader, & Printer
	0D	I/O Load Range	Magnetic Tape, Card Reader, & Printer
	11	Configuration Word	Magnetic Tape & Card Reader
Input	02	Interrupt Control	Magnetic Tape, Card Reader, & Printer
	06	Task Word	Magnetic Tape, Printer
	08	Memory Byte Address	Magnetic Tape, Card Reader & Printer
	0A	Memory Module Address	Magnetic Tape, Card Reader & Printer
	0C	Range	Magnetic Tape, Printer, & Card Reader
	10	Configuration Word	Magnetic Tape & Card Reader
	18	Status Word 1	Magnetic Tape, Printer, & Card Reader
	1A	Status Word 2	Magnetic Tape
	26	Identification Code	Magnetic Tape, Printer, & Card Reader

*Both 7- and 9-channel magnetic tape

located in the segment of the SPM dedicated to the device and is loaded there by the Output Task Word I/O command. For information pertaining to the commands that each adapter can perform, refer to the appropriate adapter manual.

2.2 FIRMWARE

Firmware, a sequence of microinstructions resident in the microprogram control store, is the primary control element of the MTC. The main function of the firmware is to interpret external and internal events or conditions and to react in a prescribed manner (i.e., setting or resetting of hardware functions). Efficient data transfer is also a result of firmware control of hardware components in the data path.

Each time the MTC is cleared, a series of firmware operations called the Quality Logic Test is performed. These operations are used to verify the integrity of the MTC hardware. Upon successful completion of the Quality Logic Test, the firmware sets up the hardware for execution of software commands. This is accomplished by setting up control information in the read/write memory and enabling the available adapters. Then the firmware enters a routine and waits for a request from the Megabus or from one of the adapters. When a request occurs, firmware analyzes status and priority before proceeding to the appropriate firmware routine for processing the request. Refer to Section IV for detailed information.

2.3 HARDWARE

The MTC hardware (see Figure 2-1) is organized into seven fundamental logic areas as described in the following subsections. Although each of these major areas is divided into

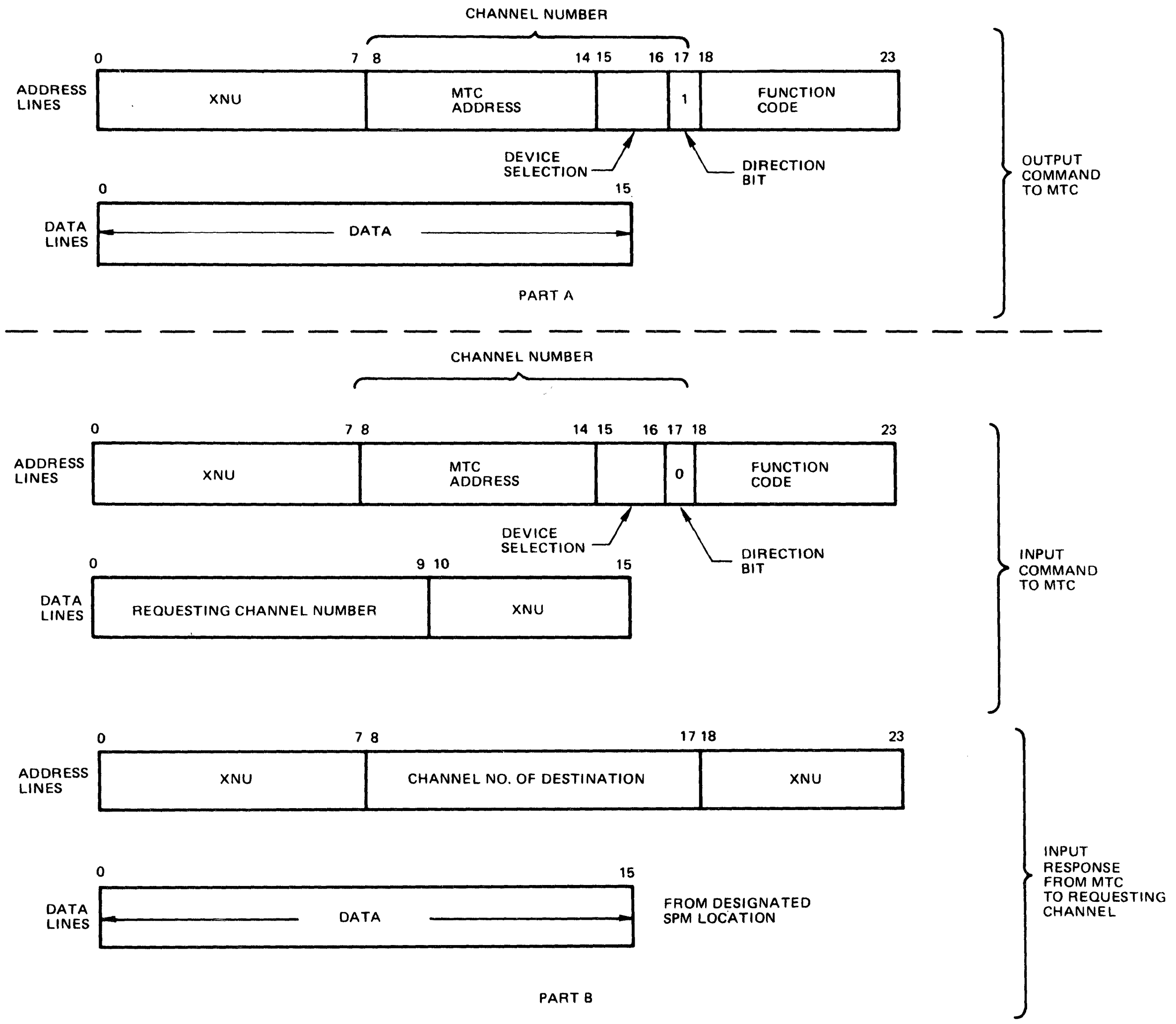


Figure 2-2 Megabus Configuration for Transferring Data to/from the MTC SPM

Although each of these major areas is divided into various hardware elements, only an overview of each is described in this section; for detailed information, refer to Section III.

2.3.1 Microprogram Control Store

The microprogram control store (UPCS) provides permanent storage for resident control firmware and diagnostic microprograms. The UPCS is enabled during normal operation when the MTC is not in the test mode. It varies address sequencing in order to execute appropriate routines depending upon priority information, test conditions, or channel activity.

The UPCS is configured of either 2K or 4K PROM chips. The internal array of each 2K PROM* chip is 512 locations by 4 bits wide, providing a maximum of 2,048 bits. The 2K PROM chips are aligned in rows of 4 to create a UPCS output 16 bits wide. The UPCS with 2K PROM chips can be expanded to a maximum of 2K word locations.

The internal array of each 4K PROM* chip is 1,024 locations by 4 bits wide, providing a maximum of 4,096 bits. The 4K PROM chips are aligned in rows of 4 to establish a UPCS output 16 bits wide. The UPCS with 4K PROM chips is expandable to a maximum of 4K word locations.

2.3.2 Arithmetic Logic Unit (ALU)/Accumulator (ACU)

The ALU is the focal point of all data operations within the MTC, between the MTC and the device adapters, and between the MTC and the Megabus. Two input multiplexers (A-operand and B-operand) to the ALU allow various combinations of registers to be used as operands within the ALU.

The ALU is capable of performing 8-bit arithmetic and logic operations on the outputs of the two multiplexers and determining the destination of the result as a function of the firmware. ALU status is generated as a result of an ALU operation and is stored in ALU status flip-flops for interrogation, or until the initiation of the subsequent ALU firmware command.

The ALU is also able to perform bit operations on one input with a data constant supplied by firmware. The constant can be loaded into the ALU by way of the B-operand multiplexer.

Through the use of multiple byte procedures, word mode operations are performed by firmware and the ALU. In a multiple byte procedure, ALU status (i.e., carry-out) is set each time a byte is operated on, and this status is used to determine the operation required for the subsequent byte.

The ACU is an 8-bit register used for temporary storage of the ALU outputs. Shift operations (left shift only) are performed using the ALU capabilities.

2.3.3 Scratch Pad Memory (SPM)

The SPM is a 256-location by 8-bit wide read/write memory used for storage of information required by or generated by each channel (i.e., data, status, commands, etc.). The SPM is segmented into four parts (64 locations each) with one part designated for each channel. The SPM is addressed by eight bits; the two high-order bits select the correct segment for the

---

*The hex rotary switch on the controller is set to 0 for 2K PROM chips, or to F for 4K PROM chips.

channel which is active, and the remaining six select location within the segment.

Data from the ALU A-operand multiplexer is written into the SPM during a firmware Memory Write command at the location specified by the address bits. Because the SPM uses the ALU A-operand multiplexer as the data input, all firmware-visible registers can be implemented as an input local register. The data out of the SPM is delivered to either one of the ALU operand multiplexers and can be subsequently distributed throughout the MTC.

2.3.4 Megabus Logic

The Megabus logic provides an interconnection between the Megabus and each channel on the MTC. Although some of the logic elements are dedicated to an individual channel, the Megabus hardware is primarily common to all the channels.

When Megabus cycles designated for the MTC are detected by a Megabus logic decoder, the response logic is enabled. If the channel being addressed has an adapter installed, a response is generated; otherwise, no response is made, indicating the absence of a device on the desired channel. If the channel is available, the response causes the information on the Megabus address and data lines to be stored. The state of certain control lines are also stored, and the Megabus logic goes busy to inhibit further transfers to the MTC until firmware has dispensed with the stored information.

When the MTC requires a Megabus transfer, firmware loads the address and data registers and sets the proper control lines prior to Megabus cycle initiation. The response of the slave unit is stored by the Megabus logic, and the Megabus logic remains busy until firmware detects completion of the cycle.

Control signals for synchronization of the MTC Megabus requests (and Megabus requests by other units) are also located in the Megabus logic. Finally, the Megabus logic determines the MTC priority with reference to other units when there are simultaneous requests for Megabus activity.

2.3.5 Adapter/Channel Control Logic

A channel consists of the MTC, device adapter, and device. Complete channel control results from a combination of hardware and firmware.

Each channel provides two types of service requests: data and nondata. These service requests are supplied to a channel request encoder where the presence of a data service request signifies that a data byte read from the device is available, or that the need for a data byte to be transferred to the device exists. The nondata service request indicates a change of device state or the detection of a device condition which requires attention (e.g., head of form on the printer).

The request encoder indicates when a channel request is active. It also sets up the priorities so that any Megabus request has priority over a channel data service request. A similar request from channel 0 would have higher priority than a request from any of the higher channel numbers.

Firmware can test the output of the priority encoder to determine if a request has occurred. It also can test the encoder to ascertain if the request with the highest priority is a Megabus request or a channel request.

The priority encoder has two outputs which are a binary function of the highest priority channel. These lines are used to enable a single adapter at a time corresponding to the appropriate channel. One exception occurs when the Master Clear signal is active: all the adapters are enabled simultaneously to receive the Master Clear signal.

2.3.6 Test Multiplexer

Subsystem conditions, including status, errors, and register bits, can be examined by firmware via the test multiplexer. Firmware operations which reqire the bypassing of the microoperation specify a signal and its condition for which a skip should occur. The test multiplexer provides the facilities for comparing the state of the specified signal with the state specified by the firmware. As a result, the instruction register is cleared for one cycle if the two conditions are equal.

2.3.7 Clock Logic

The MTC divides the output of an 8-MHz oscillator to derive a 4-MHz clock cycle. The MTC is supplied with both the assertion and negation of the clock, each having a 250-nanosecond duration. Both of these signals are utilized by the MTC to implement logic in the first and the second 250-nanosecond periods of a single clock cycle. A third type of clock pulse is utilized by the MTC as a strobe pulse for the SPM, the adapters, and various other MTC registers. This strobe occurs during the latter part of the clock cycle and is only 35 to 55 nanoseconds in duration.

2.4 OUTPUT DATA OPERATION

The following subsections describe the MTC hardware and firmware implemented when an output data operation is performed on a selected channel. The discussion centers around the data transfer path, although the firmware control, timing, and data verification checks required to execute the output data operation are also discussed.

Two functions are required to complete an output data operation: software initiation and firmware control through hardware implementation.

2.4.1 Software Initiation

Using the output function codes listed in Table 2-1 and the general bus formats illustated in Figure 2-2, the software is capable of initiating the output data operation. This is accomplished by sending the following control information from the central processor to the MTC for storage in the SPM: the configuration words (FC = 11 hex), the memory address (FC = 09 hex), and the range (FC = 0D hex).

At this point, software involvement is terminated if the addressed device is capable of only one functional operation (i.e., card reader). If the addressed device is capable of performing multiple functional operations (i.e., magnetic tape), software is required to make an additional transfer, the task word (FC = 07 hex), to the MTC for storage in the SPM.

After the last software transfer is completed, the remaining control of the output data operation is maintained by the MTC firmware.

2.4.2 Firmware Control Through Hardware Implementation (See Figure 2-1)

Firmware is made aware that an output data operation is to be performed when a bus request is detected which is outputting control information. The control information on the data lines, as a result of the bus request from the central processor, is stored in the SPM in the locations specified by the function code.

With software initiation of the output data operation, either three or four bus transfers to the MTC are executed (refer to subsection 2.4.1). The control information transferred across the bus is the configuration words, the memory address, the range, and, if necessary, the task word. As each transfer occurs, the firmware stores the control information into the SPM of the MTC. Control of the remaining portion of the output data operation (the data transfer) is maintained by the firmware.

The firmware performs all the output data operations (write or print), status verification, channel validity, range determination, and device capabilities. The task is initiated by firmware loading the adapter's control registers with the control information previously stored in the SPM as a result of the software bus request. The adapter then generates requests for data, and the MTC firmware accesses memory, receives data, and transfers the data to the adapter using the hardware path shown in Figure 2-1. All data integrity checks performed on the information sent to the adapters is accomplished by the adapter hardware or the device-specific firmware routines.

After detection of the data transfer termination (end of range), the firmware can report the operation's status to the software by generating a bus transfer. Using the transferred status, software then establishes if the operation was successful. Retries of unsuccessful operations are performed according to software parameters.

2.5 INPUT DATA OPERATION

Software and firmware involvement in the input data operation is identical to the output data operation (refer to subsection 2.4). The variations of the input data operation are the hardware data paths between the Megabus and the adapter. The output data operation transfers data from the Megabus logic through the ALU and the SPM to the adapter, whereas the input data operation transfers information from the adapter through the ALU, the SPM, and the Megabus logic to the Megabus.

III THEORY OF OPERATION - INTERMEDIATE

The MTC is divided into seven fundamental logic areas. These primary areas and the data, instructions, and control flows are illustrated in Figure 3-1. The seven basic areas which comprise the MTC are:

- Microprogram Control Store
- Arithmetic Logic Unit and Accumulator
- Scratch Pad Memory and Addressing
- Test Multiplexer
- Megabus Logic
- Adapter Logic
- Clock Logic.

Each area is an arrangement of several functional components and is described in detail under its appropriate subsection heading.

3.1 INTERFACE

The MTC contains two interface networks, one which allows connection to the Megabus and one which allows attachment of a variable configuration of device adapters.

3.1.1 Megabus/MTC Interface

The MTC/Level 6 Megabus interface is the control and transfer link between the MTC and any other unit within the system. It provides a path for address, data, and control information. This inteface also supplies the paths for determining the priority of

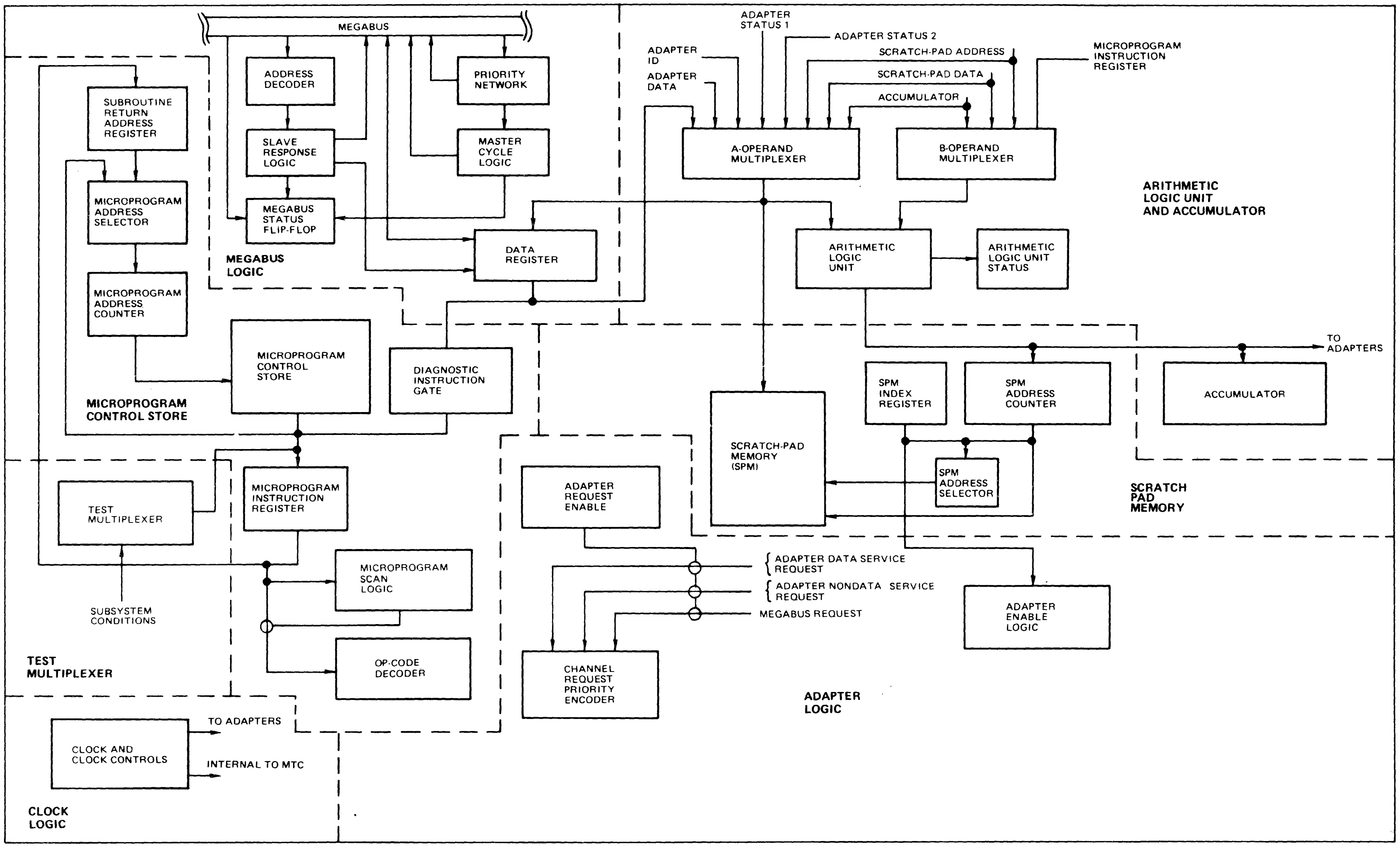


Figure 3-1 MTC Hardware Block Diagram

a request from any attached controller. Figure 3-2 identifies all the interface lines and their direction, usage, and mnemonics. Table 3-1 describes each signal line used by the Megabus/MTC interface.

3.1.2 MTC/Adapter Interface

Figure 3-3 identifies the MTC/adapter interface lines and their direction, mnemonics, and usage. It depicts the lines as seen from the MTC (not from the adapter, where many of the lines have other mnemonics). Table 3-2 describes each signal line used by the MTC/adapter interface. For a cross-reference of signal names, refer to Table 3-3.

This interface links all three adapters and enables the firmware to select the proper adapter and perform the designated operation. It provides the paths necessary to supply the adapters with data, control, and timing pulses; it also supplies the MTC with requests and data from the adapter.

3.2 MICROPROGRAM CONTROL STORE FUNCTIONAL COMPONENTS (Figure 3-4)

3.2.1 Subroutine Return Address Register (SRAR)

The SRAR is used by firmware to store a microprogram control store (UPCS) address which will be branched to at some later time by a firmware routine or subroutine. To load a location with an address, the firmware issues a Load Return Address command, which generates the write enable function LDSRAR-. This causes the address defined by bits 4 through 15 of the microprogram instruction register (UPIR) to be stored in the SRAR at the location specified by the SPM address selection bits (SPMAS0+, SPMAS1+).

The SRAR is a 4-word by 12-bit register file memory. Each word represents the return address of a particular adapter channel. The SRAR word that is being read or written is determined by the output of the Scratch Pad Memory Index Register (SPMIR0- and SPMIR1-). The Scratch Pad Memory Index Register (SPMIR) contains the number of one of the four device channels in binary code (see subsection 3.4.4).

When firmware performs a Return Branch command, the contents of the location defined by SPMAS0+ and SPMAS1+ are gated to the SRAR output. The output is then transferred by way of the microprogram address selector (UPAS) to the microprogram address counter (UPAC). Utilizing the Return Branch command, the UPAC is preset to an address previously stored in the SRAR.

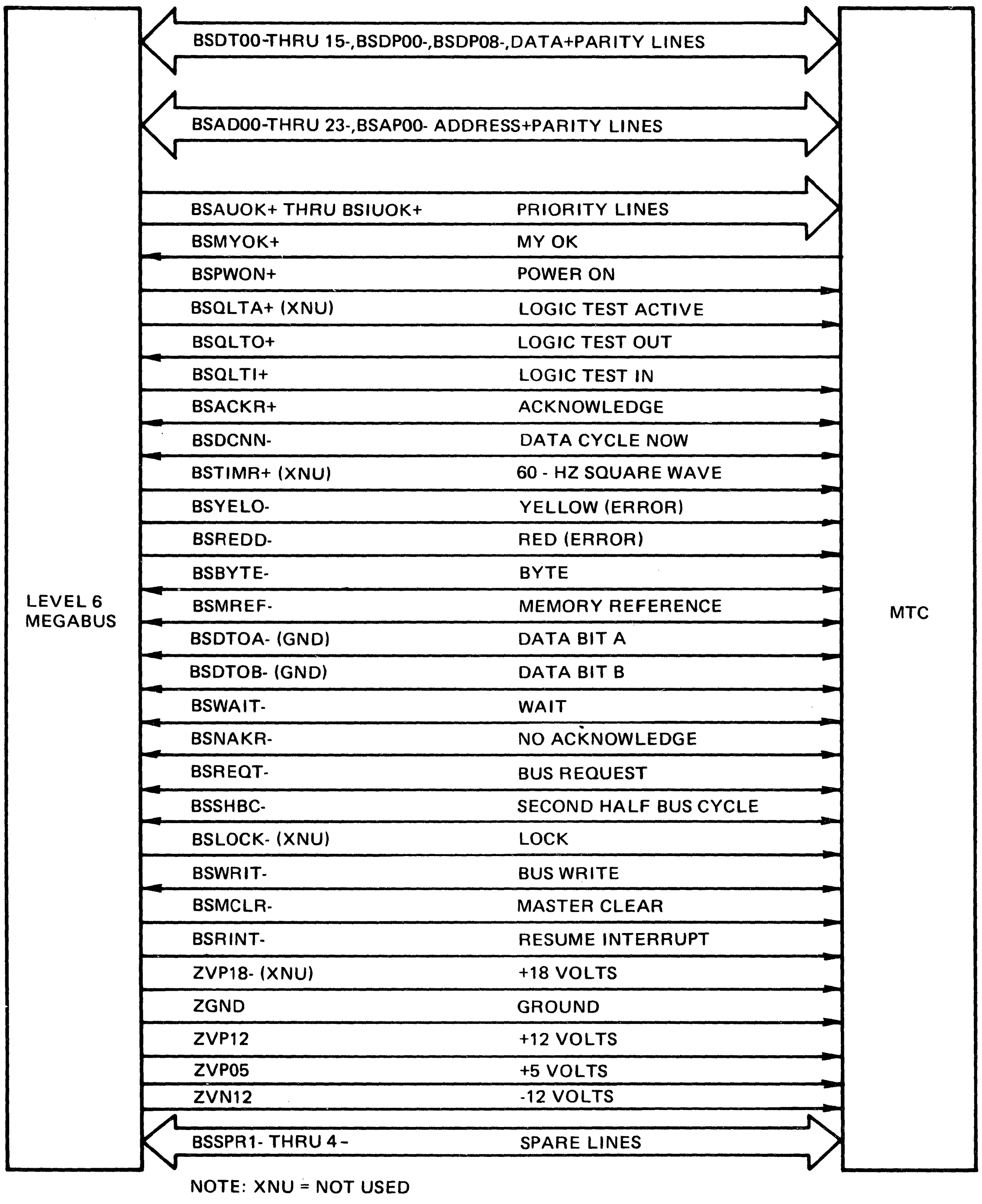


Figure 3-2 Megabus/MTC Interface

Table 3-1 Megabus/MTC Interface Signal Lines
(Sheet 1 of 2)

TERM/MNEMONIC	DESCRIPTION
Data Bits 0 to 7 (BSDT00- to 07-)	These 8 data bit lines represent the most significant byte of data.
Data Bits 8 to 15 (BSDT08- to 15-)	These 8 data bit lines represent the least significant byte of data.
Data Parity - Left Byte (BSDP00-)	This signal contains odd parity for data bits 0 through 7.
Data Parity - Right Byte (BSDO08-)	This signal contains odd parity for data bits 8 through 15.
Address Bus Bits 0 to 23 (BSAD00- to 23-)	These 24 address bit lines contain an address to be used by memory or by a controller or central processor.
Address Parity (BSAP00-)	This signal contains odd parity for the most significant byte of the address bus, bits 0 through 7.
Priority Lines (BSAUOK+ to BSIUOK+)	These lines are used to establish priority of the units attached to the Megabus.
My Ok (BSMYOK+)	This signal indicates the unit that is presently using the Megabus.
Power On (BSPWON+)	This signal is true when all power supplies in the system are operating correctly.
Logic Test In (BSQLTI+)	This signal initiates the Internal Logic Test in a unit attached to the Megabus.
Logic Test Out (BSQLTO+)	This signal indicates the unit has successfully completed running its Internal Logic Test and is used as the Logic Test In signal for the next unit attached to the Megabus.
Acknowledge (BSACKR+)	This signal indicates that the information on the Megabus has been accepted.
Data Cycle Now (BSDCNN-)	This signal indicates that the information on the bus is valid.

Table 3-1 Megabus/MTC Interface Signal Lines
(Sheet 2 of 2)

TERM/MNEMONIC	DESCRIPTION
Yellow (BSYELO-)	This signal indicates that the accompanying transferred information is correct but that a memory correction operation was performed.
Red (BSREDD-)	This signal indicates that the accompanying transferred information is in error.
Byte (BSBYTE-)	This signal indicates that the current transfer is a byte transfer rather than a word transfer.
Memory Reference (BSMREF-)	This signal indicates that the address leads contain a memory address.
Wait (BSWAIT-)	This signal indicates that the transfer will be accepted when the Megabus data register is available.
No Acknowledge (BSNAKR-)	This signal indicates that the information on the Megabus has been refused.
Bus Request (BSREQT-)	This signal indicates that one or more units on the Megabus have requested a bus cycle.
Second Half Bus Cycle (BSSHBC-)	This signal identifies the second bus cycle in response to a memory read request.
Bus Write (BSWRIT-)	This signal indicates that information on the Megabus is ready to be transferred.
Master Clear (BSMCLR-)	This signal initializes the units attached to the Megabus.
Resume Interrupt (BSRINT-)	This signal is a 200-nanosecond pulse which is issued by the central processor when it is capable of receiving interrupts again

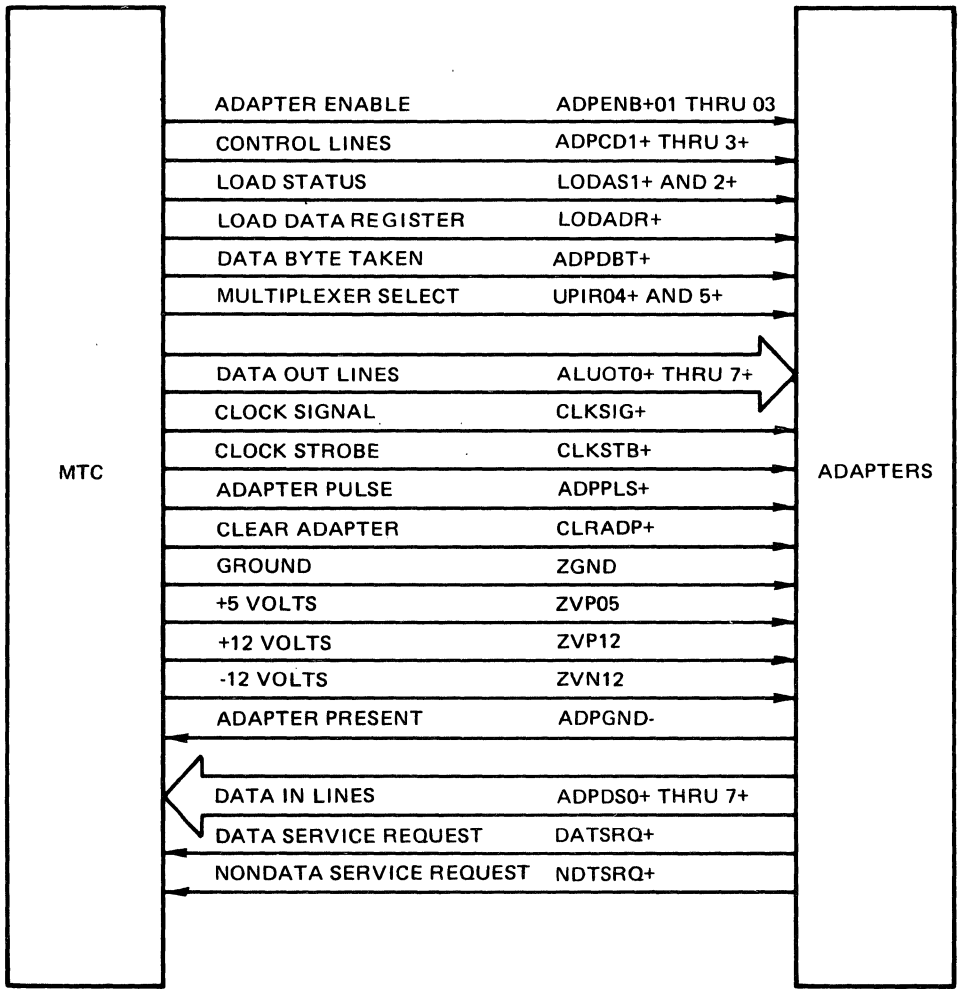

MTC
ADAPTERS
ADAPTER ENABLE ADPENB+01 THRU 03
CONTROL LINES ADPCD1+ THRU 3+
LOAD STATUS LODAS1+ AND 2+
LOAD DATA REGISTER LODADR+
DATA BYTE TAKEN ADPDBT+
MULTIPLEXER SELECT UPIR04+ AND 5+
DATA OUT LINES ALUOT0+ THRU 7+
CLOCK SIGNAL CLKSIG+
CLOCK STROBE CLKSTB+
ADAPTER PULSE ADPPLS+
CLEAR ADAPTER CLRADP+
GROUND ZGND
+5 VOLTS ZVP05
+12 VOLTS ZVP12
-12 VOLTS ZVN12
ADAPTER PRESENT ADPGND-
DATA IN LINES ADPDS0+ THRU 7+
DATA SERVICE REQUEST DATSRQ+
NONDATA SERVICE REQUEST NDTSRQ+

Figure 3-3 MTC/Adapter Interface

Table 3-2 MTC/Adapter Interface Signal Lines
(Sheet 1 of 2)

TERM/MNEMONIC	DESCRIPTION
Adapter Enable (ADPENB+01 to +03)	The Magnetic Tape Adapter uses these 3 lines as device selection bits. The low-speed device adapters use these lines as enables for the device adapters.
Control Lines (ADPCD1+ to 3+)	These 3 signals are used for adapter-specific operations.
Load Status (LODAS1+ to 2+)	These 2 signals to the device adapter are used to load data in the control registers of the device adapter.
Load Data (LODADR+)	This signal is used to load data in the data register of the device adapter.
Data Byte Taken (ADPDBT+)	This signal notifies the device adapter that the contents of its data register have been stored.
Multiplexer Select (UPIR04+ to 05+)	These 2 signals are the selection bits for the input multiplexer of the device adapter.
Data Out Lines (ALUOT0+ to 7+)	These 8 data lines from the MTC to the device adapter represent the information to be used by the adapter.
Clock Signal (CLKSIG+)	A clock function from the MTC to the device adapter.
Clock Strobe (CLKSTB+)	A clock function from the MTC to the device adapter.
Clear Adapter (CLRADP+)	This signal clears the major logic areas of the adapter.
Adapter Present (ADPGND-)	This signal notifies the MTC that an adapter is present by indicating a ground connection.
Data In Lines (ADPDS0+ to 7+)	These 8 data lines from the device adapter to the MTC represent the information to be used by the MTC.
Data Service Request (DATSRQ+)	This signal notifies the MTC that the device adapter has a data byte ready or that a data byte is required.

Table 3-2 MTC/Adapter Interface Signal Lines
(Sheet 2 of 2)

TERM/MNEMONIC	DESCRIPTION
Nondata Service Request (NDTSRQ+)	This signal notifies the MTC that the device adapter has a nondata service request to be processed (e.g., a change in device state).
Adapter Pulse (ADPPLS+)	This signal is used by the printer device adapter to generate a strobe for the printer

Table 3-3 MTC/Adapter Interface Line and Mnemonic
Cross-Reference List

MTC		ADAPTER		
FUNCTION	MNEMONIC	TAPE MNEMONIC	READER MNEMONIC	PRINTER MNEMONIC
Adapter Enable	ADPENB+01→03	ADPENB+01→03	ADPENX	ADPENX
Control Lines	ADPCD1+	ADPCD1+	XNU	XNU
	ADPCD2+	ADPCD2+	XNU	XNU
Control Lines	ADPCD3+	ADPCD3+	XNU	XNU
Load Status 1	LODAS1+	LODAS1+	ADPCN1+	ADPCN1+
Load Status 2	LODAS2+	LODAS2+	ADPCN2+	ADPCN2+
Load Adapter Data	LODADR+	LODADR+	XNU	ADPDAT+
Data Byte Taken	ADPDBT+	ADPDBT+	ADPDBT+	XNU
Data Out	ALUOT0+→7+	ALUOT0+→7+	ALUOT0+→7+	ALUOT0+→7+
Multiplexer Select	UPIR04+	UPIR04+	ADPIC0+	ADPIC0+
Multiplexer Select	UPIR05+	UPIR05+	ADPIC1+	ADPIC1+
Data Service Request	DATSRQ+	DATSRQ+	DATSRQ+	DATSRQ+
Nondata Service Request	NDTSRQ	NDTSRQ+	NDTSRQ	NDTSRQ+
Data In	ADPDS0+→7+	ADPDS0+→7+	INFOI0+→7+	INFOI0+→7+
Clock Signal	CLKSIG+	CLKSIG+	XNU	XNU
Clock Strobe	CLKSTB+	CLKSTB+	CLKSTB+	CLKSTB+
Adapter Pulse	ADPPLS+	ADPPLS+	XNU	ADPPLS+
Adapter Present	ADPGND-	ADPGND-	ZGND	ZGND
Clear Adapter	CLRADP+	CLRADP+	ADPCLR+	ADPCLR+
Ground	ZGND	ZGND	ZGND	ZGND
+5 Volts	ZVP05	ZVP05	ZVP05	ZVP05
-12 Volts	ZVN12	ZVN12	ZVN12	ZNV12
+12 Volts	ZVP12	ZVP12	ZVP12	ZVP12

Note: XNU = Not used.

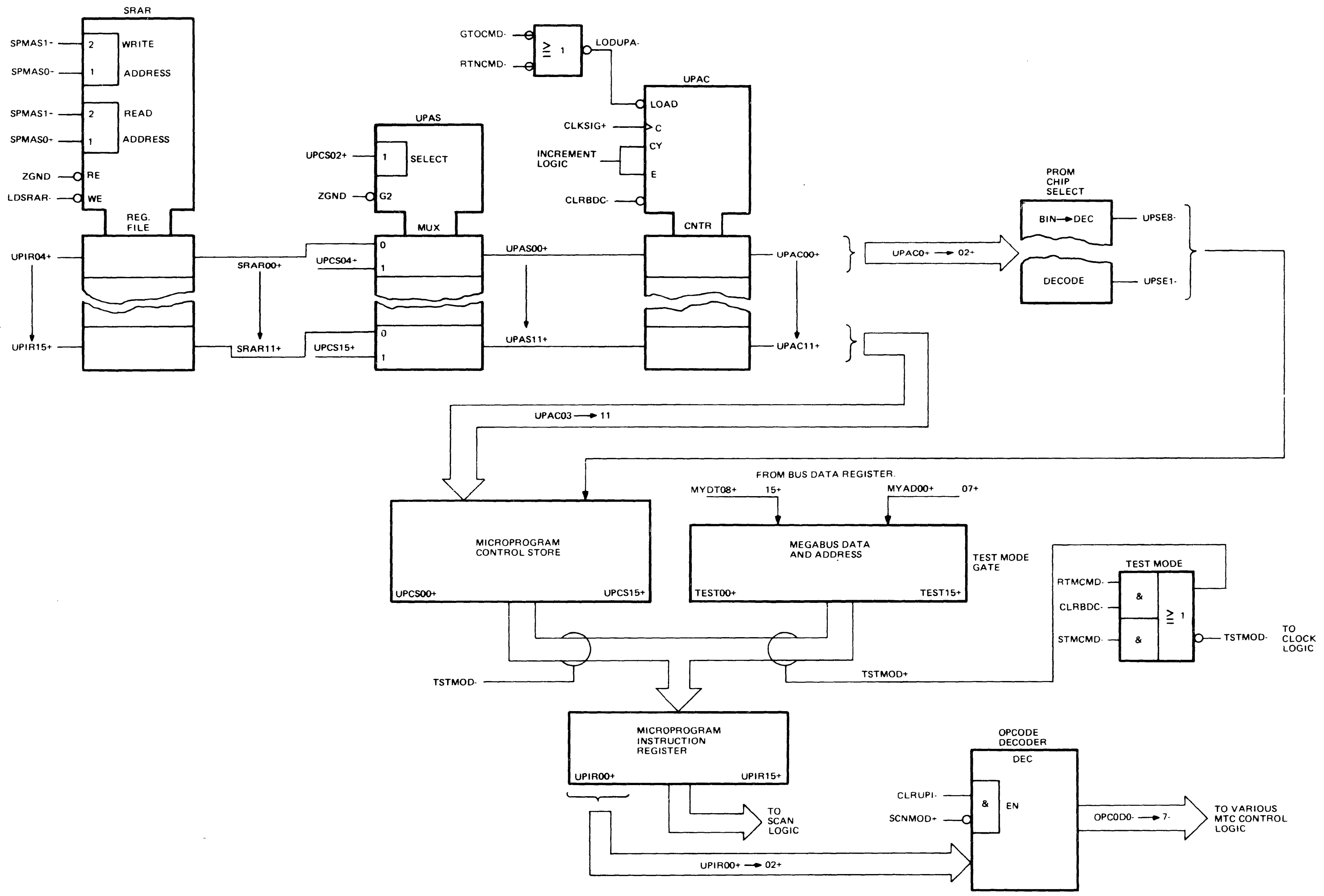


Figure 3-4 Microprogram Control Store Functionality

3.2.2 Microprogram Address Selector (UPAS)

The UPAS selects one of its two inputs to be used as a preset for the UPAC. When performing firmware operations with the output of the Microprogram Control Store bit 2 (UPCS02+) high, indicating a Go To command, the 12-bit output of the UPAS reflects bits 4 through 15 of the UPCS. When UPCS02+ is low, the 12 bits of the SRAR are selected for the UPAS outputs to implement a Return Branch command (RTN).

3.2.3 Microprogram Address Counter (UPAC)

The UPAC is a 12-bit counter which is incremented once at the start of every clock cycle, except in the instance of a clear or load operation. The UPAC is cleared by CLRBCD-, which is active only during an MTC Initialize (Master Clear); it is loaded when a Load UPAC operation (LODUPA-) is active. The LODUPA- function causes the UPAC to reflect the address on the microprogram address selector (UPAS) output. The Load UPAC operation is performed when a Go To command or a Return Branch command is decoded in the address control logic.

The low-order nine bits (3 through 11) of the UPAC are gated directly to the address lines of the UPCS. The high-order three bits (0 through 2) are fed to the Microprogram Control Enable (UPCSE1- through UPCSE8-), which is a 3-bit to 1-of-8-line decoder. The eight UPCSE functions are used to enable one of the eight rows within the microprogram control store.

3.2.4 Microprogram Control Store (UPCS)

For a description of the UPCS, shown in Figure 3-4, refer to subsection 2.3.1 of this manual.

3.2.5 Diagnostic Instruction (Test) Gate

When the diagnostic mode is specified in the control word issued to the MTC by software, the MTC firmware executes a Set Test Mode (STMCMD-) command. This command sets the TSTMOD+ flip-flop, which in turn puts the MTC clock in the step mode and disables the microprogram control store outputs. At this time the diagnostic instruction (test) gate is enabled, allowing the transfer of information from the bus data register to the ORing network of the microprogram control store. Any subsequent transfer to the MTC while the TSTMOD+ flip-flop is set generates one clock cycle. During this cycle the data on the Megabus is transferred by the test mode gate and the ORing network to the microprogram instruction register. It is then executed as if it were a resident firmware command.

The TSTMOD flip-flop remains set until software issues a Reset Test Mode (RTMCMD-) instruction or until a Master Clear signal occurs.

3.2.6 Microprogram Instructon Register (UPIR)

The microprogram instruction register (UPIR) is a 16-bit wide register used to store the output of the microprogram control store (UPCS) or the test gate for one clock cycle during a micro-instruction execution. The UPIR is loaded at the leading edge of each cycle by CLKSIG+ unless the signal Clear Microprogram Instruction Register (CLRUPI-) is active, which causes a reset to Zero. CLRUPI is active during the Master Clear operation and during a skipped cycle due to a successful test instruction.

3.2.7 Op-Code Decoder (OPCOD)

The high-order three bits (UPIR00+ through UPIR02+) of the UPIR are fed to the op-code decoder, which performs a 3-bit to 1-of-8-lines decode. These lines indicate what type of firmware command is being performed. The op-code decoder is enabled unless the MTC is in the process of performing a UPCS scan or unless the UPIR is being cleared by the CLRUPI- function. The eight output lines of the op-code decoder, in conjunction with various other UPIR bits, implement and control the MTC and device adapter hardware.

To increase the speed of operation during the firmware routines, the op-code of the branch (Go To) command is decoded directly from the output of the UPCS rather than from the UPIR. For the same reason, the address for the Go To and Return command is taken from the output of the UPCS. For all other command types, the op-code is decoded from the output of the UPIR.

3.2.8 Scan Logic (Figure 3-5)

The scan logic in the MTC ensures the integrity of the UPCS. This logic performs a longitudinal check on each bit position of the UPCS outputs and causes the MTC clock to halt if an error is detected. If no errors are detected, the MTC goes on to execute the firmware portion of the Quality Logic Test.

The Scan Mode flip-flop (SCNMOD+) sets with Master Clear (MSTCLR-), disabling the op-code decoder (see Figure 3-4) and inhibiting branch commands. With the commands inhibited, the MTC loads the UPIR with the output of the UPCS and increments the UPAC to the next location at each sequential clock cycle. This process continues from location 000 to location FFF, at which time a carry-out of the UPAC (UPAC2C-) is detected. The UPAC then returns to Zero, beginning another scan.

The scan bit selectors (SCNBSH+ and SCNBSL+) select one bit of the UPIR, using the output of the scan bit address counter as an address. The selected UPIR bit provides the input to the Scan Bit Sum flip-flop (SCNBSM+). The SCNBSM+ half adds the selected bit at the start of each clock cycle with its previous contents. At the end of one scan through the UPCS, the Scan Bit Sum flip-flop has added the specified bit of each location with an

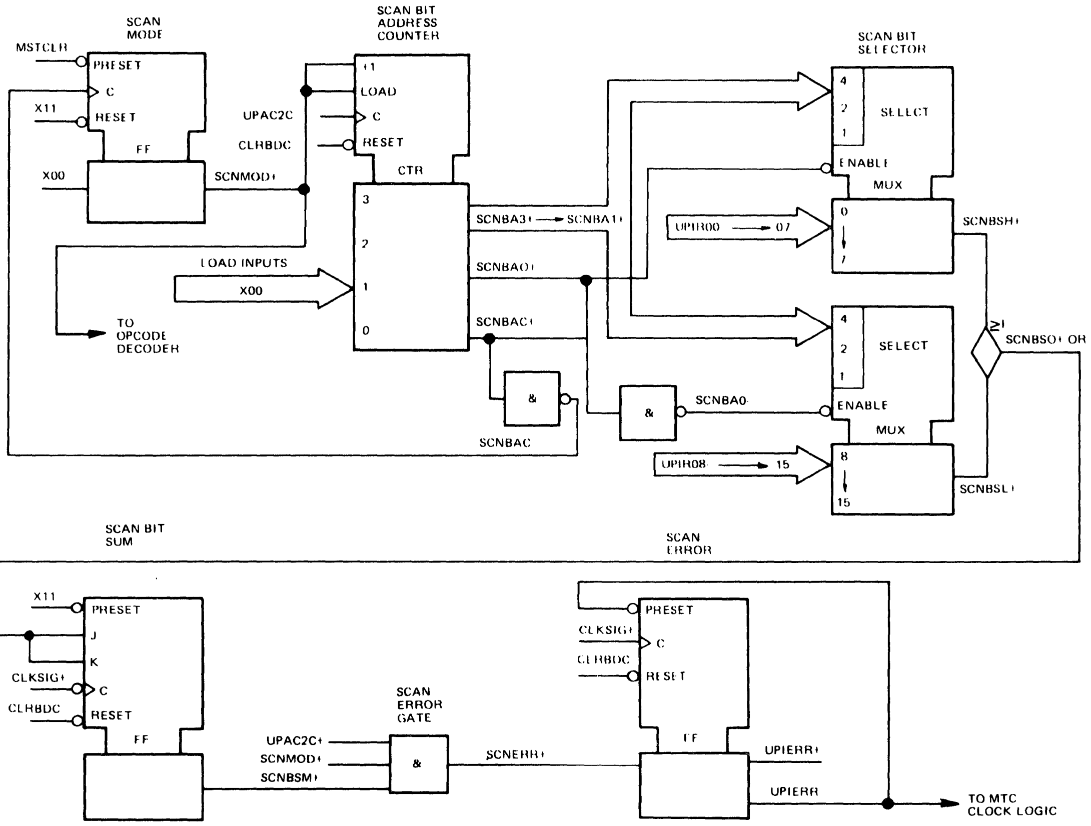


Figure 3-5 Scan Logic

expected result of zero (no error). If the sum is not zero at the termination of each scan, the Scan Error gate output (SCNERR+) is high and sets the Scan Error flip-flop (UPIERR+). UPIERR- causes the MTC clock to halt, and the contents of the scan bit address counter indicates which bit column of the UPCS is in error.

If an error is not detected at the end of a scan, the scan bit address counter increments due to the overflow of the UPAC (UPAC2C-), and the UPAC wraps around to location 0. The scan operation is repeated with the next bit of the UPIR as an input to the Scan Bit Sum flip-flop. The scan operation continues until an error is detected or until all bits of the UPIR have been checked, at which time the scan bit address counter (SCNBAC+) overflows and resets the Scan Mode flip-flop. When the Scan Mode flip-flop is reset, the op-code decoder and branch commands are re-enabled, and normal execution of firmware begins.

3.3 ARITHMETIC LOGIC UNIT AND ACCUMULATOR FUNCTIONAL COMPONENTS

3.3.1 A-Operand Multiplexer (ALUAX) (Figure 3-6)

The A-operand multiplexer (AOP MUX) selects one of five types of data fields according to a 3-bit multiplexer address defined by bits 3 through 5 of the UPIR. Table A of Figure 3-6 shows the various bit configurations and the selected register.

For addresses equal to 4 through 7, UPIR bit 3 set, the output of the adapter's data selector is used as an input to the AOP MUX. Bits 4 and 5 of the UPIR then select one of the four adapter registers visible to the MTC: data, ID, status 1, or status 2.

Only the adapter which is active has its data enabled at the selector output. The outputs of the adapter's data selectors are ORed on the MTC, and the ORed outputs are fed only to the AOP MUX.

The destinations of the 8-bit output of the AOP MUX are the ALU, the test multiplexer, the scratch pad memory, and the bus data register.

3.3.2 B-Operand Multiplexer (ALUBX) (Figure 3-6)

The B-operand multiplexer (BOP MUX) selects one of four data fields as a B-input to the ALU. The BOP is defined during ALU commands by UPIR bits 6 and 7. When performing Constant commands, the BOP MUX selects the output of the UPIR bits 6 through 10, 12, 14, and 15 for the input to the ALU. These particular UPIR bits represent the data constant to be operated with during the Constant command.

The address for the data field to be loaded into the BOP MUX is a two-function decode (ALUBS1+, ALUBS2+) of the operation code (UPIR bits 0+ through 2+) and UPIR bits 6- and 7-. Table B of Figure 3-6 shows the four possible configurations of ALUBS1+ and ALUBS2+ and the data input selected.

3.3.3 Arithmetic Logic Unit (ALU) (Figure 3-6)

The ALU performs an 8-bit arithmetic or logic operation on the data supplied by the AOP and BOP multiplexers. The type of operation performed is determined by the four mode signals (ALUMD3+, UPIR11+, ALUMD1+, UPIR13+), the carry enable function (ALUMCE-), and the carry input (ALUCIN-) to the ALU. Table 3-4 shows the relationship between these functions and the operations performed by the ALU.

These mode signals, as well as Carry Enable and Carry In, are explicitly defined by bits of the UPIR during ALU commands (op-code of 3). The firmware can also set Carry Enable.

TABLE C ALU CARRY FLIP–FLOP INDICATIONS

ALU CARRY–IN (ALUCIN–)	ALU CARRY–OUT (ALUCOT+)	AOP AND BOP RELATIONSHIP
1	0	$A \leqslant B$
1	1	$A > B$
0	1	$A \geqslant B$
0	0	$A < B$

TABLE A AOP MULTIPLEXER INPUT SELECTION

UPIR BIT CONFIGURATION				
ADDRESS BITS				
03	04	05	SELECTED REGISTER	MNEMONIC
0	0	0	ACCUMULATOR	ALUAC0+ – 7+
0	0	1	SCRATCH PAD DATA	SPMOT0+ – 7+
0	1	0	SCRATCH PAD ADDRESS	SPMAC0+ – 7+
0	1	1	BUS DATA	MYAD16+ – 23+*
1	0	0	ADAPTER DATA	ADPDS0+ – 7+
1	0	1	ADAPTER ID	ADPDS0+ – 7+
1	1	0	ADAPTER STATUS 1	ADPDS0+ – 7+
1	1	1	ADAPTER STATUS 2	ADPDS0+ – 7+

*THE DATA SEEN AT THIS POINT DEPENDS ON THE NUMBER OF SHIFTS PERFORMED ON THE BUS DATA REGISTER.

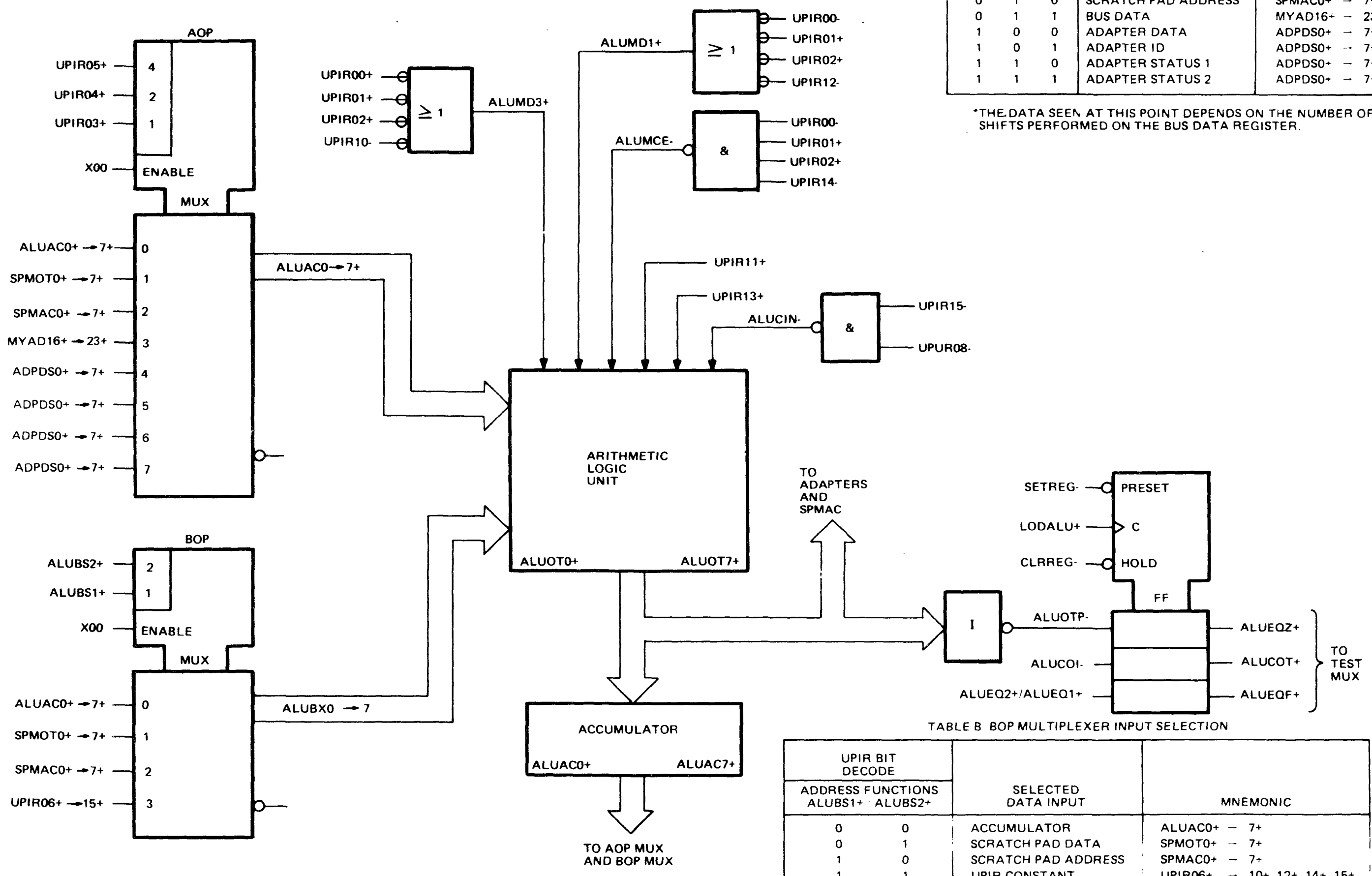


TABLE B BOP MULTIPLEXER INPUT SELECTION

UPIR BIT DECODE			
ADDRESS FUNCTIONS ALUBS1+	ALUBS2+	SELECTED DATA INPUT	MNEMONIC
0	0	ACCUMULATOR	ALUAC0+ – 7+
0	1	SCRATCH PAD DATA	SPMOT0+ – 7+
1	0	SCRATCH PAD ADDRESS	SPMAC0+ – 7+
1	1	UPIR CONSTANT	UPIR06+ – 10+, 12+, 14+, 15+

Figure 3-6 ALU and ACU Functionality

Table 3-4 ALU Functionality

HEX CODE	ALUMD3+	UPIR11+	ALUMD1+	UPIR13+	BINARY CODE	ALUMCE- = 1 =H, LOGIC OPERATIONS	POSITIVE LOGIC	
							ALUMCE- = 0 = L, ARITHMETIC OPERATIONS	
							ALUCIN- = 1	ALUCIN- = 0
0	L	L	L	L	(0000)	$F = \overline{A}$	F = A	F = A plus 1
1	L	L	L	H	(0001)	$F = \overline{A + B}$	F = A + B	F = (A + B) plus 1
2	L	L	H	L	(0010)	$F = \overline{A}B$	$F = A + \overline{B}$	$F = (A + \overline{B})$ plus 1
3	L	L	H	H	(0011)	F = 0	F = minus 1 (2s compl)	F = ZERO
4	L	H	L	L	(0100)	$F = \overline{AB}$	F = A plus $A\overline{B}$	F = A plus $A\overline{B}$ plus 1
5	L	H	L	H	(0101)	$F = \overline{B}$	F = (A + B) plus $A\overline{B}$	F = (A + B) plus $A\overline{B}$ plus 1
6 ↑	L	H	H	L	(0110)	$F = A \oplus B$	F = A minus B minus 1	F = A minus B
7	L	H	H	H	(0111)	$F = A\overline{B}$	F = $A\overline{B}$ minus 1	$F = A\overline{B}$
8	H	L	L	L	(1000)	$F = \overline{A} + B$	F = A plus AB	F = A plus AB plus 1
9 ↑↑	H	L	L	H	(1001)	$F = \overline{A \oplus B}$	F = A plus B	F = A plus B plus 1
10	H	L	H	L	(1010)	F = B	$F = (A + \overline{B})$ plus AB	$F = (A + \overline{B})$ plus AB plus 1
11	H	L	H	H	(1011)	F = AB	F = AB minus 1	F = AB
12	H	H	L	L	(1100)	F = 1	F = A plus A*	F = A plus A plus 1
13	H	H	L	H	(1101)	$F = A + \overline{B}$	F = (A + B) plus A	F = (A + B) plus A plus 1
14	H	H	H	L	(1110)	F = A + B	$F = (A + \overline{B})$ plus A	$F = (A + \overline{B})$ plus A plus 1
15	H	H	H	H	(1111)	F = A	F = A minus 1	F = A

↑ Subtract Mode
↑↑ Add Mode
* Each bit is shifted to the next more significant position

NOTE

When $\overline{CE}$ is a logic ONE, $\overline{C_{in}}$ is irrelevant.

When an ALU command is not being performed, the mode signals (ALUMD3+, ALUMD1+), and the Carry Enable (ALUMCE-) signal default to a One state.

The two remaining ALU mode lines (UPIR11+, UPIR13+) do not have a default state but rather vary according to the output of the UPIR bits 11 and 13. These two bits select one of four operations which can be performed when executing a constant command.

The ALU data output is fed to the accumulator, the scratch pad memory address counter, and to a set of amplifiers which distribute the ALU data to the device adapters. In addition, the ALU outputs are inputs to a set of inverters whose wire-ANDed outputs provide the data for the ALU Equal Zero (ALUEQZ+) flip-flop. This status flip-flop, along with the ALU Equal FF (ALUEQF+) flip-flop and ALU Carry-out (ALUCOT+) flip-flop, is set during the ALU operation by the signal LODALU+. In this manner, the ALU status filp-flops are set or reset during an ALU command and remain valid for firmware interrogation until the subsequent ALU operation.

The ALUEQZ+ and ALUEQF+ status flip-flops indicate an all-Zeros ALU output and an all-Ones ALU output, respectively. The ALUCOT+ flip-flop indicates a carry-out of the ALU, which serves to indicate the relative magnitudes of the AOP and BOP fields during an ALU subtract operation. Table C of Figure 3-6 shows the relationship between the ALU carry-in, the ALU carry-out, and the size of the AOP and BOP fields.

For diagnostic purposes, each of the ALU status flip-flops can be set independently of the ALU outputs by the function SETREG-. They can also be cleared (reset) by the signal CLRREG-. Both SETREG- and CLRREG- can be enabled by firmware commands, with CLRREG- also being active during a Master Clear. During a Master Clear, CLRREG- causes the ALU status flip-flops to be zeroed. After the first ALU command, these flip-flops reflect the proper state indicated by the ALU outputs.

3.3.4 Accumulator (ACU) (Figure 3-6)

The ACU, which can be cleared by the function CLRREG-, is an 8-bit register that provides temporary storage of the ALU output. The ACU is loaded at CLKSTB+ time during the following operations.

- When performing a Load Constant command with the scratch pad memory (SPM) or the ACU defined as the A-operand.

- When performing an ALU command with the destination of the result specified as either the A-operand (SPM or ACU) or the B-operand (SPM or ACU).

The outputs of the ACU are fed only to the AOP and BOP multiplexers.

3.4 SCRATCH PAD MEMORY FUNCTIONAL COMPONENTS

3.4.1 Scratch Pad Memory Index Control Flip-Flop (SPMICF) (Figure 3-7)

The SPMICF is set or reset as a result of a Set or Reset Index command (SPICMD+) according to the state of the UPIR bit 9 (UPIR09+). The SPMICF is also reset as a result of the CLRREG- function being active.

The output of the SPMICF determines the source of the SPM address. When the SPMICF is reset, the memory is accessed by all eight bits (absolute address) of the SPM address counter (SPMAC). However, when the SPMICF is set, the SPM is addressed by the six low-order bits of the SPMAC, and the two high-order bits (relative address) are taken from the contents of the SPM index register (SPMIR).

3.4.2 Scratch Pad Memory Index Register (SPMIR) (Figure 3-7)

The MTC can be configured with up to four devices (channels). Although every device can be busy simultaneously, the MTC multiplexes the channel activity. The SPMIR (SPMIR0+, SPMIR1+) is used to specify which channel is active at any given time. The contents of the SPMIR are utilized as the two high-order bits to

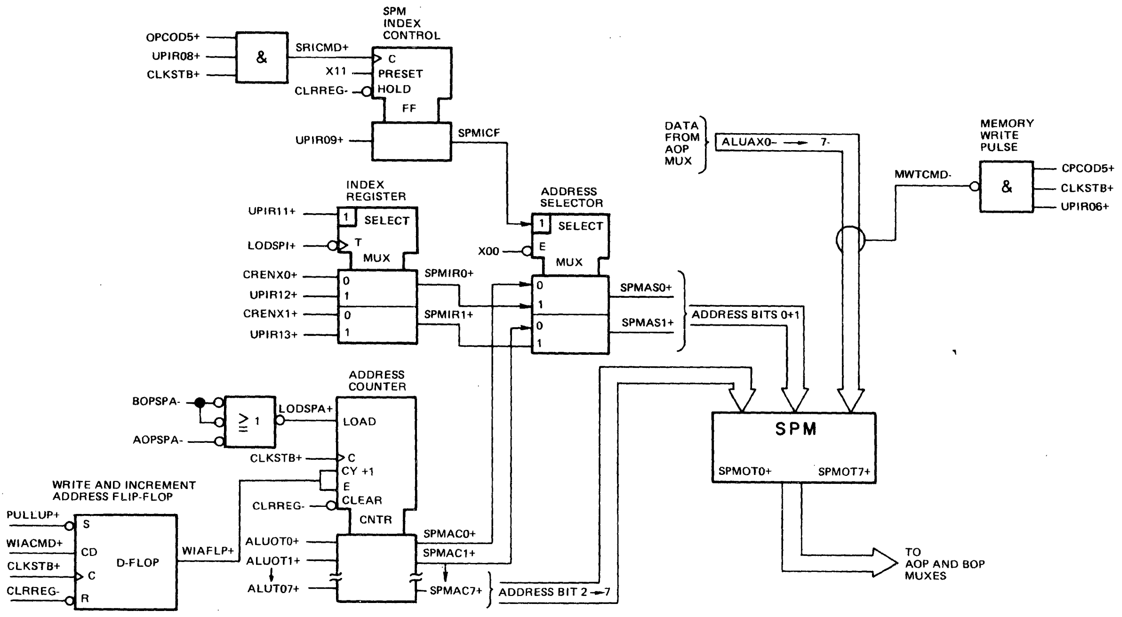


Figure 3-7 Scratch Pad Memory Functionality

address the SPM or the subroutine return address register (see Figure 3-4). The SPMIR also determines which adapter is enabled for proper firmware command execution.

The two-bit SPMIR can be loaded with a specific channel number from the firmware (UPIR12+, UPIR13+). It can also be loaded with the channel number designated by the channel request priority encoder (CRENX0+, CRENX1+). The input to the SPMIR is selected as a result of bit 11 of the instruction register (UPIR11+) and is loaded whenever the function LODSPI+ is active.

3.4.3 Scratch Pad Memory Address Counter (SPMAC) (Figure 3-7)

The SPMAC is an 8-bit counter which is parallel loaded with the output of the ALU or sequentially incremented by a Write and Increment Address (WIACMD) command. The ALU output (ALUOT0+ through ALUOT7+) is loaded into the SPMAC at CLKSTB+ time whenever LODSPA+ is active. The control signal LODSPA+ is generated for those ALU commands (AOPSPA-, BOPSPA-) which specify that the SPMAC be loaded with the ALU output. It is also generated during Constant commands, which utilize the SPMAC as the A-operand.

The SPMAC is incremented whenever the control signal WIACMD is active at CLKSTB+ time, causing the contents of the SPMAC to be increased by one. The Increment (WIAFLP) flip-flop is set during the execution of a WIACMD and reset at the completion of the subsequent command. In this manner the write is performed during the initial cycle of a WIACMD and the increment occurs one cycle later.

The CLRREG- signal, which is generated by a Master Clear and under firmware control, resets the SPMAC to zero and also resets the WIAFLP+ flip-flop.

3.4.4 Scratch Pad Memory Address Selector (SPMAS) (Figure 3-7)

Firmware normally indexes the SPM, thereby dividing the memory into quadrants, one for each available channel. Each quadrant of memory has the same topology (refer to Table 4-11). However, the activity within a specific quadrant may vary according to the device type.

The SPMAS determines the two most significant bits of the SPM address. When the SPMICF is set, the SPMAS selects the contents of the SPMIR as the address. If the SPMICF is reset, the output from the SPMAS reflects the two high-order bits of the SPMAC.

3.4.5 Scratch Pad Memory (SPM) (Figure 3-7)

The SPM is a 256-location by 8-bit read/write memory which stores information that is required by or generated by each channel. Data from the AOP multiplexer is written in the SPM during the Memory Write command at the relative location defined by bits 0 and 1 of the SPMAS and bits 2 through 7 of the SPMAC. Since the input data to the SPM is received from the AOP multi plexer, all firmware-visible registers can be implemented as an input local register. (Refer to subsection 3.5.1.) The data out of the SPM is delivered to both AOP and BOP multiplexers.

The decode of a Memory Write command is ANDed with the clock signal CLKSTB+ producing the write pulse MWTCMD-. The MWTCMD is approximately 40 nanoseconds wide and occurs just prior to the end of an MTC cycle, guaranteeing data setup and hold times. (Refer to subsection 3.8 for a description of the clock signals.)

3.5 MEGABUS LOGIC FUNCTIONAL COMPONENTS

3.5.1 Bus Data Register (BDR) (Figure 3-8)

This logic consists of a 40-bit register for address and data storage plus 5 bits for parity storage, parity generating and checking logic, and a series of flip-flops which control register access by the MTC or the Megabus. The BDR, primarily an input/ output buffer for the Megabus, can be parallel loaded by the Megabus or byte-serial loaded by the MTC.

3.5.1.1 Parallel Load Operation

The BDR is cleared and then parallel loaded with address, data, and parity from the Megabus each time the MTC acknowledges (ACK) a Megabus cycle. The register, parallel loaded during any Megabus cycle during which a load is permissible, occurs concurrently with address decoding and Megabus cycle response generation. The information is loaded into the BDR in this manner because the data on the Megabus becomes invalid at the leading edge of the MTC's response.

The Clear BDR signal (CLRBDR-) clears the BDR prior to the time the parallel loading occurs. This prevents ORing the incoming data and the present register contents. CLRBDR- becomes active whenever the BDR is to be loaded (LODBDR+) and remains active until the data on the Megabus becomes valid.

3.5.1.2 Byte-Serial Load Operation

For loading/unloading from the MTC, the BDR is configured as a 9-bit, 5-byte shift register, each byte containing eight data bits and a parity bit. The MTC loads and unloads the BDR one byte at a time with the AOP-multiplexer-visible registers both receiving and supplying data. The Shift Bus Data Register (SFTBDR+) control signal is active whenever a Shift Data Register command (SDRCMD-) is executed, causing the register to shift one byte position. During the shift operation for unloading the BDR, the contents of the low-order byte (byte 4) is reflected in the AOP multiplexer; each of the other bytes (bytes 0 through 3) is

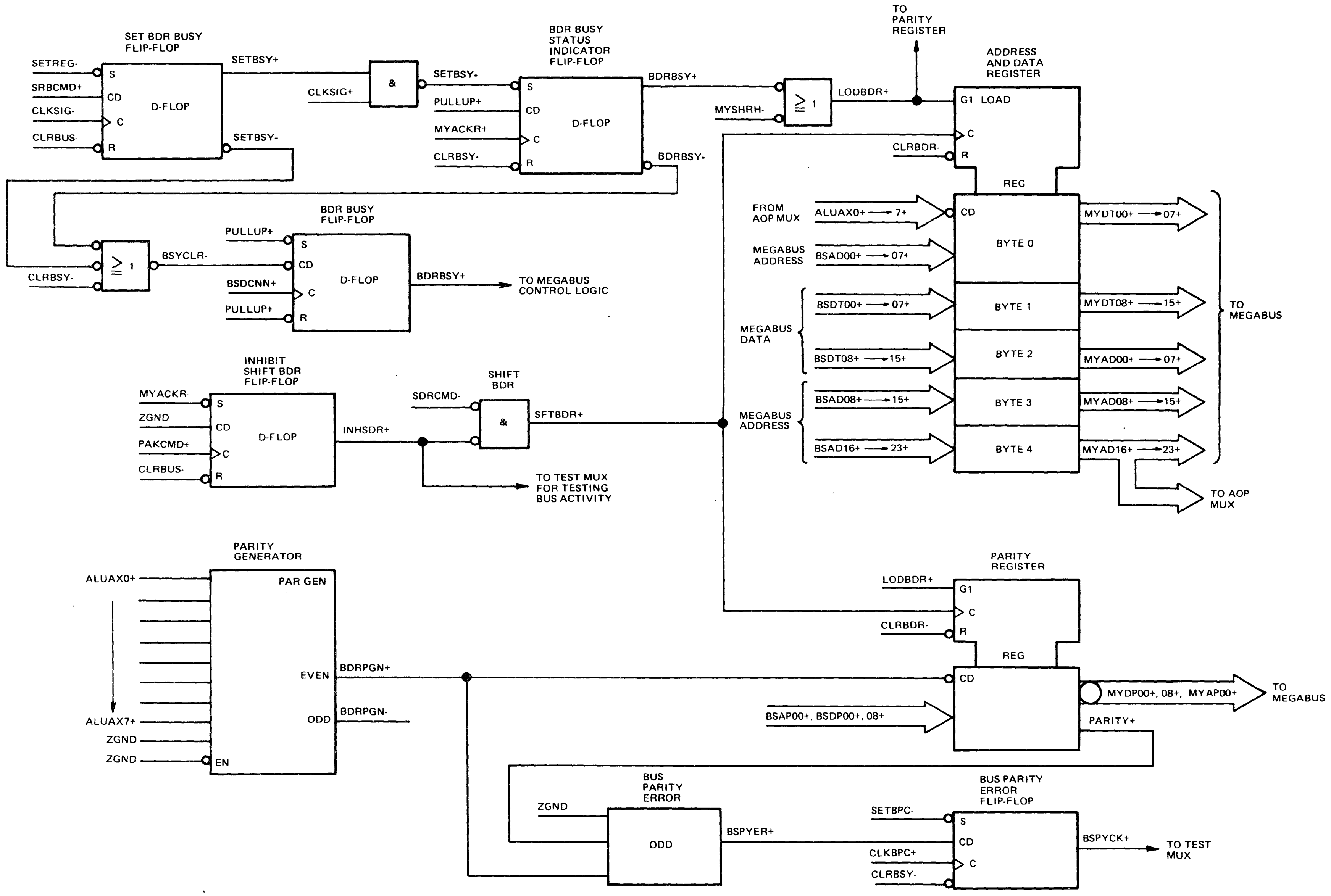


Figure 3-8 Bus Data Register

shifted down one byte position. At CLKSTB+ time the output of the AOP multiplexer is reloaded into byte position 0, effectively performing an end-around-byte shift. When performing a Load BDR operation, the same procedure is followed except that the output of the AOP multiplexer reflects one of the seven other firmware-visible registers (refer to subsection 3.3.1).

The parity bit for a Load BDR operation is computed on the output of the AOP multiplexer and written into the parity register in the position corresponding to byte 0. During a BDR unload, the parity is also generated on the output of the AOP multiplexer (data from the BDR). This parity is compared with the parity output of the BDR for detection of a Bus Parity Error (BSPYER+). If the two compared inputs are not equal, the BSPYER+ sets, and the Bus Parity Error flip-flop (BSPYCK+) sets at CLKBPC+ time. Table 3-5 indicates the BDR configuration when the MTC is the master (initiator of bus request) and when it is the slave (receiver of bus request) for each byte during a Magabus cycle.

3.5.2 Bus Data Register Control (Figure 3-8)

The bus data register is an input/output buffer for the Megabus and is, therefore, subject to simultaneous access by the MTC and by incoming Megabus cycles. Controls are required to limit register access to one user at a time. To accomplish this, five cycle parameters (see Table A of Figure 3-9) and several gates serve to inhibit or enable various register operations.

The Bus Data Register Busy (BDRBSY+) flip-flop controls Megabus access to the BDR. This flip-flop is clocked at the leading edge of each Megabus cycle by the signal BSDCNN+ and the Megabus interface logic is busy if any of the following conditions are met.

1. The BDR is busy as indicated by the BDR Busy status indicator (BDRBSY+) flip-flop.

2. The MTC firmware is in the process of setting the BDR busy flip-flop as indicated by the Set BDR Busy (SETBSY+) flip-flop.

3. The MTC is in the process of resetting the BDR Busy flip-flop as indicated by the CLRBSY- signal.

The BDR Busy flip-flop being set inhibits the Megabus from accessing the BDR and normally causes a Wait response to be generated for the Megabus.

The BDR Busy status indicator flip-flop can be set by the MTC firmware or by the Megabus hardware. The output of this flip-flop is examined at the leading edge of each Megabus cycle to determine if an Acknowledge signal can be generated. The output also generates the load pulse for the BDR (LODBDR+).

Table 3-5 Bus Data Register Application

BYTE	MTC MASTER	MTC SLAVE
0	MSB Data	MSB Address
1	LSB Data	MSB Data
2	MSB Address	LSB Data
3	MID Address	MID Address
4	LSB Address	LSB Address
NOTE Shift input is loaded into byte 0 from the AOP multiplexer. Register output is obtained from byte 4 and delivered to the AOP multiplexer.		

When the MTC firmware requires access to the BDR, it must first set the register to the busy state in order to inhibit the clearing of the register and to inhibit Megabus access. To accomplish this, a Set Register Busy command is executed, which sets the SETBSY+ flip-flop for one clock cycle. The SETBSY+ flip-flop being set enables the BDR Busy status indicator flip-flop at the initiation of the subsequent clock cycle, disabling the LODBDR+ function.

The Megabus can set the BDR Busy status indicator flip-flop with the leading edge of the Acknowledge (ACK) signal generated by the MTC. In either case, whether this flip-flop is set by firmware through the SETBSY+ or with the leading edge of ACK, the BDR Busy status indicator flip-flop stays set until CLRBSY- is activated. The signal CLRBSY- can be activated by the firmware through a firmware command or by the Megabus as a result of a Master Clear.

An ACK to the Megabus by the MTC sets the Inhibit Shift BDR (INHSDR) flip-flop. This flip-flop inhibits the firmware from performing operations on the BDR until a Processor Acknowledge command (PAK) is executed by firmware, acknowledging the Megabus cycle. A PAK command resets the INHSDR+ flip-flop and allows the firmware to unload the BDR. If an ACK has been generated while the firmware is trying to set the BDR busy, the INHSDR+ flip-flop sets, keeping the firmware from accessing the BDR.

3.5.3 Master Cycle Logic and Priority network (Figure 3-9)

When the MTC as the master initiates a transfer over the Megabus to the main memory or to the central processor, it first sets the BDR Busy flip-flop and loads the register with address and data. It then loads the cycle parameters into an AOP

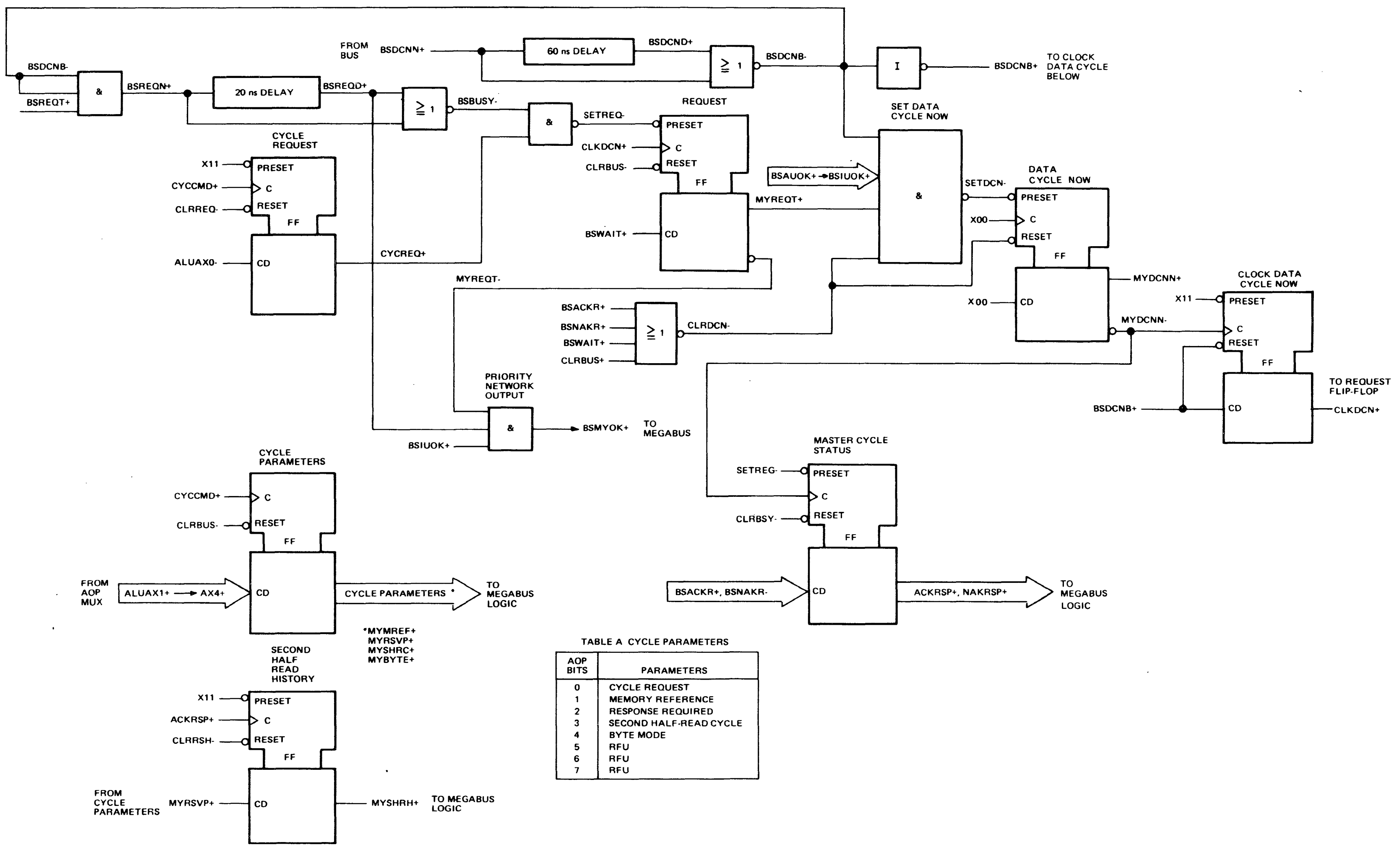


TABLE A CYCLE PARAMETERS

AOP BITS	PARAMETERS
0	CYCLE REQUEST
1	MEMORY REFERENCE
2	RESPONSE REQUIRED
3	SECOND HALF-READ CYCLE
4	BYTE MODE
5	RFU
6	RFU
7	RFU

Figure 3-9 Master Cycle Logic and Priority Network

register (refer to subsection 3.3.1) and issues a Cycle command. The master cycle logic has control of the Megabus cycle once the Cycle command is executed, and firmware must test the Master Cycle status flipflops to determine when the Megabus cycle is complete. When a Cycle command is executed, the cycle parameters are set according to the outputs of the SOP multiplexer. In addition, the Cycle Request (CYCREQ+) flip-flop sets, initiating a Megabus request. Table A of Figure 3-9 shows the usage of the cycle parameters.

When the CYCREQ+ flip-flop is set and when the Megabus is not busy (BSBUSY-), the Set Request (SETREQ-) signal becomes active, causing the Request flip-flop (MYREQT+) to set. The Request flip-flop goes to one of the inputs of the function Set Data Cycle Now (SETDCN-). This Megabus request inhibits other units on the Megabus from initiating a new request and causes the output of the MTC priority network (BSMYOK+) to go low.

The priority network consists of nine input signals (BSAUOK+ through BSIUOK+) and one output signal (MYDCNN-). The MTC has priority when all the inputs are high, indicating that no other unit with higher priority has a Megabus request active. The priority network is sampled when the Megabus is inactive, as determined by the CLRDCN- and BSDCNB+ functions. The priority network output being high to other units on the Megabus with lower priority shows that the MTC does not have a request active and that the unit generating the BSIUOK+ signal coming into the MTC does not have a request active.

With the Request flip-flop set and with the completion of any previous Megabus cycle (BSDCNB+), the MTC sets its Data Cycle Now (MYDCNN) flip-flop provided it has the highest priority on the Megabus. When all conditions of SETDCN- are met, the MYDCNN flip-flop sets and stays set until the slave unit responds. The response of the slave (CLRDCN-) clears the MYDCNN- flip-flop, resetting the request unless the response is a Wait, in which case the Request flip-flop remains set.

An ACK Response (ACKRSP+) by the slave to a first half read cycle sets the Second Half Read History (MYSHRH+) flip-flop. The MYSHRH+ flip-flop generates the load and clear signals to the BDR during the second half read cycle. This is necessary because the BDR becomes busy prior to the first half read cycle. The busy condition inhibits other units on the Megabus from accessing the MTC's BDR between first half and second half read cycles and does not allow the MTC to acknowledge the slave's response. The MYSHRH+ flip-flop aids in differentiating between the slave's response and other cycles addressed to the MTC.

3.5.4 Slave Response Logic (Figure 3-10)

The MTC, as the slave, can respond to a Megabus cycle in four ways which are prioritized and have the following significance.

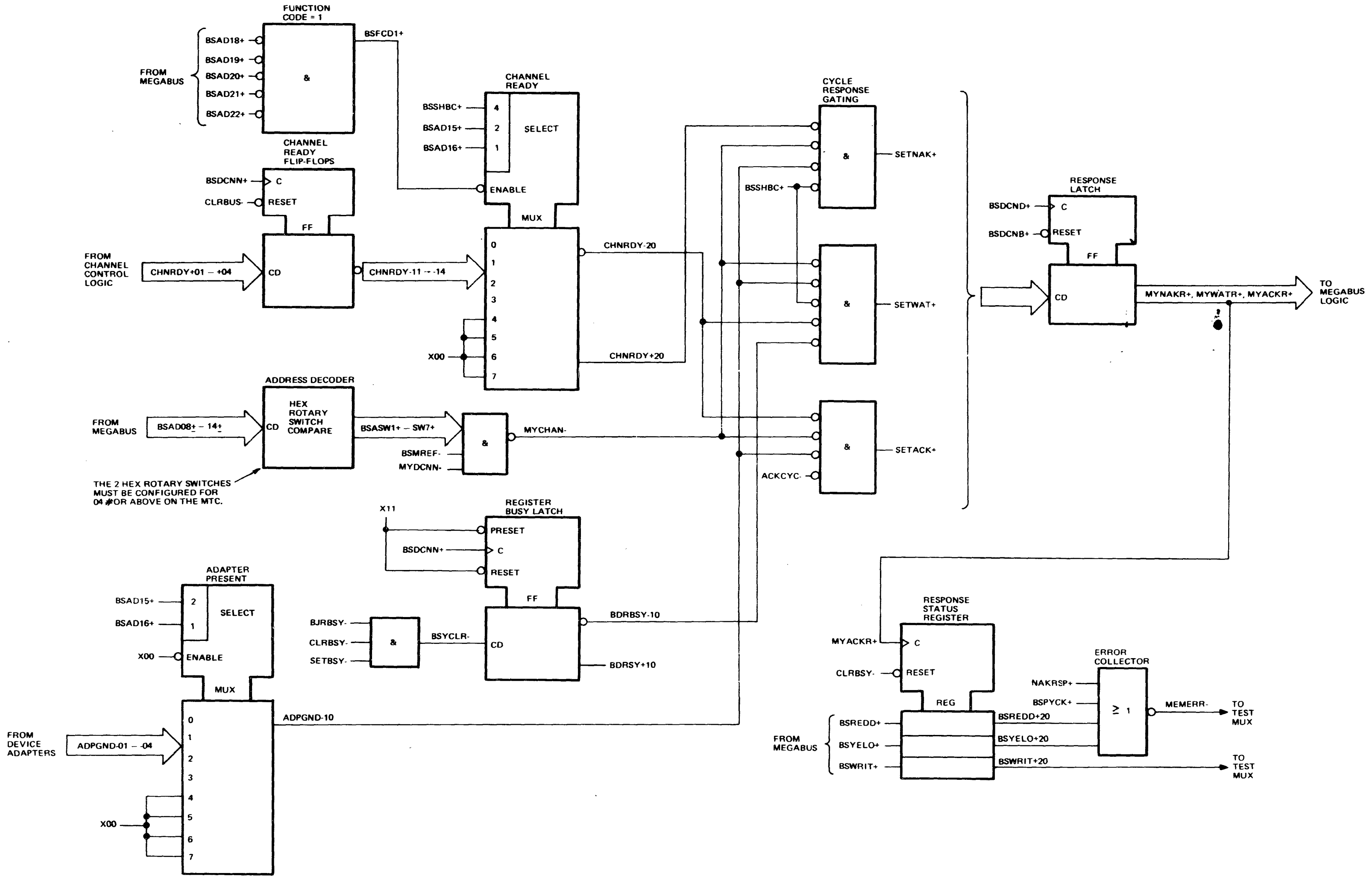


Figure 3-10 Slave Response Logic

1. No response indicates that the addressed channel has no associated device adapter installed.

2. A NAK response indicates that the addressed channel is not in a ready condition. A channel may be busy doing a device operation.

3. A Wait response indicates that the addressed channel is ready but that the BDR is busy. The master unit should retry the Megabus cycle.

4. An ACK response indicates that the addressed channel is ready and that the MTC has stored the information on the Megabus in the BDR.

Each device adapter has an output level (ADPGND-01 through -04) which indicates that the adapter is installed or a tape device address (channel) is allowed. The level from each adapter goes to the adapter present multiplexer, which selects the level corresponding to the addressed channel. The output of the adapter present multiplexer (ADPGND-10) is required by the cycle response logic. A high output indicates that an adapter is not installed, inhibiting a response by the MTC.

Firmware uses four channel ready flip-flops to store the status of each channel at the leading edge of each Megabus cycle. The outputs of these flip-flops (CHNRDY-11 through -14) are gated to the channel ready multiplexer. The state of the addressed channel, as reflected by the channel ready flip-flop, is selected by the channel ready multiplexer and fed to the cycle response logic. The address of the channel is augmented by the Second Half Bus Cycle (BSSHBC+) signal, causing the addressed channel to appear ready for any second half read cycle. The channel can also be forced to the ready state whenever the function code on the address bus is a One (Initialize).

The register busy latch stores the status of the BDR at the start of each Megabus cycle. As described in subsection 3.5.1, the BDR is considered busy if it is being set busy, already busy, or being reset from the busy state. The output of the register busy latch goes to the cycle response logic and is used to generate a Wait response.

The last input to the cycle response logic is the output of the address decoder which determines if the MTC is being addressed by the Megabus. The decoder, consisting of two hexadecimal rotary switches and gating, compares the address on the Megabus against the address of the MTC. The address of the MTC is determined by the setting of the switches and is not visible to the MTC until the first Megabus cycle acknowledged by the MTC. The decoder output is active (low) when bits 8 through 14 of the Megabus agree with the switch settings and when the Memory Reference (BSMREF-) signal on the Megabus is low. Bits 15 and 16 of the Megabus (BSAD15+, BSAD16+) determine which of the four MTC channels is being addressed and are not examined by the decoder. These bits are the selection bits for the adapter present and channel ready multiplexers.

The cycle response gating generates the ACK, NAK, or Wait response if the MTC is addressed and the addressed channel has an adapter installed. The output of the response gates is stored in the response latch 60 nanoseconds after the cycle was initiated with BSDCND+ coming high. The output of the response latch goes to the Megabus drivers and to various logic on the MTC (i.e., the ACK response latch is used to clock the Response Status Register).

The Response Status Register stores pertinent Megabus signals, including write mode (BSWRIT+) and red (BSREDD+)/yellow (BSYELO+) status signals. These signals (except BSWRIT+), along with the NAK response (NAKRSP+) and the bus parity check (BSPYCK+), are fed to the error collector, which transmits its output (MEMERR-) to the test multiplexer.

3.6 CHANNEL CONTROL FUNCTIONAL DESCRIPTION

3.6.1 Channel Ready Signals

The channel ready (CHNRDY) register (Figure 3-11) stores the ready state of a device channel (set or reset). A channel is normally ready unless a device operation or a software interrupt is in progress on the device channel. The CHNRDY register is enabled by the decode of a firmware command (CHNRDE-), and selection of the CHNRDY register is achieved by the Scratch Pad Memory Index Register bits (SPMIR0+, SPMIR1+). The selected channel ready signal output is set or reset according to the state of the UPIR bit 12 (UPIR12+) of the firmware command which enabled the CHNRDY register. The outputs of the register (CHNRDY+01 through +04) are sent to the Channel Ready flip-flops (Figure 3-10).

3.6.2 Adapter Enable Gate

The Adapter Enable gate (Figure 3-11) generates one enable signal (ADPENB+) for each device channel, using the decoded contents of the SPMIR (SPMIE0- through SPMIE3-) to determine which device adapter is to be enabled. Those device adapters which are not enabled ignore all the control signals on the MTC/adapter interface and also disable their data selector outputs, allowing the MTC to communicate with a single device adapter at one time. The Adapter Enable gate outputs (ADPENB+01 through +04) do not affect the dialogue between the device and device adapter whether selected or not.

3.6.3 Adapter Control Signals

The adapter control signals (Figure 3-11) control the device adapters. These control signals (ADPPLS+, ADPDBT+, ADPCD1+

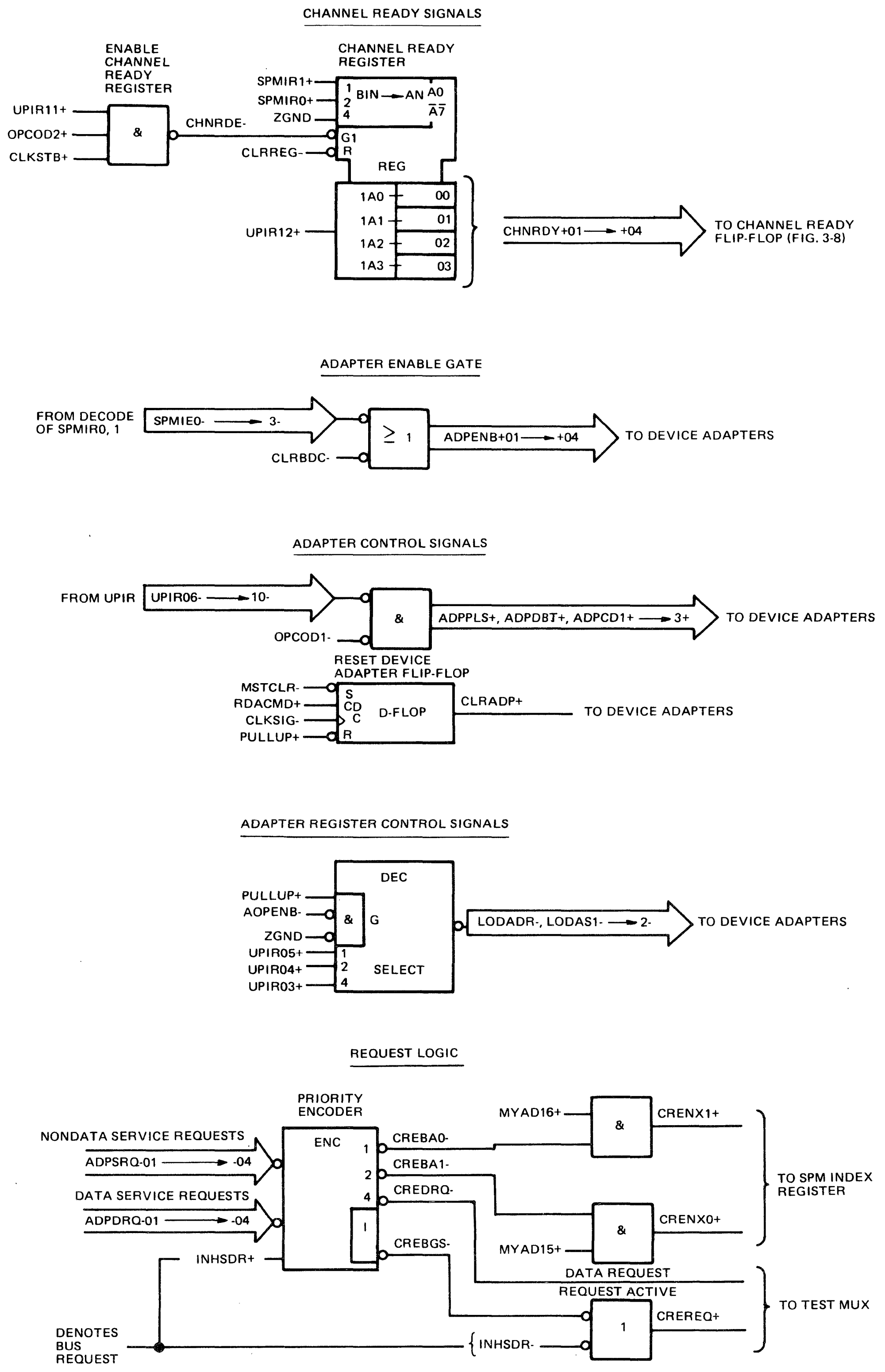


Figure 3-11 Channel Controls

through ADPCD3+) are generated from a firmware command and the UPIR bits (UPIR06- through UPIR10-) of the same command. The reset signal (CLRADP+) sent to the device adapters is a result of a Master Clear (MSTCLR-) or firmware command (RDACMD+).

3.6.4 Adapter Register Control Signals

The adapter register control signals (Figure 3-11) load data into device adapter registers and are developed by an octal-to-decimal decoder. The enable (AOPENB-) for the decoder is developed by either a Constant command or by an ALU command, with the result stored in the device adapter register. The decoding of the device adapter register to be loaded is achieved by the UPIR bits (UPIR03+ through UPIR05+) of the same firmware command that enabled the decode. The decoded outputs Load Data Register (LODADR-), Load Status 1 (LODAS1-), and Load Status 2 (LODAS2-) are sent to the enabled device adapter.

3.6.5 Request Logic

The request logic (Figure 3-11) determines the channel activity required by the MTC. The nondata service requests (ADPSRQ-01 through -04) and data service requests (ADPDRQ-01 through -04) from each device adapter are fed to the inputs of the priority encoder. When the MTC receives a device service request (i.e., data or nondata) and the priority encoder is enabled by the absence of a bus request (INHSDR+ is low), the high-order output (CREDRQ-) indicates the type of request active. The two low-order outputs (CREBA0-, CREBA1-) represent a binary decode of the requesting device channel. Since the address bits (MYAD15+, MYAD16+) from the bus data register are high due to firmware continually clearing the BDR, the SPMIR is loaded with the requesting device channel number.

For a bus request, the enable input (INHSDR+) of the decoder is high and forces all the priority encoder outputs high, ignoring the inputs. The Request Active signal (CRERERQ+) is set by INHSDR-. When the firmware determines that a request is active (CREREQ+), the bus request is detected by examination of INHSDR+. With the priority encoder outputs high, the SPMIR is loaded with the device selection bits (MYAD15+, MYAD16+) from the BDR.

3.7 TEST MULTIPLEXER FUNCTIONAL DESCRIPTION (Figure 3-12)

Firmware intelligence is derived from the capability of testing data, status, and errors within the subsystem. Test commands specify a test item and a test condition. The next sequential command is skipped if the state of the test item equals the specified test condition.

All test items specified by a 5-bit field go to one of the test multiplexers whose outputs are TSTXB0+ through TSTXB3+.

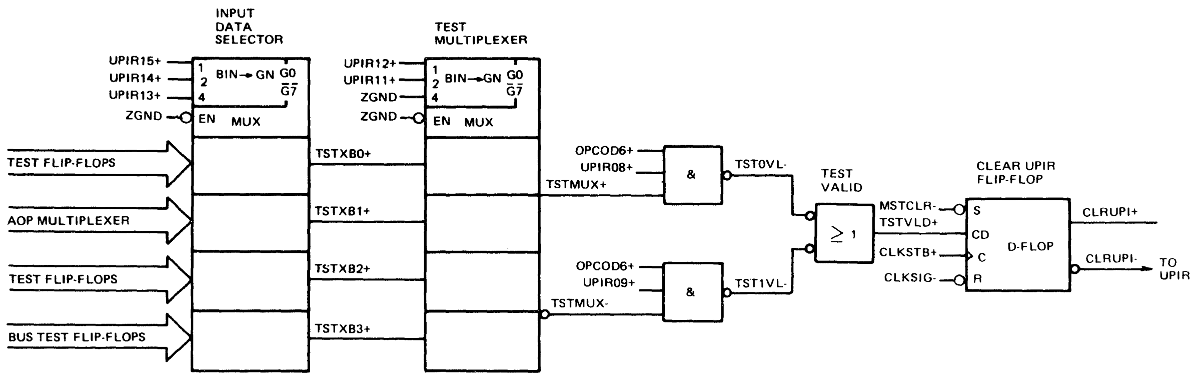


Figure 3-12 Test Multiplexer

These multiplexers use the three low-order bits of the UPIR 5-bit field to select one of eight inputs. The high-order two bits of the test item field select the test item inputted to the multiplexer (TSTMUX+). The state of the test item, as defined by the output of TSTMUX+, is compared against the test condition, and the Test Valid signal (TSTVLD+) goes high if the comparison is true. A valid test is recorded in the Clear Microprogram Instruction Register (CLRUPI-) flip-flop, which clears the UPIR during the next clock pulse (performs a NOP). If the test is not valid, the UPIR is not cleared, and the next sequential instruction is executed normally.

3.8 CLOCK AND CLOCK CONTROL FUNCTIONAL DESCRIPTION (Figure 3-13)

The output of an 8-MHz crystal oscillator (CLKOCS+) is divided to obtain a 250-nanosecond clock cycle. Two clock signals (CLKSIG+) and a clock strobe (CLKSTB+) are developed during each clock cycle for controlling the MTC and adapter hardware.

The Clock Signal (CLKSIG+) flip-flop is a J-K flip-flop which changes state at the negative edge of the CLKOSC+00 signal as long as the inputs are high. When the Clock Halt (CLKHLT) flip-flop is set, the J-input goes low causing the clock signal to be reset or to remain reset. The assertion and negation of the CLKSIG+ flip-flop are inverted for distribution throughout the MTC. The negation clock signal, after inversion, is fed to a 100-nanosecond delay line. The 40-nanosecond tap (CLKDLY+03) and the 90-nanosecond tap (CLKDLY+08) generate strobe CLKSTB+, which occurs during the later part of each clock cycle. The clock, a minimum of 35 nanoseconds in duration, occurs at a point in the cycle which meets data setup and hold times sufficient for writing into the scratch pad memory. CLKSTB+ also clocks registers within the MTC and device adapters.

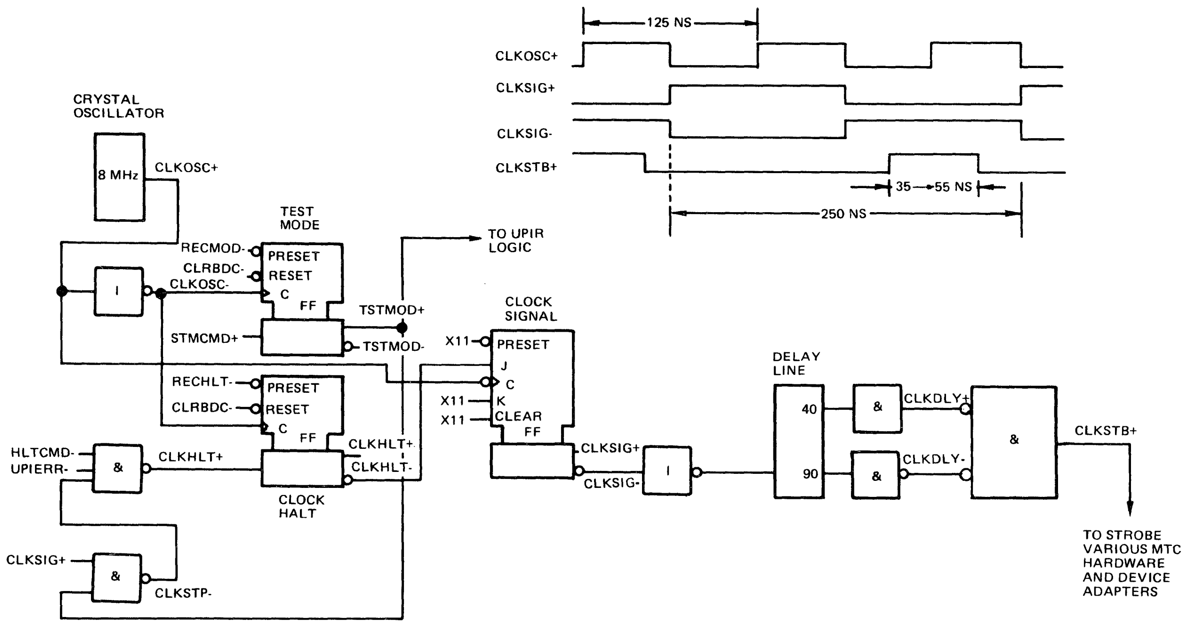


Figure 3-13 Clock and Clock Control

The Clock Halt (CLKHLT) flip-flop is set when:

1. Detecting a critical error and stopping the MTC
2. Entering the test mode
3. Detecting a microprogram control store error.

The CLKHLT flip-flop is set by execution of a firmware Halt command (HLTCMD). In the test mode, the Halt condition is recirculated by the signal RECHLT- until a Master Clear signal occurs. When the Test Mode (TSTMOD) flip-flop is on, the RECHLT- function goes inactive each time the MTC acknowledges a Megabus cycle. The next transition of the oscillator (CLKOSC-) causes the CLKHLT flip-flop to be reset, putting a One at the J-input to the Clock Signal (CLKSIG) flip-flop, which is set at the second transition of CLKOSC+. CLKSIG+ feeds back to the CLKHLT flip-flop, causing it to be set during the third transition of CLKOSC-. In this manner, one clock cycle is generated in the test mode for each Megabus transfer to the MTC.

The TSTMOD flip-flop is set by the firmware command Set Test Mode (STMCMD+) and remains set due to the Recirculate Test Mode signal (RECMOD-) until a Master Clear signal occurs or a command is received on the Megabus which resets the Test Mode and the Clock Halt flip-flops. In addition to controlling the stepping of the clock, the Test Mode flip-flop controls the input to the UPIR. When set, TSTMOD+ disables the microprogram control store input to the UPIR and enables the test mode input (refer to subsection 3.2.5 of this manual).

IV
THEORY OF OPERATION - CYCLE FLOW

4.1 MICROINSTRUCTIONS

A microinstruction and its purpose are defined by the bit structures within a firmware word (refer to subsection 2.3.1 of this manual). A combination of one or more microinstructions is collectively referred to as a firmware command (see Figure 4-1). Sequences of firmware commands utilized to perform a particular function are known as microprograms or firmware routines. In the MTC there are eight basic types of firmware commands, each of which is subdivided into various bit configurations (micro-instructions) to perform a specific operation. The eight basic types of firmware commands are:

- Miscellaneous
- Device Adapter
- Megabus Logic
- ALU
- Constant
- Memory
- Test
- Branch

Each of the major categories of firmware commands is identified by a particular op-code. The op-code is the coding of bits 0, 1, and 2 of the microprogram control store word (firmware command).

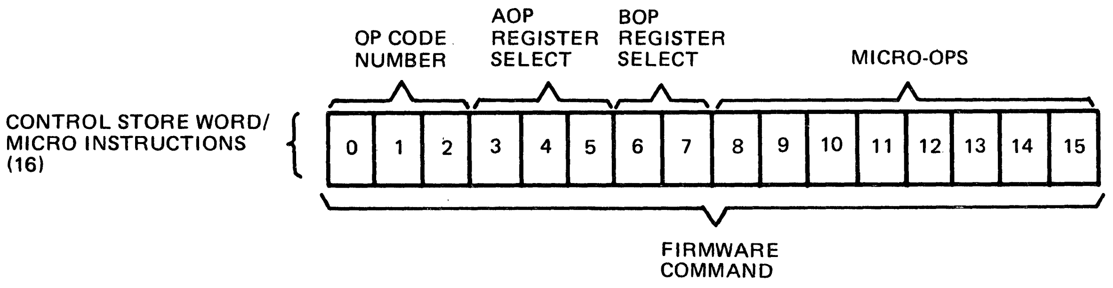


Figure 4-1 Microinstruction General Field Format

4.1.1 Miscellaneous Commands

Miscellaneous commands, which have an op-code of 000, are used primarily to perform clear and set operations on registers and flip-flops in the MTC and the device adapters. These commands are comprised, typically, of a single microinstruction which is contained within bits 3 through 15 of the microprogram control store word. Table A of Figure 4-2 lists the microinstructions within the miscellaneous category, their mnemonics, and the binary and hexadecimal values for each word.

4.1.2 Device Adapter Commands

Device adapter commands have an op-code of 001 and are used to generate control signals going from the MTC to the device adapter. The device adapter which is enabled as a result of the scratch pad memory index register (refer to subsection 3.4.2) is the only one of the three possible adapters which reacts to and executes the command.

Device adapter commands use microinstructions which utilize the ALU A-operand multiplexer in conjunction with device-specific microinstructions to develop a complete firmware command. Table B of Figure 4-2 is a listing of the device adapter commands, their mnemonics, and the hexadecimal and binary values for each word.

4.1.3 Megabus Logic Commands

Megabus logic commands have an op-code of 010 and are a result of a combination of microinstructions particular to the bus command and ALU A-operand microinstructions. The resulting firmware command is utilized to perform control functions on hardware associated with the Megabus. Table C of Figure 4-2

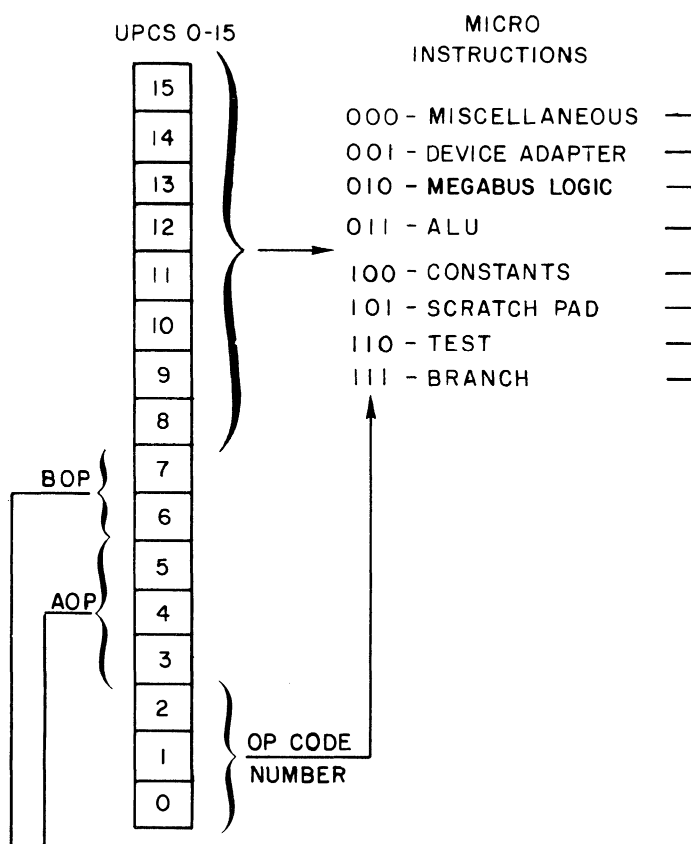


Table A Miscellaneous Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
No Operation	0000000000000000	NOP	0000
Clear	0001000000000000	CLR	1000
Set Error And Status Flip-Flops	0000100000000000	SEF	0800
Initialize Bus Logic	0000010000000000	IBL	0400
Enter Test Mode/Halt	0000000011000000	ETM	00C0
Reset Diagnostic Mode	0000000100000000	RSD	0100
Set Diagnostic Mode	0000000010000000	STD	0080
Halt	0000000001000000	HLT	0040
Clear Registers, Flip-Flops	0000000000010000	CRF	0010
Reset Device Adapter	0000000000001000	RDA	0008
Set Qlt Flip-Flop	0000000000000100	QLT	0004
Initialize	0000010000010000	INI	0410

Table B Device Adapter Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Pulse	0010001000000000	PLS	2200
Data Byte Taken	0010000100000000	DBT	2100
Adapter Code 1	001AAA0010010100	AC1	N/A
Adapter Code 2	001AAA0001010100	AC2	N/A
Adapter Code 3	001AAA0000110100	AC3	N/A

AAA = Selects AOP Multiplexer Input

Table C Megabus Logic Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Data Register Taken	0100000100000000	DRT	4100
Processor Acknowledge	0100000010000100	PAK	4084
Shift BDR, Clear Inhibit	0100000011000100	SAK	40C4
Set Channel Ready	0100000000011000	SCR	4018
Shift Data Register	010AAA0001000000	SDR	N/A
Set Register Busy	0100000000000100	SRB	4004
Reset Interrupt Latch	0100000000000001	RIL	4001
Reset Channel Ready	0100000000010000	RCR	4010
Clear Bus Status	0100000100000100	CBS	4104
Clear Bus Logic	0100000100100100	CBL	4124
Cycle Bus	010AAA0000100000	CYC	N/A

AAA - Used As Register Selection Bits

Table D ALU Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Add A to B	011AAABBCS100101	ADD	N/A
AND A to B	011AAABBCS101110	AND	N/A
AOP Negation to ACU	011AAABBCS000010	ANT	N/A
BOP Negation to ACU	011AAABBCS010110	BNT	N/A
AOP Minus One	011AAABBCS111101	DEC	N/A
AOP Plus One	011AAABBCS000000	INC	N/A
Left Shift AOP	011AAABBCS110001	LSH	N/A
NAND A to B	011AAABBCS010010	NND	N/A
NOR A to B	011AAABBCS000110	NOR	N/A
OR A to B	011AAABBCS111010	ORR	N/A
Subtract B from A	011AAABBCS011000	SUB	N/A
AOP to ACU	011AAABBCS111110	XFA	N/A
BOP to ACU	011AAABBCS101010	XFB	N/A
XNOR A to B	011AAABBCS100110	XNR	N/A
XOR A to B	011AAABBCS011010	XOR	N/A
Zero to ACU	011AAABBCS001110	ZER	N/A

AAA - AOP register selection
BB - BOP register selection
C - Determines carry IN
S - Determines A or B Result Storage
S = 0 = B
S = 1 = A

Table E AOP Multiplexer Input Selection

UPIR BIT CONFIGURATION				
Address Bits 03	04	05	SELECTED REGISTER	MNEMONIC
0	0	0	Accumulator	ALUAC0 → 7
0	0	1	Scratch Pad Data	SPMOT0 → 7
0	1	0	Scratch Pad Address	SPMAC0 → 7
0	1	1	Bus Data	MYAD16 → 23*
1	0	0	Adapter Data	ADPDS0 → 7
1	0	1	Adapter ID	ADPDS0 → 7
1	1	0	Adapter Status 1	ADPDS0 → 7
1	1	1	Adapter Status 2	ADPDS0 → 7

*The data seen at this point depends on the number of shifts performed on the bus data shift register.

Table F BOP Multiplexer Input Selection

UPIR BIT DECODE			
Address Functions ALUBS1	ALUBS2	SELECTED DATA INPUT	MNEMONIC
0	0	Accumulator	ALUAC0 → 7
0	1	Scratch Pad Data	SPMOT0 → 7
1	0	Scratch Pad Address	SPMAC0 → 7
1	1	UPIR Constant	UPIR06 → 10,12,14,15

Table G Constant Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
AOP ANDed to Constant	100AAACCCCC0C1CC	ACN	N/A
Load Constant to AOP	100AAACCCCC0C0CC	LCN	N/A
OR AOP and Constant	100AAACCCCC1C0CC	OCN	N/A

AAA - Register selection
C - Constant value to be operated with

Table H Memory Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Load Channel 0	1010000000110000	LC0	A030
Load Channel 1	1010000000110100	LC1	A034
Load Channel 2	1010000000111000	LC2	A038
Load Channel 3	1010000000111100	LC3	A03C
Load Requesting Channel	1010000000100000	LRC	A020
Memory Write	101AAA10000000X0	MWT	N/A
Reset Index Mode	1010000010000000	RIM	A080
Check Parity Error	101AAA0000000010	RPC	N/A
Set Index Mode	1010000011000000	SIM	A0C0
Memory Write + Increment	101AAA11000000X0	WIA	N/A

AAA - AOP Register Selection
X = 1 = Check Parity Error X = 0 = Don't Check Parity Error

Table J Test and Return Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Go to via SRAR	1100001000000000	RTN	C200
Skip if test = 1	110AAA0010TTTTTT	TFO	N/A
Skip if test = 0	110AAA0001TTTTTT	TFZ	N/A

AAA - AOP register select
TTTTTT - Test multiplexer input select

Table K Test Parameters

MNEMONIC	FUNCTION	HEX CODE	DESCRIPTION
TACK	ACKRSP+00	07	Bus ACK response
TAX0	ALUAX0-00	08	AOP multiplexer, bit 0
TAX1	ALUAX1-00	09	AOP multiplexer, bit 1
TAX2	ALUAX2-00	0A	AOP multiplexer, bit 2
TAX3	ALUAX3-00	0B	AOP multiplexer, bit 3
TAX4	ALUAX4-00	0C	AOP multiplexer, bit 4
TAX5	ALUAX5-00	0D	AOP multiplexer, bit 5
TAX6	ALUAX6-00	0E	AOP multiplexer, bit 6
TAX7	ALUAX7-00	0F	AOP multiplexer, bit 7
TBSY	BDRBSY+00	18	BDR busy indicator
TCOT	ALUCOT+00	05	ALU carry out
TDRQ	CREDRQ-00	15	Data request
TEQF	ALUEQF+00	04	ALU outputs = FF
TEQZ	ALUEQZ+00	03	ALU outputs = 00
TERR	MEMERR-00	16	Cycle error flag
TIDS	INHSDR+00	19	Inhibit data shift
TIE0	SPMIE0-00	10	Index = 0
TIE1	SPMIE1-00	11	Index = 1
TIE2	SPMIE2-00	12	Index = 2
TIE3	SPMIE3-00	13	Index = 3
TINT	RESINT+00	1A	Resume interrupt
TLON	LOGIC1+01	01	Logic 1 for T & D
TLZR	ZGND	00	Logic 0 for T & D
TNAK	NAKRSP+00	1B	NAK response
TPTY	BSPYCK+00	1D	Bus parity check
TQLT	BSQLT0-00	17	QLT line
TRED	BSREDD+20	1E	Condition red
TREQ	CREREQ+00	06	Channel request
TRSP	BSRSVP+30	02	Bus response required
TYEL	BSYEL0+20	1F	Condition yellow

Table L Branch Commands

OPERATION	BINARY VALUE	MNEMONIC	HEX CODE
Go to	1111AAAAAAAAAAAA	GTO	N/A
Load SRAR	1110AAAAAAAAAAAA	LRA	N/A

A - Represents address to be utilized

Figure 4-2 MTC Firmware Commands

lists the Megabus logic commands, their mnemonics, and the hexadecimal and binary values for each word.

4.1.4 ALU Commands

A-operand microinstructions, B-operand microinstructions, and microinstructions specific to the ALU are used to fully define one ALU command. ALU commands, having an op-code of 011, perform specific logic or arithmetic operations on the contents of the A- and/or B-operand registers. The result is then stored in the A-operand or B-operand as determined by the ALU microinstructions. Mode, Carry Enable, and Carry In are also specified by the ALU microinstructions. Table D of Figure 4-2 is a list of the microinstructions and their mnemonics. The A-operand and B-operand selection fields (see Tables E and F of Figure 4-2) are also shown, and the binary value is given for those bits pertaining to the particular command.

4.1.5 Constant Commands

Constant commands have an op-code of 100 and are comprised of a combination of ALU A-operand microinstructions and two types of constant microinstructions. The two types of constant microinstructions specify a data constant to be used and the type of ALU operation to be performed. Table G of Figure 4-2 lists the types of constant commands, their mnemonics, and also necessary binary bit configurations.

4.1.6 Memory Commands

Memory commands are used to write into the scratch pad memory and to control or alter the scratch pad address. The memory commands have an op-code of 101 and are combinations of ALU A-operand microinstructions and specific memory microinstructions. Table H of Figure 4-2 lists the memory commands, their binary and hexadecimal values, and their mnemonics.

4.1.7 Test and Return Commands

Test commands (op-code 110) allow firmware to determine the state of certain hardware elements in the MTC and device adapters. The parameters that the firmware is capable of testing, their mnemonics, and their hexadecimal values are listed in Table K of Figure 4-2.

Return commands have the same op-code (110) as test commands and allow firmware to load the microprogram address counter with an address that was previously stored in a subroutine return address register. Table J of Figure 4-2 lists test and return commands, their binary and hexadecimal values, and their mnemonics.

4.1.8 Branch Commands

Branch commands have an op-code of 111 and are utilized to load the microprogram address counter or the index subroutine return address register with an address within the firmware word. Table L of Figure 4-2 lists the branch commands, their binary and hexadecimal values, and their mnemonics.

4.2 SCRATCH PAD MEMORY (SPM)

The scratch pad memory is divided into four quadrants with one quadrant dedicated to a particular channel. A quadrant has 64 addressable locations which store 8-bit bytes of data or control information. Table 4-1 shows the topology of a single quadrant indicating the hexadecimal address, mnemonic, and the contents of the location. Note that when a location of SPM is unique to an adapter or is specifically for software usage, the byte is described in the appropriate manual.

When a 16-bit word is used to store information, two SPM locations are required: the most significant byte (MSB) is bits 0 through 7, and the least significant byte (LSB) is bits through 16.

Table 4-2 shows the control words which have specific significance for the MTC firmware, and Figures 4-3 through 4-6 indicate the bit structure of the bytes unique to the MTC.

4.3 FIRMWARE CYCLE FLOW

The MTC firmware is divided into nine major functional areas or routines. Figure 4-7 is an overview flow chart of the firmware routines associated with the MTC. This figure indicates the names, interconnecting paths between routines, and the exit paths to the specific device adapter support routines.

4.3.1 Quality Logic Test

The Quality Logic Test (QLT) is a firmware routine that functionally verifies major logic components of the MTC and tape adapter to determine operational capabilities. This routine is for verification of the MTC and tape adapter only and is not utiliized to determine if any other type of device (printer, etc.) is functional.

The Quality Logic Test Bus subroutine (QLT-BUS) portion of the QLT verifies the MTC bus logic. It is used only for those ports which have a recognizable device identification code (see Table 4-3) and a device in the ready state. Entry into the QLT-BUS subroutine is from the Interrupt routine after the device support firmware has completed the device-specific initialization sequence.

Table 4-1 Scratch Pad Memory Topology
(Sheet 1 of 2)

HEX ADDRESS	MNEMONIC	DESCRIPTION
00	CWD1	Control word, LSB
01	CWD2	Control word, MSB
02	ILC1	Interrupt level, LSB
03	ILC2	Interrupt level, MSB
04	SFC1	Startup function code
05	FWRV*	Firmware revision
06	TSK1**	Task, LSB
07	TSK2**	Task, MSB
08	ADR1	Address, LSB
09	ADR2	Address, MSB
0A	MOD1	Module Address
0B	QLTI***	Quality Logic Test Pass/Fail Flag
0C	RNG1	Range, LSB
0D	RNG2	Range, MSB
0E	Spare	
0F	Spare	
10	CNF1**	Configuration word 1, LSB
11	CNF2**	Configuration word 2, MSB
12	CNF3**	Configuration word 3, LSB
13	CNF4**	Configuration word 4, MSB
14	Spare	
15	Spare	
16	Spare	
17	Spare	
18	STS1**	Status word 1, LSB
19	STS2**	Status word 2, MSB
1A	STS3**	Status word 3, LSB
1B	STS4**	Status word 4, MSB
1C	POLQ**	Poll Queue
1D	POLP**	Poll Pointer
1E	TSKQ**	Task Queue
1F	TSKP**	Task Pointer

Table 4-1 Scratch Pad Memory Topology
(Sheet 2 of 2)

HEX ADDRESS	MNEMONIC	DESCRIPTION
20	DTA1	Data, LSB
21	DTA2	Data, MSB
22	CMSK	Channel mask
23	Spare	
24	MON1	Channel monitor
25	DMA1	DMA control
26	DID1**	Device ID, LSB
27	DID2**	Device ID, MSB
28	CHN1	Channel number, LSB
29	CHN2	Channel number, MSB
2A	CPC1	CP address, LSB
2B	CPC2	Cp address, MSB
2C	IDF1	Interrupt vector, LSB
2D	IDF2	Interrupt vector, MSB
2E	WL01**	Work location 1
3C	WL15**	Work location 15
3D	RICP	Resume interrupt control
3E	TMW1*	Test mode work LOC1
3F	TMW2*	Test mode work LOC2
CB	ENQB***	Enqueue buffer

*Utilized by software only
**Device-specific locations
***Absolute location

Table 4-2 Scratch Pad Memory Word Description
(Sheet 1 of 2)

TERM/MNEMONIC	DESCRIPTION
Control Word (CWD1 and CWD2)	A bit-specific control word which specifies immediate channel operations (see Figure 4-3).
Interrupt Level (ILC1 and ILC2)	The interrupt level word consists of 2 bytes, with bits 0 through 9 containing the CP's channel number. Bits 10 through 15 represent a binary value of from 0 to 63, indicating the priority of the MTC's interrupt to the CP.
Startup Function Code (SFC1)	The start function code is an 07 for the magnetic tape, and an 0D for all other devices. It is used to compare with the function code from the Megabus to determine if the command requires initiation of device action.
Address (ADR1 and ADR2)	This word is the main memory address sent across the Megabus to access information.
Module Address (MOD1)	This byte is part of the main memory address and is used by main memory to select a 64K module.
Range (RNG1 and RNG2)	The range is utilized to determine the number of data bytes to be transferred across the Megabus. This word is decremented for each byte transferred until it reaches zero.
Data (DAT1 and DAT2)	These two byte locations are used for temporary storage of data to or from the device.
Channel Monitor (MON1)	See Figure 4-4 for a bit-specific usage of the channel monitor byte.
DMA Control (DMA1)	See Figure 4-5 for a bit-specific usage of the data management control byte.
Device ID (MSB) (DID2)	This is the MSB of the device identification code. It is supplied by the MTC firmware and is always a 20. The LSB of the ID code is supplied by the adapter.
Channel Number (CHN1 and CHN2)	These two bytes supply the unique channel number of a particular MTC.

Table 4-2 Scratch Pad Memory Word Description
(Sheet 2 of 2)

TERM/MNEMONIC	DESCRIPTION
CP Address (CPC1 and CPC2)	The CP address is utilized by the MTC for loading the address lines of the BDR whenever a request for data transfer to the CP occurs.
Interrupt Vector (IDF1 and IDF2)	The interrupt vector word consists of two bytes, with bits 0 through 9 containing the MTC channel number. Bits 10 through 15 represent a binary value of from 0 to 63, indicating the priority of the MTC interrupt to the CP.
Resume Interrupt Control (RICP)	See Figure 4-6 for a bit-specific usage of the resume interrupt control parameter byte.
Enqueue Buffer (ENQB)	The enqueue buffer has one location and is accessed only in the nonindexed mode. The buffer is used as storage for an indicator byte where bits 0, 1, 2, and 3 are the 'stalled' indicator bits of the respective channels which have read/write orders pending.
Channel Mask (CMSK)	The channel mask is an identifier byte stored for each channel. The mask allows a flag of a specific channel number to be set by an ORing of the channel mask with another parameter byte. The channel numbers are represented with the byte by the four MSBs of the byte. These bits refer to channel numbers 0, 1, 2, and 3, respectively.
Quality Logic Test Flag (QLTI)	The Quality Logic Test flag is set to 00 if the Quality Logic Test fails or to FF if the Quality Logic Test is successful (see Figure 4-7 and subsection 4.3.1).

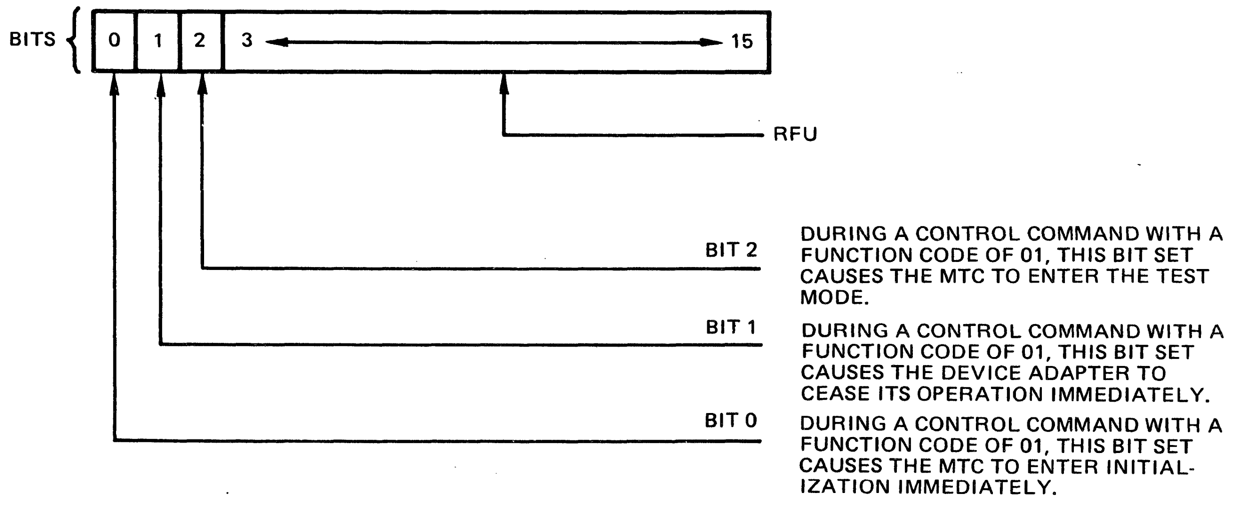

BITS
0
1
2
3
15
RFU
BIT 2
DURING A CONTROL COMMAND WITH A FUNCTION CODE OF 01, THIS BIT SET CAUSES THE MTC TO ENTER THE TEST MODE.
BIT 1
DURING A CONTROL COMMAND WITH A FUNCTION CODE OF 01, THIS BIT SET CAUSES THE DEVICE ADAPTER TO CEASE ITS OPERATION IMMEDIATELY.
BIT 0
DURING A CONTROL COMMAND WITH A FUNCTION CODE OF 01, THIS BIT SET CAUSES THE MTC TO ENTER INITIALIZATION IMMEDIATELY.

Figure 4-3 Control Word Bit Significance

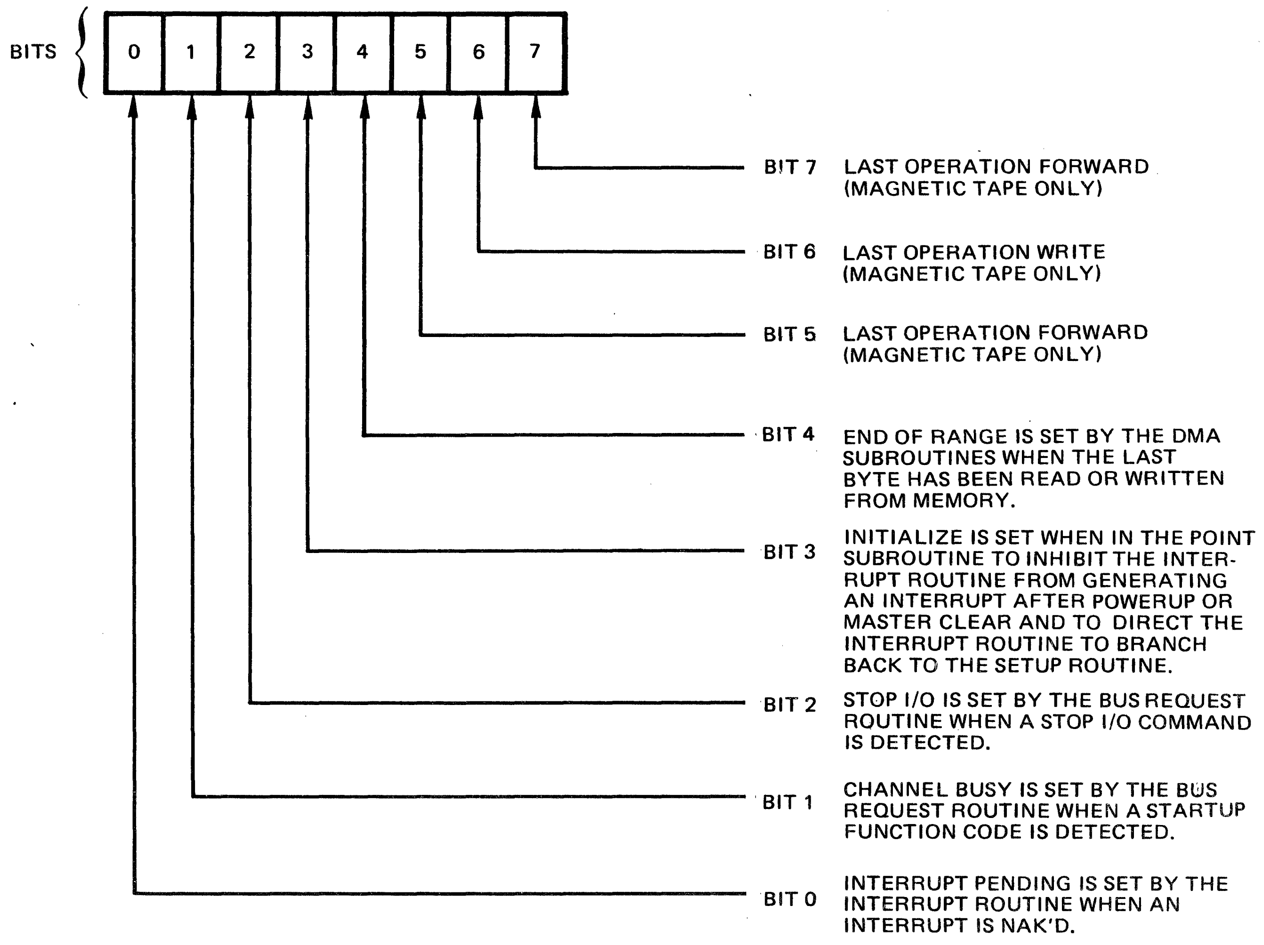

BITS
0
1
2
3
4
5
6
7
BIT 7
LAST OPERATION FORWARD (MAGNETIC TAPE ONLY)
BIT 6
LAST OPERATION WRITE (MAGNETIC TAPE ONLY)
BIT 5
LAST OPERATION FORWARD (MAGNETIC TAPE ONLY)
BIT 4
END OF RANGE IS SET BY THE DMA SUBROUTINES WHEN THE LAST BYTE HAS BEEN READ OR WRITTEN FROM MEMORY.
BIT 3
INITIALIZE IS SET WHEN IN THE POINT SUBROUTINE TO INHIBIT THE INTERRUPT ROUTINE FROM GENERATING AN INTERRUPT AFTER POWERUP OR MASTER CLEAR AND TO DIRECT THE INTERRUPT ROUTINE TO BRANCH BACK TO THE SETUP ROUTINE.
BIT 2
STOP I/O IS SET BY THE BUS REQUEST ROUTINE WHEN A STOP I/O COMMAND IS DETECTED.
BIT 1
CHANNEL BUSY IS SET BY THE BUS REQUEST ROUTINE WHEN A STARTUP FUNCTION CODE IS DETECTED.
BIT 0
INTERRUPT PENDING IS SET BY THE INTERRUPT ROUTINE WHEN AN INTERRUPT IS NAK'D.

Figure 4-4 Channel Monitor Byte Bit Structure

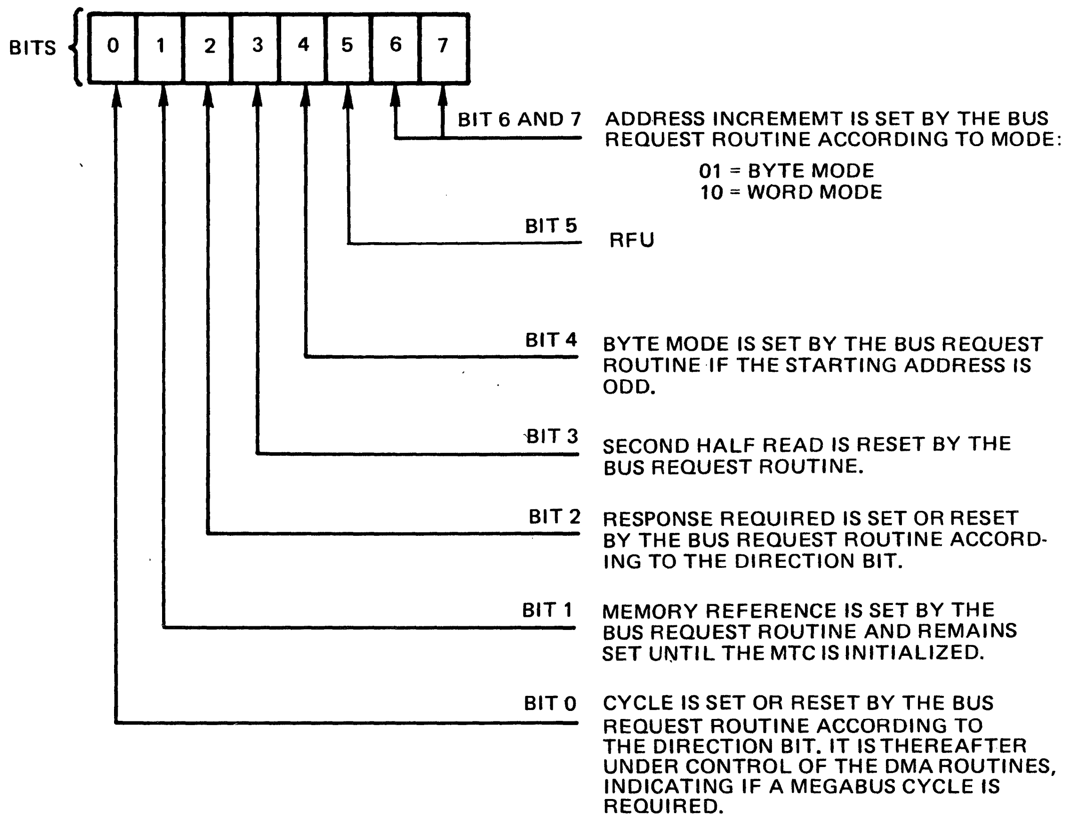

BITS
0
1
2
3
4
5
6
7
BIT 6 AND 7
ADDRESS INCREMEMT IS SET BY THE BUS REQUEST ROUTINE ACCORDING TO MODE:
01 = BYTE MODE
10 = WORD MODE
BIT 5
RFU
BIT 4
BYTE MODE IS SET BY THE BUS REQUEST ROUTINE IF THE STARTING ADDRESS IS ODD.
BIT 3
SECOND HALF READ IS RESET BY THE BUS REQUEST ROUTINE.
BIT 2
RESPONSE REQUIRED IS SET OR RESET BY THE BUS REQUEST ROUTINE ACCORDING TO THE DIRECTION BIT.
BIT 1
MEMORY REFERENCE IS SET BY THE BUS REQUEST ROUTINE AND REMAINS SET UNTIL THE MTC IS INITIALIZED.
BIT 0
CYCLE IS SET OR RESET BY THE BUS REQUEST ROUTINE ACCORDING TO THE DIRECTION BIT. IT IS THEREAFTER UNDER CONTROL OF THE DMA ROUTINES, INDICATING IF A MEGABUS CYCLE IS REQUIRED.

Figure 4-5 DMA Flag Byte Bit Structure

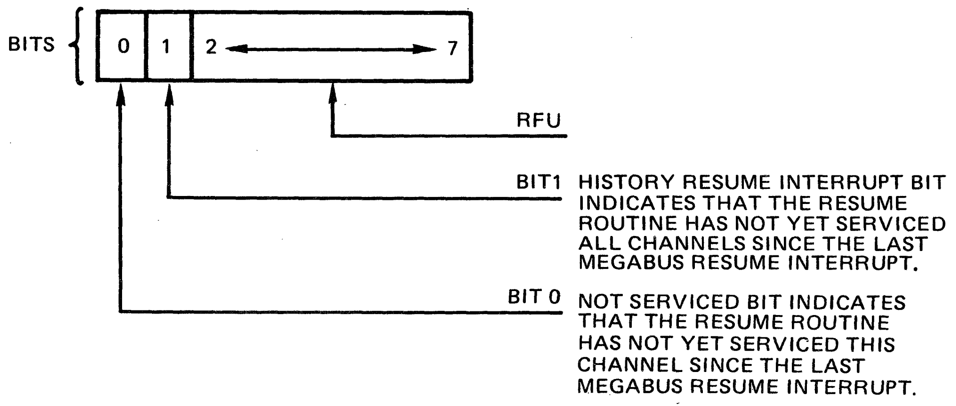

BITS
0
1
2
7
RFU
BIT1
HISTORY RESUME INTERRUPT BIT INDICATES THAT THE RESUME ROUTINE HAS NOT YET SERVICED ALL CHANNELS SINCE THE LAST MEGABUS RESUME INTERRUPT.
BIT 0
NOT SERVICED BIT INDICATES THAT THE RESUME ROUTINE HAS NOT YET SERVICED THIS CHANNEL SINCE THE LAST MEGABUS RESUME INTERRUPT.

Figure 4-6 Resume Interrupt Control Parameter Byte Bit Structure

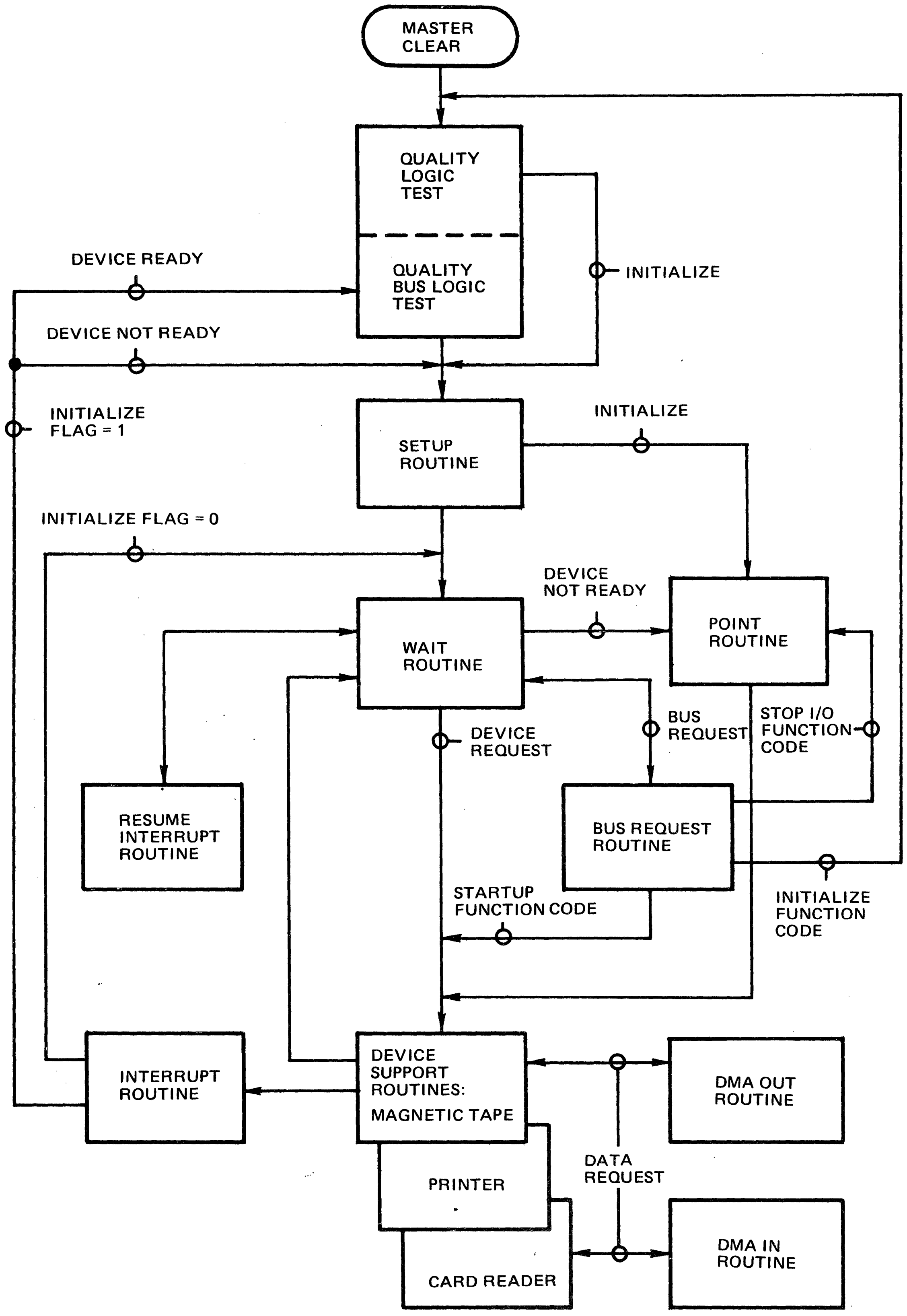


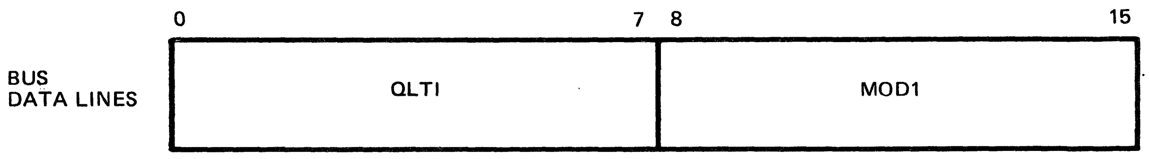
Figure 4-7 Firmware Overview Flowchart

Table 4-3 MTC Device Identification Codes

PERMISSIBLE RANGE OF NUMBERS (Hex)	DEVICE
2000 - 2007	Line Printer/Serial Printer
2008 - 200F	Card Reader
2040 - 205F	9-Channel Magnetic Tape
2060 - 207F	7-Channel Magnetic Tape

The QLT is executed as a result of a Master Clear or the the initialize bit being set in an output control word which resets the QLT Done flip-flop. This results in the forcing of the microprogram address counter to zero and illumination of the red LED located on the front edge of the MTC module. Upon successful completion of the test, firmware sets the QLT Done flip-flop and extinguishes the LED.

The firmware uses the QLTI location of SPM for flag storage to determine if the QLT has been successfully completed. This location of SPM is also available to software for examination via an input function code of hexadecimal 0A. When software reads the QLTI location, the memory module address (MOD1) location is also loaded onto the Megabus data lines but should be disregarded by software. The Megabus data line format is shown in the following example.



WHERE:
QLTI = QLT INDICATOR (FF# IS SUCCESSFUL)
MOD1 = MEMORY MODULE ADDRESS (XNU)

There are basically three failure type conditions reflected by the configuration of the QLTI byte:

- The byte is a hexadecimal 00 indicating that the basic MTC internal QLT failed (i.e., ALU, BDR, SPM test, etc., failure)
- The byte is a hexadecimal F0 which indicates that a QLT-BUS sequence failed.
- Bits 0 through 3 of the byte are selectively reset for individual channel failures on the tape adapter wrap-around sequences (see Figure 4-8 for bit significance). For channels having no device attached in its position, the corresponding bit is defaulted (set) to a One.

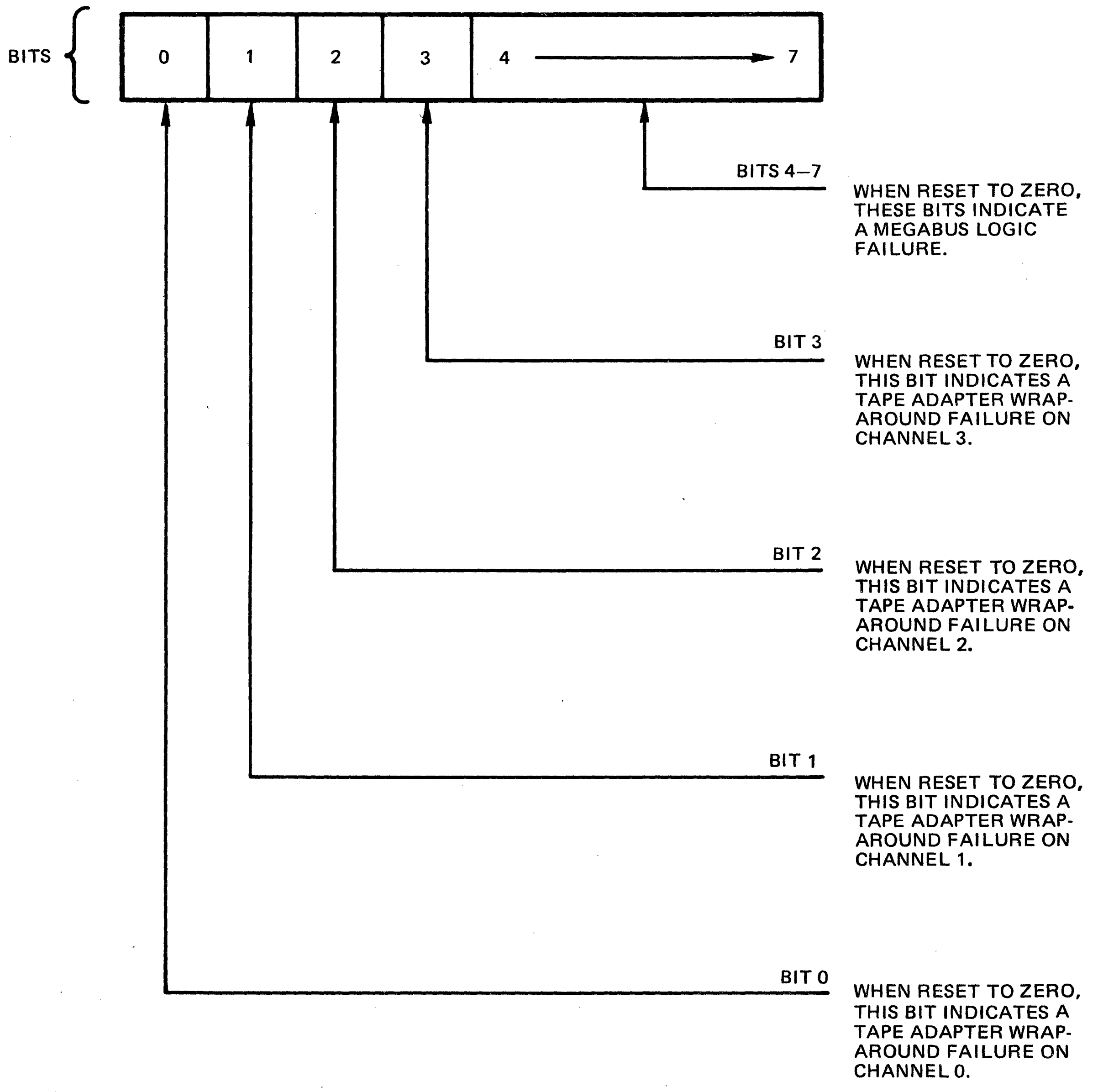

BITS
0
1
2
3
4
7
BITS 4—7
WHEN RESET TO ZERO, THESE BITS INDICATE A MEGABUS LOGIC FAILURE.
BIT 3
WHEN RESET TO ZERO, THIS BIT INDICATES A TAPE ADAPTER WRAP-AROUND FAILURE ON CHANNEL 3.
BIT 2
WHEN RESET TO ZERO, THIS BIT INDICATES A TAPE ADAPTER WRAP-AROUND FAILURE ON CHANNEL 2.
BIT 1
WHEN RESET TO ZERO, THIS BIT INDICATES A TAPE ADAPTER WRAP-AROUND FAILURE ON CHANNEL 1.
BIT 0
WHEN RESET TO ZERO, THIS BIT INDICATES A TAPE ADAPTER WRAP-AROUND FAILURE ON CHANNEL 0.

Figure 4-8 QLTI Byte Bit Structure

As the final step of the initialization (QLT), the QLTI location of SPM is examined. If a hexadecimal FF is present, which was preset at the completion of the basic MTC internal QLT (ALU, BDR, SPM test, etc.), the rear-edge LED is extinguished. Any value other than a hexidecimal FF in the QLTI location indicates a basic MTC internal QLT, a QLT-BUS, or a tape adapter wraparound failure, and the rear-edge LED remains illuminated.

4.3.2 Setup Routine

At the completion of the QLT, the Setup routine (SETUP) is entered. The first portion of the Setup routine initializes scratch pad memory and is utilized by the QLT as a subroutine for zeroing SPM.

The subsequent segment of the Setup routine enables each adapter independently and branches to the Point routine. Upon execution of the Point routine, the Device Support routine, and the Interrupt routine, the Setup routine is re-entered and enables the next sequential adapter. After each adapter has been enabled and set up, the Setup routine branches to the Wait routine.

4.3.3 Wait Routine

The Wait routine (WAIT) tests the channel request priority encoder and the resume interrupt flip-flop to determine what, if any, firmware action is required.

If a channel request is active, the Wait routine loads the scratch pad memory index register with the number of the requesting channel with the highest priority. This enables the MTC logic and the device adapter associated with that channel. After the channel number is loaded, the Wait routine branches to the Bus Request routine if a Megabus transfer to the MTC has occurred. If no Megabus transfer has occurred, the Wait routine returns to the Device Support routine provided the adapter is ready.

If no channel requests are active, the Wait routine tests the resume interrupt flip-flop and branches to the Resume Interrupt routine should the flip-flop be set.

The Wait routine delays until a channel request occurs or the resume interrupt flip-flop sets and performs the prioritizing of the execution of channel requests.

4.3.4 Bus Request Routine

The Wait routine loads the requesting channel number and branches to the Bus Request routine upon detection of a Megabus transfer to the MTC. Since the DMA routines, Interrupt routine, and Resume Interrupt routine complete all Megabus transfers which they initiate, the Megabus transfer causing the request is unsolicited. This means that the Megbus cycle was initiated by the central processor or by another controller.

Should no response be required, the pertinent information from the bus data register data and address segments is stored in the scratch pad memory. This is acomplished by using the function code as the low-order 6 bits of the SPM address. The data is then decoded, if appropriate, causing a branch to the Setup routine, Quality Logic Test, Device Support routine, or back to the Wait routine, depending on the function code type and the data field contents.

If a response is required, the scratch pad memory is accessed using the function code and index register as an address. Data from the scratch pad is loaded in the bus data register, and the response cycle is completed prior to branching back to the Wait routine.

4.3.5 Interrupt Routine

Device Support routines branch to the Interrupt routine whenever a potential interrupt condition is detected or when initialization occurs. If initialization has occurred, the Interrupt routine branches to the QLT-BUS subroutine of the Quality Logic Test to verify some of the MTC bus logic.

If initialization has not occurred, the Interrupt routine generates an interrupt provided the interrupt level is not zero. Data to be utilized during the Megabus cycle is stored in SPM for use by the Resume Interrupt routine, and the interrupt pending flip-flop is set if the interrupt is NAKed. The Wait routine is branched back to at the completion of the Interrupt routine.

4.3.6 Resume Interrupt Routine

The Wait routine branches to the Resume Interrupt routine provided there are no channel requests pending and the resume interrupt flip-flop is set. Upon completion of the Resume Interrupt routine, firmware branches back to the Wait routine.

The Resume Interrupt routine enables each adapter sequentially and retransmits interrupts that have been previously NAKed. No activity occurs if a channel does not have an interrupt pending, as indicated by the interrupt pending flag in the channel monitor byte.

The Resume Interrupt routine sets the channel ready flip-flop and resets the interrupt pending flag if the retransmitted interrupt is ACKed. The interrupt pending flag remains set and the channel remains busy if the retransmitted interrupt is not ACKed. All pending interrupts are retried regardless of the central processor response to earlier interrupts.

4.3.7 Point Routine

The Point routine has two entry points; one is utilized by the Setup routine and the other by the Wait routine and Bus Request routine. The entry point used by the Setup routine sets the initialize flag in the channel monitor byte and then goes to the second entry point.

The second entry point stores the device identification (ID) code in the scratch pad memory and then uses the ID code to determine the type of device connected. This is accomplished by the MTC polling each channel. This results in the adapter transmitting its device ID code to the MTC. The ID code is compared for validity with a MTC-stored ID code. If the device type is supported, the startup function code associated with the device is stored, the SRAR is loaded with the address of the applicable device support routine, and the Channel Ready flip-flop is set. The Point routine then returns to the starting address it has loaded into the SRAR.

If a device is not supported, the Point routine loads the selected SRAR with the starting address of the Interrupt routine and returns to the Interrupt routine.

4.3.8 DMA Out Routine

The DMA Out (DMAOT) routine is used by the device support routines to transfer data from the main memory to the MTC. In a device output operation, the DMA Out routine initiates a Megabus cycle to read the data from the main memory.

The DMA Out routine computes address and range and, when applicable, signifies the end-of-range to the device support routine. This is accomplished by setting the end-of-range flag in the channel monitor byte located in the SPM. Upon completion of the DMA Out routine, a return to the device support routine is achieved by using the indexed SRAR as an address.

4.3.9 DMA In Routine

The DMA In (DMAIN) routine is used by the device support routines to transfer data from the MTC to main memory. When writing data in the main memory, the DMA In routine initiates a Megabus cycle to accomplish the device input operation.

The DMA In routine updates address and range and, when applicable, signifies the end-of-range to the device support routine. This is indicated by the setting of the end-of-range flag in the channel monitor byte located in SPM. Upon completion of the DMA In routine, a return to the device support routine is achieved by using the indexed SRAR as an address.

PLEASE FOLD AND TAPE –
NOTE: U. S. Postal Service will not deliver stapled forms

M&TO HARDWARE PUBLICATIONS
USER COMMENTS FORM

DOCUMENT TITLE: ______________________________

PART NO.: ______________

ORDER NO.: ______________

ERRORS:

HOW DO YOU USE THIS DOCUMENT?

THEORY ______ ☐

MAINTENANCE ____ ☐

TROUBLESHOOTING ☐

OTHER: ______________________________

DOES THIS MANUAL SATISFY YOUR REQUIREMENTS?

YES ☐ NO ☐

IF NOT, PLEASE EXPLAIN ______________________________

FROM: NAME ______________________ DATE __________

TITLE ______________________________

COMPANY ______________________________

ADDRESS ______________________________

FIRST CLASS
Permit No. 39531
Waltham, Ma.

Business Reply Mail NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

HONEYWELL INFORMATION SYSTEMS INC.
200 SMITH STREET
WALTHAM, MA. 02154

MAIL STATION 872A
HARDWARE PUBLICATIONS, BILLERICA

Honeywell

FM88, Rev. 1