Information Systems

Processor Equipment

C66-91, OKL BGE 2

No. 43A177875

TITLE: ENGINÉERING PRODUCT SPECIFICATION - 1
Microprogrammed Peripheral Controller

Total Pages 112 Cont. on Page 2 Page 1

REVISION RECORD

REVISION LETTER	DATE	PAGES AFFECTED	APPROVALS	AUTHORITY
A ISSUED	AUG 17 1970	1, 2, 2.1-2.4, 3-108F	-	

GENERAL ELECTRIC

COMPANY CONFIDENTIAL INFORMATION
(ISG - Class 3 Restricted)

This document shall not be reproduced in whole or in part except as may be necessary for the conduct of the internal business of the Information Systems Group and shall not be disclosed outside of the Company without Department General Manager approval with legal concurrence.

TABLE OF CONTENTS

1. GENERAL DESCRIPTION

This specification defines the Basic Microprogrammed Peripheral Controller (MPC) including its performance, functional capabilities, interface requirements and general design requirements.

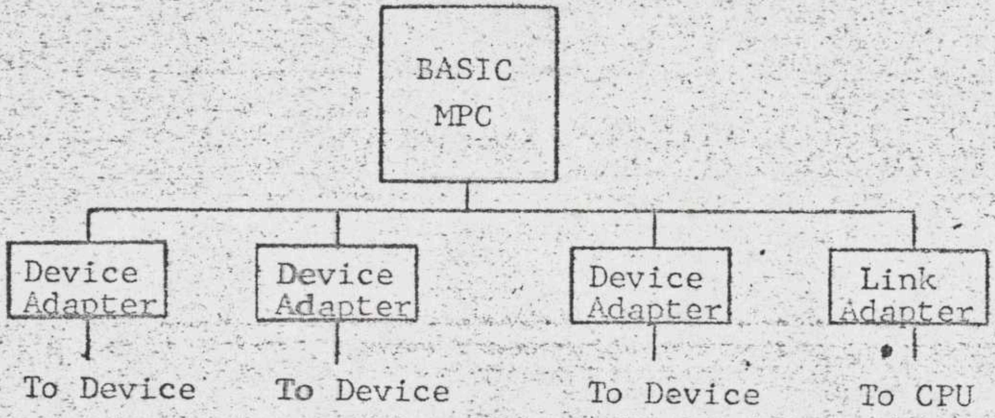
The specification is written in response to Functional Specification ______________.

The following documents support this specification:

- MPC Device Adapter Interface Specification 43A177876
- MPC Microprogramming Manual
- Common Design Requirements 43A177851
- Basic MPC Interior Decor Specification 43A177877
- APL Line System Specification, dated March 2, 1970

1.1 MPC CONCEPT

The Basic MPC is the basic control element used in peripheral controller configurations:



As shown above, a peripheral controller configuration consists of the Basic MPC and one or more Device Adapters and CPU Link Adapters.

The MPC is a general purpose, register-to-register microinstruction processor. Its basic elements are a control store, an optional main memory and a processing structure. By means of microinstructions which reside in the control store, the MPC can be microprogrammed to manipulate data between registers within the MPC, between registers and local MPC main memory, and between registers and Device/Link Adapters.

The Device Adapter consists of the specialized logic and circuitry required for the control of a specific peripheral device(s).

The CPU Link Adapter is a special form of Device Adapter, and consists of the required logic and interface circuits to connect to a central processor I/O channel.

Device Adapters and Link Adapters are not part of the Basic MPC, and are not covered by this EPS.

In order to implement a particular controller configuration, the appropriate Device Adapters, Link Adapters and their associated control microprograms are combined with the Basic MPC. In this way the Basic MPC can serve as the basic control element across a broad range of peripheral types and configurations. Controllers dedicated to a single device type, or to mixed device types can be created by the combination of appropriate Device Adapters and microprograms, with the Basic MPC.

1.1.1 Definition of Terms

The following terms are used in connection with the MPC specification and application:

- Control Store — The storage medium which is used to contain the microinstruction routines in the Basic MPC. May or may not be implemented in read/write technology. However, the MPC accesses control store on a read-only basis.

- Device Adapter

  Refers to the electronic module which interfaces a device and device-oriented-electronics to the Basic MPC through the Device Adapter Interface.

- Device Adapter Interface

  The standard I/O interface by which the Basic MPC interfaces with Device/Link Adapters.

- Interior Decor

  Refers to standard software instruction set specifically intended for the MPC in those configurations requiring software capability. The software instruction set is implemented by an Interior Decor microprogram.

- ITR

  Isolation Test Routines. Refers to special microprograms designed to exercise both the Basic MPC as well as connected Device/Link Adapters for the purpose of diagnosing the logic hardware.

- Link Adapter

  Refers to the electronic module which interfaces the Basic MPC to an external computer interface. (Through the Device Adapter Interface.)

- LSDAI — Low-Speed Device Adapter Interface. Refers to a standard interface intended for low-speed device control (generally 100 kilobytes or less). The Basic MPC interfaces to the LSDAI by means of the Multiplexor Device Adapter. The LSDAI is generally used for connection of the MPC to remote, low-speed free-standing device adapters (card readers, printers, etc.).

- Microinstruction — In the context of the MPC, a microinstruction is a 16-bit field which can be encoded to call out a wide range of register-to-register type operations to be executed by the MPC. Sequences of microinstructions are contained in control store, to form the microprograms for device control, etc.

- MPC — Microprogrammed Peripheral Controller. When referred to as the "Basic MPC", refers to the microprogrammed processor itself, without Device Adapter or Link Adapters.

  When referred to as a specific configuration type (such as Unit Record MPC, or Magnetic Tape MPC), the complete controller configuration, including Basic MPC and Device/Link Adapters is implied.

• Multiplexor Device Adapter	Special Device Adapter which expands one DA Port on the Basic MPC to some number "n" low-speed interface ports. Used on Unit Record MPC and others.
• Unit Record	That MPC configuration dedicated to controlling "low" speed devices (generally 100 kilobytes or less).

1.2 PERFORMANCE CHARACTERISTICS

1.2.1 Microinstruction Execution

Provision will be made in the MPC to allow microinstructions to be executed out of either control store or main memory.

When executed out of control store, the design goal for microinstruction execution time shall be 250 nanoseconds, and must not exceed 300 nanoseconds.

When executed out of main memory, the design goal shall be to execute microinstructions at a rate equal to main memory cycle time. However, slower execution time out of main memory may be acceptable if design economies so dictate.

1.2.2 Concurrent Branch Execution

The MPC will be capable of executing a Branch Type microinstruction concurrent with the execution of any other type microinstruction.

As defined elsewhere in this specification, microinstructions are retrieved from control store two-at-a-time, as "even/odd" pairs. The even microinstruction is executed first, followed by the odd microinstruction execution. If the odd microinstruction is a Branch Type, it will be executed concurrent with the execution of the even microinstruction. This results in a net execution time of "0" nanoseconds for Branch microinstructions.

1.2.3 Connectability

The MPC will provide interfaces for four Device Adapter ports. The configuration of adapter types connected to an MPC is completely flexible and a function of the requirements of the specific controller configuration.

1.2.4 Device Control Capabilities

The MPC must be able to control peripheral devices ranging in speed from slow-speed communication lines to high-speed random access mass storage devices. In addition, the MPC must be able to accommodate a variety of configurations of peripheral devices.

1.2.4.1 Dedicated High-Speed Device Control

The MPC will sustain a maximum burst transfer rate of "N" megabytes from a single, high-speed device where "N" is defined as follows:

1.6 megabytes minimum

2.0 megabytes target

In this mode data is transferred two bytes at a time, under microprogram control, between a Link Adapter and a Device Adapter, with each two-byte transfer passing through an MPC hardware accumulator.

1.2.4.2 Two Simultaneous High-Speed Data Transfers

In this mode the MPC will sustain simultaneous high-speed data transfers from two devices. The maximum data rates which can be controlled in this mode is a function of the microprogram control defined for a particular application. However, as a target two simultaneous 832 kilobyte transfers should be attained, utilizing tailored microprogram, and two-byte data transfers between MPC, Device Adapter and Link Adapter. The minimum acceptable performance is two simultaneous 624 kilobyte transfers.

1.2.4.3 Multiplexed Control of Slow-Speed Devices

In this mode of operation the MPC will simultaneously control a number of slow-speed devices, each device being serviced on an interrupt basis. The number of devices which can be accommodated simultaneously is a function of the speed and control requirements of each device type.

In this configuration main memory will be used for the storage of both control words and data as required.

1.2.4.4 High-Speed Device Control Concurrent with Multiplex Control

In this mode of operation the MPC will simultaneously sustain a single high-speed data transfer while maintaining the multiplexed control of a number of slow-speed devices. The number and speed of devices which can be simultaneously accommodated in this configuration is dependent on the individual device characteristics, and complexity of control.

In this configuration main memory will be used for the storage of both control words and data as required.

1.2.5 Typical Application Configurations

The following paragraphs illustrate some typical controller configurations utilizing the MPC. These configurations are shown only as an example of the types of applications for which the MPC is intended.

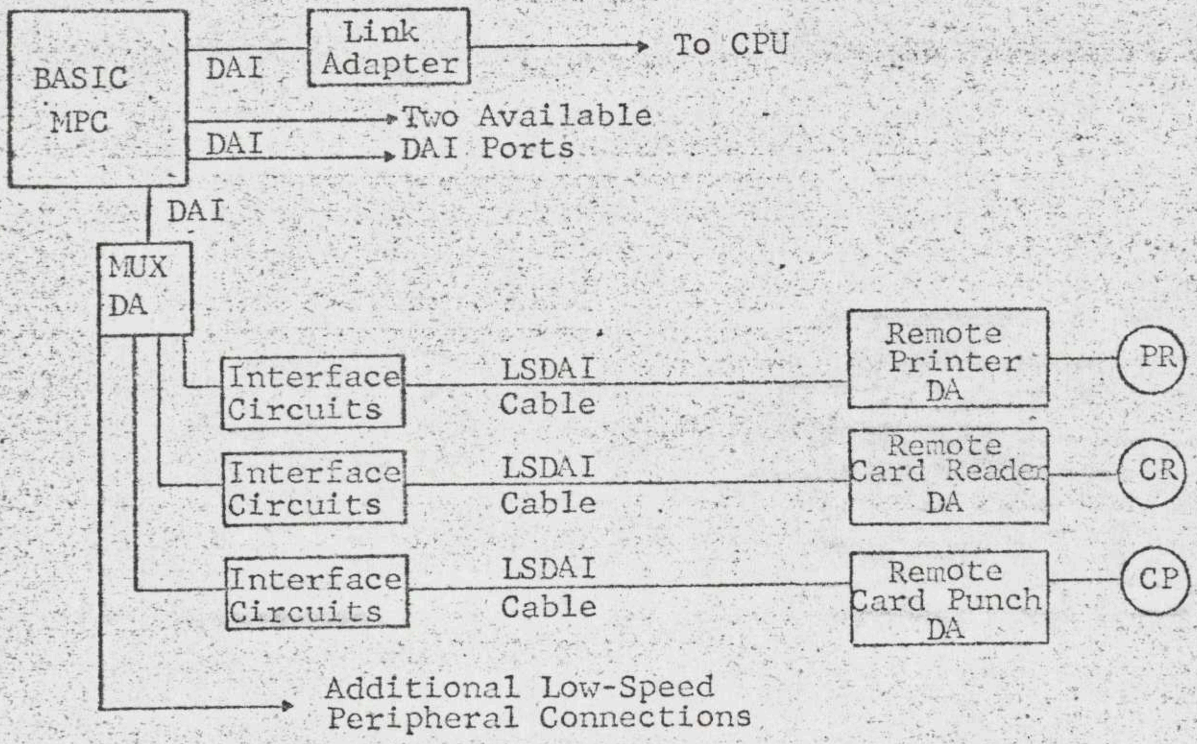
All MPC oriented controller configurations will be covered by freestanding specifications.

In all cases actual MPC configurations must be analyzed with respect to loading and interference factors as a function of the devices to be controlled and the microprogrammed functions required to control each Device Adapter.

1.2.5.1 Unit Record Configuration

In this configuration the MPC is utilized to control, on a multiplexed basis, a number of slow-speed devices. In general, this configuration includes, but is not limited to, card readers, punches, printers, communication lines, document handlers, etc.

A typical block diagram of a Unit Record Configuration is shown below:



MUX DA	-	Multiplexor Device Adapter
PR	-	Printer Mechanism
CR	-	Card Reader Mechanism
CP	-	Card Punch Mechanism
DAI	-	Device Adapter Interface
LSDAI	-	Low-Speed Device Adapter Interface

As shown in the block diagram, the low-speed devices are connected to the MPC through a Multiplexor Device Adapter. This Device Adapter provides the capability of controlling multiple remote (from the MPC cabinet) Device Adapters, each connected through the standard Low-Speed Device Adapter Interface (LSDAI).

The Link Adapter provides the interface between the MPC and the central processor. The Link Adapter interfaces to the MPC through the Device Adapter Interface, and can be designed for the Common Peripheral Interface, the APL Processor-Subsystem-Interface (PSI), or any other computer interface required.

In the block diagram shown, two DAI ports are open, for connection of Communications Device Adapters, or any other Device Adapter required for the particular configuration.

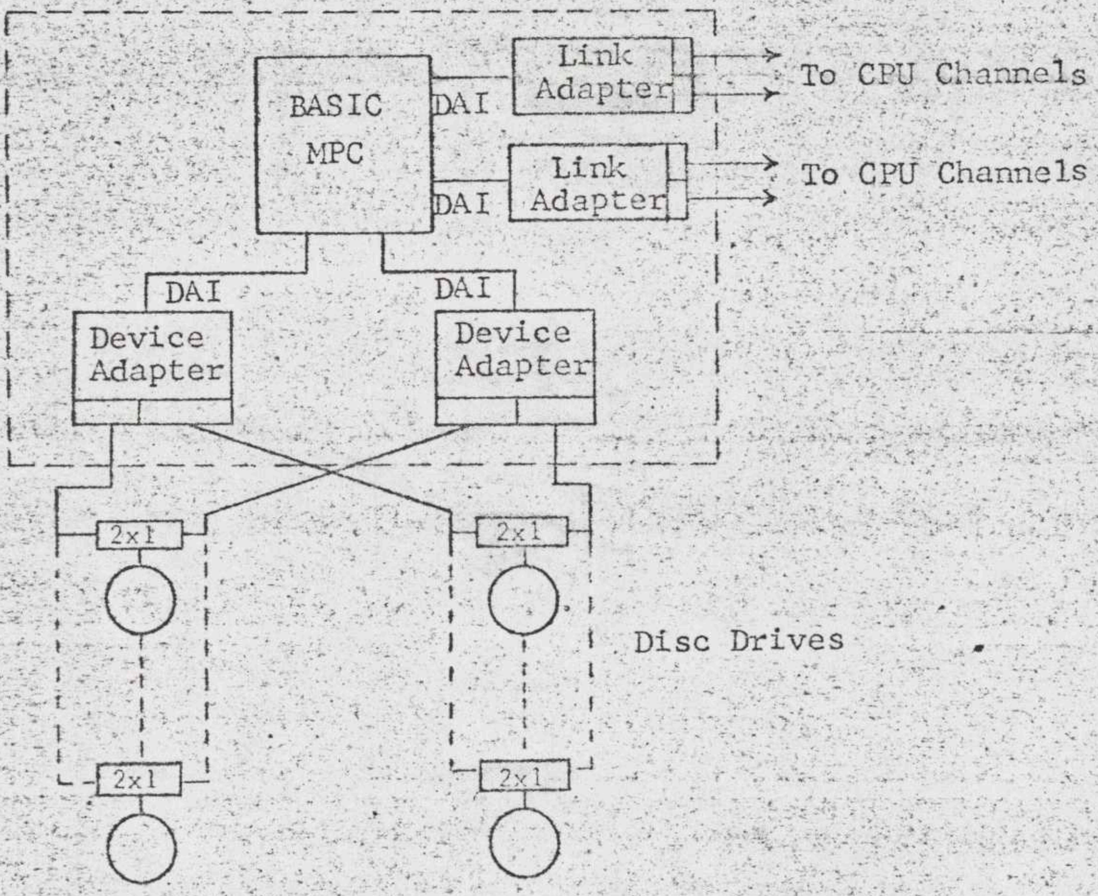
1.2.5.2 High-Speed Disc Configuration

In this configuration the MPC is used to control one or more Disc Device Adapters, connected to one or more strings of Disc Devices. A typical block diagram for such a configuration is shown on the following page.

As shown in the diagram, next page, the Disc Device Adapter is connected to the MPC through a Device Adapter Interface port. The MPC and the Device Adapters will be packaged in the same cabinet, and will share a common power supply.

The Device Adapters interface to strings of devices.

The MPC can control two simultaneous data transfers of at least 624 kilobytes each, where each transfer takes place between a unique Device Adapter and Link Adapter. (Actual transfer limit is a function of final MPC performance characteristics). At higher transfer rates, only one data transfer can be controlled at a time.

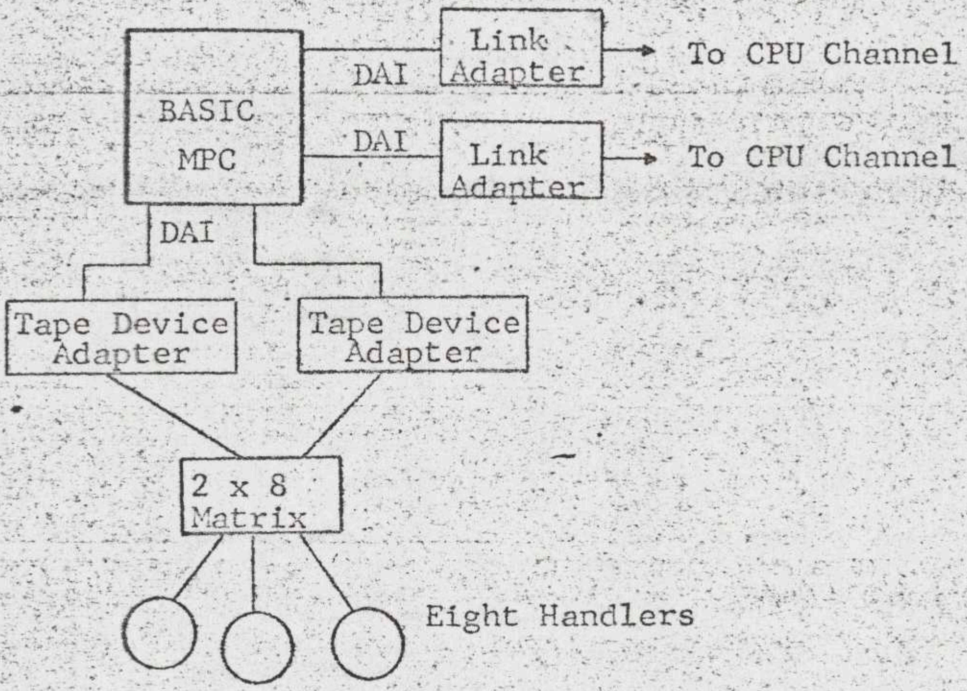


The Link Adapter provides the connection between the MPC and the central processor. The diagram shows the Link Adapter as having two ports, to two independent CPU channels. The Link Adapter can be designed to have only one port.

The Device Adapter and Link Adapter provide the basic functions required for configuring MPC disc subsystems to service a variety of application requirements.

1.2.5.3 Magnetic Tape Configuration

In this configuration the MPC is used to control one or more Magnetic Tape Device Adapters. The block diagram for such a configuration is as follows:



The MPC and Tape Device Adapters will be packaged in the same cabinet, and will share a common power supply. The Matrix will either be in the Device Adapter cabinet, or packaged with the handlers.

1.3 REQUIRED SUPPORT

There are many supporting packages, both software and specifications, that must be included in the overall development process of MPC oriented controller subsystems. These are described in the following paragraphs.

1.3.1 Software Packages

The following software packages must be provided for utilization of the MPC.

1.3.1.1 Microprogram Assembler

An assembly program must be generated which will take microcode statements and assemble them into the equivalent binary microinstructions.

Such an assembler must provide program and error printouts for microprogrammers. It must also provide output required to generate the physical microprograms.

1.3.1.2 Microprogram Simulator

The microprogram simulator is highly desirable to allow simulation of microprograms.

1.3.1.3 Interior Decor Assembly Program

As described in 1.3.3.1 below, an "Interior Decor" software instruction set may be required for some MPC applications.

A corresponding software assembly program will be required to assemble MPC software programs.

1.3.2 Supporting Specifications for MPC Configurations

Since the MPC is only one element in any complete controller configuration, the other elements of the controller configuration must be covered by individual specification.

1.3.2.1 Personalized Controller Specifications

All controller configurations utilizing the Basic MPC for control of specific peripherals will be covered by freestanding Engineering Performance Specifications. These EPS's will define the functions to be performed by the combination of device, Device Adapter and associated MPC microprograms.

1.3.2.2 Link Adapter Specifications

An EPS must be written for each unique Link Adapter and its controlling microprogram. It is possible that in some controller configurations, the Link Adapter microprogram may be unique, and written as part of another Device Adapter microprogram.

1.3.2.3 Device Adapter ITR Specifications

An Isolation Test Routine specification must be written for each unique Device Adapter. This specification will define the functions to be performed by the DA-ITR microprogram which is used to diagnose the DA.

1.3.3 Common Firmware Packages for MPC

There are several common firmware packages which will have utility in various MPC configurations.

1.3.3.1 Common MPC Interior Decor

It is anticipated that for some MPC controller configurations a software level will be desirable in which sequential read/write memory words are pulled from main memory and interpreted as software instructions by a fixed set of Interior Decor microprograms.

An Interior Decor specification must be provided which defines this standard software personality, and the associated microprograms.

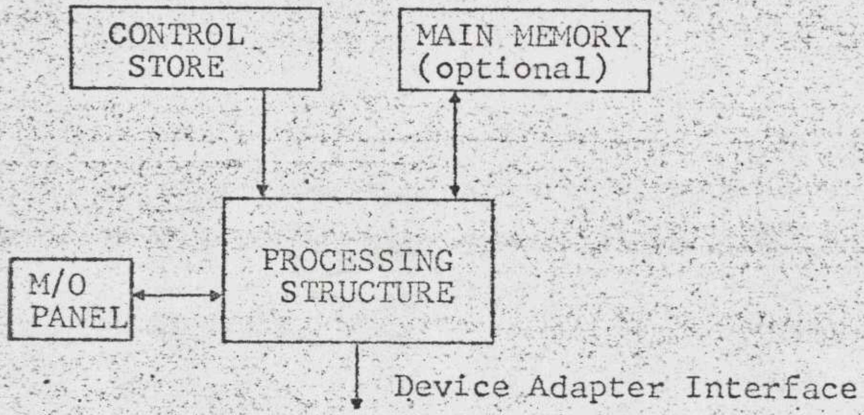
1.3.3.2 Common MPC Isolation Test Routine

The maintainability philosophy for the MPC (as defined in paragraph 6) requires the execution of Isolation Test Routine microprograms.

The Isolation Test Routine microprograms must be defined, specified and generated. The microprograms must be common across all MPC configurations.

2. FUNCTIONAL DESCRIPTION

A block diagram of the major components of the MPC is shown as follows:



The processing structure and the optional main memory provide the general purpose hardware. The control store provides the personalized control for any given application. A description of these major components is contained in the following paragraphs.

2.1 PROCESSING STRUCTURE

The processing structure provides the control, register storage and logical/arithmetic capability required to access, decode and execute micro-instructions which reside in either control store or in main memory.

In order to facilitate running the various simultaneous device configurations described in paragraph 1.2.3, the processing structure will contain the registers and control required to sustain two, independent, concurrent micro-processes. This capability will allow switching from one process to the other without requiring a safestore of all working and control registers.

The basic elements of the processing structure, and their relationship to main memory and control store, are shown in the following diagram. A description of these elements is given in subsequent paragraphs.

2.1.1 Registers

The registers defined in this paragraph are those registers which are either directly referenced by microinstructions, or required for specific functional reasons. There will be other registers which are a function of the hardware implementation, and will be defined during the design phase.

2.1.1.1 Working Registers

There are two independent sets of working registers, in order that two independent microprocesses can coexist without requiring safe storing of registers. These registers are shown as crosshatched in the preceding MPC detailed block diagram.

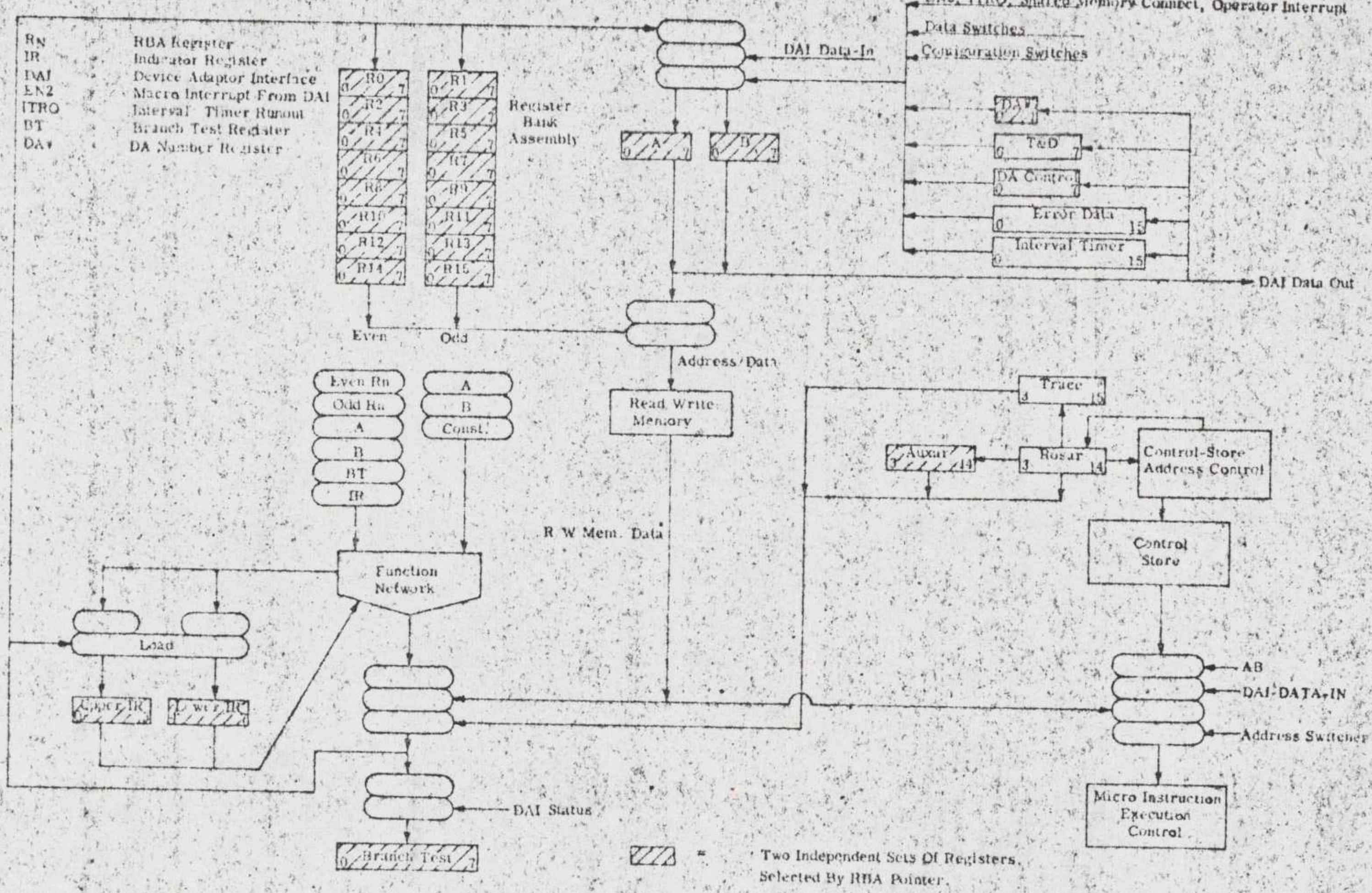
All microinstructions are formatted as though there were only one set of working registers. The physical set of registers accessed during the execution of a microinstruction is determined by the state of an "RBA Pointer," as defined in paragraph 2.1.3.

The functions which can be performed on these registers are defined in Appendix A, "Microinstruction Repertoire."

1. Register Bank -- This is a group of 16 general purpose working registers, each one-byte wide (8-bits plus parity), used for data storage, accumulating of arithmetic and logical operation results, etc.

   They are accessible under microinstruction control on either a single or two-byte basis. When accessed on a two-byte basis, the registers must be considered as even/odd pairs (R0/R1; R2/R3, etc.).

   For purposes of definition and discussion throughout this specification, each of the two independent groups of 16 registers is referred to as either the "Upper RBA Half" or the "Lower RBA Half." Within an RBA Half, individual registers are referred to as R0-R15.


RN = RBA Register
IR = Indicator Register
DAI = Device Adaptor Interface
EN2 = Macro Interrupt From DAI
ITRO = Interval Timer Runout
BT = Branch Test Register
DA# = DA Number Register
R0
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
0
7
Register Bank Assembly
Even
Odd
DAI Data-In
EN2, ITRO, Shared Memory Connect, Operator Interrupt
Data Switches
Configuration Switches
A
B
DA#
T&D
DA Control
Error Data
Interval Timer
15
DAI Data Out
Address/Data
Read Write Memory
R W Mem. Data
Even Rn
Odd Rn
A
B
BT
IR
A
B
Const.
Function Network
Load
Upper IR
Lower IR
DAI Status
Branch Test
Trace
3
15
Auxar
14
Rosar
Control-Store Address Control
Control Store
AB
DAI-DATA-IN
Address Switches
Micro Instruction Execution Control
= Two Independent Sets Of Registers, Selected By RBA Pointer.

Cont. on Sh. 20

Associated with each RBA Half is a unique set of supporting registers, as defined in paragraphs 2.1.1.1(2) through 2.1.1.1(6) below.

2. Accumulator A/Accumulator B -- These two registers are general purpose accumulators, each one byte wide (8-bits plus parity), used for data storage, accumulation of arithmetic and logical operation results, etc.

   They are accessible, under microinstruction control, individually, or combined as a two byte register. When accessed as a two byte register Accumulator A is always the most significant.

3. Auxiliary Control Store Address Register (AUXAR) -- This is a 12-bit address register used to safestore the next microinstruction address when:

   a. a hardware interrupt occurs (AUXAR contains address of next microinstruction to be executed upon return).

   b. a store address and branch microinstruction is executed (AUXAR contains address of last even/odd microinstruction pair executed).

   This register is normally referred to as AUXAR_{3-14}. The register always holds an even address.

4. Indicator Registers -- There are two four-bit indicator registers. One is referred to as the Upper Indicator Register; the other is referred to as the Lower Indicator Register.

   Both indicator registers can be used optionally to store indicator results of a microinstruction execution.

   The Upper Indicator Register has the added optional capability of propagating certain indicator conditions over a sequence of microinstruction executions.

The microinstruction format will allow designation of which indicator register is to be used during its execution.

Both registers will maintain the following conditions:

● For Arithmetic Operations:	Most Significant Bit Value
	Overflow
	Zero
	Carry
● For Logical Operations:	Most Significant Bit Value
	All Ones Condition
	Zero Condition
	Odd/Even
● For Shift Operations: (Lower IR Only)	Shifted Out Bit, or
	Bit to be Shifted into Register

5. Device Adapter Number Register -- This is a two-bit register used to define which Device Adapter port is to be accessed by a Device Adapter Interface microinstruction.

   This register can be loaded by microinstruction. It is also loaded automatically with the interrupting Device Adapter number when a Device Adapter Interface interrupt (EN-1) is executed by the Interrupt Mechanism.

6. Branch Test Register -- This is an 8-bit register used for conditional branch testing and vector segment branching.

   The results of a microinstruction execution can be directed into the Branch Test Register (in addition to its normal destination) and tested in subsequent conditional or vector segment branch operations.

Device Adapter Interface status can also be loaded into this register for subsequent branch testing.

2.1.1.2 General Control Registers

The following registers are required for the general control and operation of the MPC.

There is only one set of these registers, used to sustain microinstructions execution out of either set of working registers defined in paragraph 2.1.1.1.

1. Control Store Address Register (ROSAR) -- This is a 12-bit register which serves as the microinstruction address for microinstructions out of Control Store via the read-only Control Store interface.

   This register is normally referred to as $ROSAR_{3-14}$, and always holds an even address.

2. Device Adapter Control Register -- This 8-bit register is used to control the execution of Device Adapter interrupts, and Link Adapter selection during the execution of Device Adapter microinstructions.

   The register can be loaded under microinstruction control.

   The register consists of the following fields:

   - Level Field (Bits 0,1)

     This field contains two bits, bit 0 for Device Adapter port "0," and bit 1 for Device Adapter port "1."

     These bits define the EN-1 interrupt level for Device Adapter ports "0" and "1" as follows:

     Level Bit = set, EN-1 from DA port
     considered to be "high-level."

     Level Bit = reset, EN-1 from DA port
     considered to be "low-level."

Paragraph 2.1.2.2 of this specification defines the interrupt priority structure and execution in the MPC.

- Link Adapter (LA) Field (Bits 2,3)

A Device Adapter Interface microinstruction can specify that the instruction is addressed either to the DA port whose number is contained in the DA Number Register, or to a generic Link Adapter (CPU connection) port.

The 2-bit LA Field of the DA Control Register is used to define which physical DAI port is to be accessed by a DAI microinstruction addressed to a generic Link Adapter port.

The definition is made as a function of the current contents of the DA Number Register, as follows:

DA Number Register points to DA port 0:

LA Field Bit 2 = 0	Link Adapter defined to be DAI port 2.
LA Field Bit 2 = 1	Link Adapter defined to be DAI port 3.

DA Number Register points to DA port 1:

LA Field Bit 3 = 0,	Link Adapter defined to be DAI port 2.
LA Field Bit 3 = 1,	Link Adapter defined to be DAI port 3.

If the DA Number Register points to a DAI port other than "0" or "1,' a DAI microinstruction addressed to the Link Adapter will always access DAI port 3.

(See Appendix A for a description of the DAI microinstruction.)

- EN-1 Mask Field (Bits 4-7)

This field contains one bit for each of the four Device Adapter ports, (bit 4 for DA0, bit 5 for DA1, etc.). When the mask bit is set for a particular Device Adapter port, EN-1 interrupt requests from that DA are masked. The interrupt request will be honored when the mask condition is removed, if still being held up by the Device Adapter.

3. Test and Diagnostic Register -- This register provides the capability of setting up special control modes to allow Isolation Test microprograms to exercise the internal logic of the MPC to the degree required to isolate failures. The register can be read and loaded under microprogram control. It can also be initialized from the Operator/Maintenance Panel, or as the result of an external "Initialize" signal (paragraph 3.1.1), or an initialize microinstruction.

   Bits will be provided in this register for the following functions:

   - Ignore Errors

     When this bit is set the MPC will ignore the occurrence of all errors. An error interrupt will not be generated. Detected errors will be loaded into the Error Data Register, however.

   - Halt On Any Internal Error

     When this bit is set the MPC will halt upon the detection of any internal error.

   - Halt On External Error

     When this bit is set the MPC halts upon the detection of a main memory error or a Device Adapter Interface error. The error may be either parity or timeout.

   - Halt On Control Store Error

     When this bit is set the MPC halts upon the detection of an error involving the access of a microinstruction from control store.

   - Halt On Interval Timer Runout

     When this bit is set, and the Test and Diagnostic Mode bit is set, the MPC will error halt upon the detection of Interval Timer runout.

When this bit is set, and the T&D Mode bit is not set, the MPC will error interrupt upon the detection of Interval Timer not reloaded, as defined in paragraph 2.1.5.

- Test and Diagnostic Mode

  When set, this bit defines the existence of the "T&D Mode," and causes a unique microprogram entry to be made upon the detection of an error or Low-Level EN-1 interrupt.

  These microprogram entry points will be fixed, and will be different from the normal error interrupt entry point. (See paragraph 3.5.)

- Interval Timer Count Mode

  When set, this bit causes the Interval Timer to be incremented once for each microinstruction execution. In this mode the Interval Timer counts the number of microinstructions executed.

  When this bit is reset, the Interval Timer is incremented as a fixed time interval. In this mode the Interval Timer functions as a real time clock (paragraph 2.1.5).

4. Error Data Register -- This 16-bit register provides storage for the indication of any detected error within the MPC.

   At the time an Error Interrupt is generated, this register will define the specific error causing the interrupt.

   The register can be read and reset under microprogram control. It can also be reset from either the Operator Panel, or as the result of an external "Initialize" signal.

Typical of the category of errors detected by this register are:

- Internal parity errors
- External input bus parity errors (DAI or main memory)
- Memory parity errors
- Control store parity errors
- Timeout errors

5. Trace Register -- This 13-bit register is used to hold the control store address of the microinstruction on which an error is detected.

   This register can be read under microinstruction control. It can also be reset as the result of an "Initialize" signal from the Device Adapter Interface, initialize microinstruction, or Operator/ Maintenance Panel.

   This register is normally referred to as $TRACE_{3-15}$.

2.1.2 Interrupt Capability

There are four distinct levels at which the MPC can be operating at any one instant of time. These levels are defined as follows:

Error Interrupt Service (highest priority)
High-Level Interrupt Service
Low-Level Interrupt Serivce
Normal Mode

Each of the first three levels is entered as the result of a corresponding hardware interrupt:

Error Interrupt
High-Level EN-1 Interrupt
Low-Level EN-1 Interrupt

By definition, when the MPC is not in an interrupt service level, it is in the Normal Mode.

Hardware interrupts occur at the microinstruction level. If the interrupt request is detected prior to the initiation of the execution of an "odd" microinstruction, the interrupt will be executed immediately following the completion of the "odd" microinstruction execution.

If the interrupt request is detected prior to the initiation of the execution of an "even" microinstruction, the interrupt will not be executed until the completion of the next "odd" microinstruction. (In the event the odd instruction is a "branch," it is executed concurrently with the even microinstruction, and the interrupt then occurs immediately after the even microinstruction execution.)

This function allows a "program" interrupt capability to be microprogrammed into a microroutine.

The above interrupt capabilities are defined in the following paragraphs.

2.1.2.1 Error Interrupt

This is the highest level interrupt, and occurs upon the detection of any error by the MPC. The Error Interrupt can be inhibited by the T&D Register.

At the time this interrupt is executed the Error Data Register will contain a bit defining the specific error causing the interrupt.

The execution of the interrupt will cause a forced branch to be made to a fixed control store address (paragraph 3.5), where the error servicing microprogram must be located.

At the time the interrupt is executed, the interrupted control store address, representing the next normal microinstruction to be executed, will be saved in AUXAR.

In addition, the address of the microinstruction being executed at the time of error detection will be saved in the TRACE register. (Note that the address saved in AUXAR will always differ from the address saved in TRACE, since AUXAR is always incremented to point to the next even/odd pair of microinstructions.)

During the time the Error Interrupt service is in progress, no other interrupt requests will be executed.

The Error Interrupt service level is reset by the execution of the reset version of the Branch microinstruction (see Appendix).

2.1.2.2 High-Level EN-1 Interrupt/Low-Level EN-1 Interrupt

These two types of interrupts occur as the result of a signal on the EN-1 line from a Device Adapter port.

- Level and Priority

  For Device Adapter ports 0 and 1, the level of the interrupt generated by the EN-1 line is determined by the state of the "Level Field" of the DA Control Register (paragraph 2.1.1.2(2)).

  For Device Adapter ports 2 and 3, the EN-1 line is always interpreted as a Low-Level EN-1 interrupt.

The priority between DA ports governing which existing interrupt request is allowed is determined by the following:

DA Port 0 - High or Low Level (Highest Priority)
DA Port 1 - High or Low Level
DA Port 2 - Low Level
DA Port 3 - Low Level (Lowest Priority)

As can be seen from the above, any "interrupt request" on DA port 0 has higher priority than any interrupt request on DA ports 1, 2 and 3.

- Interrupt Execution

At the time a High-Level EN-1 interrupt is executed, a forced branch is made to the control store location specified by the contents of the Auxiliary Control Store Address Register (AUXAR) plus two, and the contents of AUXAR replaced by the interrupted control store address, (address of the next normal microinstruction to be executed upon return from interrupt service).

At the time a Low-Level EN-1 interrupt is executed, a forced branch is made to a fixed control store address (paragraph 3.5), where the Low-Level dispatching microprogram will be located. The interrupted control store address is stored in AUXAR.

Upon executing an EN-1 interrupt, the MPC will set an "Interrupt In Progress" mode defining that (a) an interrupt is in progress, and (b) the level (High or Low) of the interrupt. This mode will remain on until reset by the execution of the reset version of the Absolute Branch microinstruction.

During the time the "Interrupt In Progress" mode bit is on, all interrupt requests which are of equal or lower priority than the level in progress will be held off. However, an interrupt request of higher priority will be granted, and will interrupt the interrupt service in progress.

This means that if a High-Level interrupt service is in progress, it cannot be interrupted by any other EN-1 interrupt request. However, a Low-Level interrupt service can be interrupted by a High-Level request.

It should be noted that an EN-1 interrupt request is not automatically reset when the interrupt request is honored. The EN-1 interrupt request is held up by the requesting Device Adapter until a DAI microinstruction is executed by the interrupt service microprogram specifically causing the Device Adapter to reset the interrupt request line. As long as the interrupt request line remains set, all interrupt requests on lower-priority DA ports will be inhibited.

2.1.2.3 Macro Interrupt

As stated above, the Macro Interrupt is not a hardware interrupt, but is actually a set of conditions which can be tested for by microinstruction, allowing appropriate action to be taken by the microprogram.

A four-bit Interrupt Mechanism status bus is provided for testing by the "Vector Segment Branch" microinstruction. Conditions are defined on this bus as follows:

Bit 0 - Manual Mode

This condition is set under control of an Operator Panel Switch (paragraph 4.2). The condition remains set until manually reset.

By the setting and resetting of this mode, operator control can be exercised over the execution of a microprogram.

Bit 1 - Operator Interrupt

This condition arises as the result of the actuation of the Operator Interrupt switch on the Operator Panel (paragraph 4.2).

The condition is reset upon execution of the version of the "Store Interrupt Mechanism Register" microinstruction which saves the state of this bit in Accumulator B.

Bit 2 - Interval Timer Runout

Bit 3 - Macro Interrupt Condition

This condition will be set whenever any one or more of the following indications is present:

- Shared Memory Connect

  This condition indicates an interrupt has been set from an MPC sharing the same main memory. (Results from "Start Shared Memory Cycle" microinstruction.)

  The condition is reset by the execution of the reset version of the "Start Shared Memory Cycle" microinstruction by either "Memory Sharing" MPC.

- Signal on EN-2 line from any of the four DA ports.

  Execution of the "Store Interrupt Mechanism Register" microinstruction allows the transfer of all the above conditions to Accumulator B for subsequent testing.

2.1.2.4 Special Interrupt Control

The following interrupt functions can be set or reset by execution of the appropriate version of the "Change Interrupt Mechanism Conditions" microinstruction.

- Inhibit All EN-1 Interrupts

  When this function is set, all Device Adapter EN-1 interrupt requests are masked. Normal honoring of interrupt requests resumes when the inhibit is reset.

- Simulate Device Adapter Interrupt

  Setting of this function causes an interrupt request to be set from all four DA ports. The interrupt requests will remain until this function is reset.

- Inhibit Macro Interrupts

  This inhibit is set by the execution of the appropriate version of the "Change Interrupt Mechanism Conditions" microinstruction.

  When set, this condition inhibites detection of the following Macro Interrupt conditions:

  Shared Memory Connect
  EN-2 from Device Adapters
  Interval Timer Overflow

- Simulate Error Interrupt

  This function is set by special microinstruction, and causes the execution of a pseudo error interrupt. At the time the interrupt is executed the Error Data Register will contain a bit defining the error as simulated.

2.1.3 Register Bank Assignment

As defined in paragraph 2.1, there are two independent sets of working registers, referred to as the "Upper RBA Half" and the "Lower RBA Half."

All microinstructions are formatted as though there were only one set of working registers. The register bank actually accessed by any microinstruction execution is determined by the state of an Active RBA Pointer. This Active RBA Pointer is maintained by the Interrupt Mechanism as a function of the current level at which the MPC is running:

- Error Interrupt Service
- High-Level Interrupt Service
- Low-Level Interrupt Service
- Normal Mode

The following paragraphs define the control of the Active RBA Pointer for each of these four operating levels.

2.1.3.1 Normal Mode

In this state of the machine no interrupt is in the process of being serviced, and the Active RBA Pointer is controlled by the state of a "Normal RBA Pointer."

The "Normal RBA Pointer" can be set or reset by the appropriate version of the "Change Interrupt Mechanism Condition" microinstruction. When this pointer is set, the Upper RBA Half will be active.

2.1.3.2 Interrupt in Progress

When an "EN-1 Interrupt" is executed, the Active RBA Pointer is automatically set by the Interrupt Mechanism as a function of the following elements:

- "Level Field"of the DA Control Register
- Dual Channel Mode Bit
- Normal RBA Pointer

The "Level Field" of the DA Control Register defines the interrupt level (high or low) at which DA ports 0 and 1 are to operate. (See paragraph 2.1.1.2(2)). DA ports 2 and 3 always operate at the Low-Level.

A Dual Channel Mode Bit, which can be set or reset by the appropriate version of the "Change Interrupt Mechanism Condition" microinstruction, determines which RBA half DA port 1 is to use when it is operating at the "High" interrupt level.

The Normal RBA Pointer is set and reset by the appropriate version of the "Change Interrupt Mechanism Condition" microinstruction, and determines which RBA half is to be used when the MPC is operating in the "normal" mode.

1. Low-Level Interrupt Service -- When a Low-Level Interrupt service is initiated, for any DA port, the Active RBA Pointer will be set to point to the opposite RBA Half from that used by the Normal Mode of operation.

   Stated another way, Low-Level Interrupt service will use the RBA Half opposite from the RBA Half pointed to by the "Normal RBA Pointer."

2. High-Level Interrupt Service, DA Port 0 -- A High-Level Interrupt service initiated from DA port 0 will always be serviced out of the Lower RBA Half.

3. High-Level Interrupt Service, DA Port 1 -- A High-Level Interrupt service initiated from DA port 1 can be serviced optionally out of either the Upper or Lower RBA Half, depending on the state of the "Dual Mode Bit":

Dual Mode Bit Set:	High-Level service out of Upper RBA Half.
Dual Mode Bit Reset:	High-Level service out of Lower RBA Half.

2.1.4 Function Network

The function network is an arithmetic/logical network capable of performing the following operations with two, one-byte operands:

Binary Add
Binary Subtract
AND
OR
Exclusive OR
Negate
Complement
Shift By One Bit
Shift By 8 Bits

The function network is duplicated, and the output of the two networks is compared for equality.

Parity on the function network output is formed by using the result from one network and parity as generated on the results from the other network.

2.1.5 Interval Timer

The Interval Timer provides a real-time clock function. It can be both loaded and read by microinstruction.

When the Interval Timer runs out, it causes the "Interval Timer Runout" Macro Interrupt condition to be set, as described in paragraph 2.1.2.3.

The Interval Timer will be continuously counted at a fixed frequency to be determined by hardware design, but the interval chosen will be between 5 and 20 microseconds.

The maximum set table timeout will be determined during design, but will be between 50 and 500 milliseconds.

Under control of the Test and Diagnostic Register (paragraph 2.1.1.2(3)), an error interrupt will be generated if the Interval Timer is not reloaded within a specific time interval after runout. This time interval will be determined during design, but must be between 50 and 500 milliseconds.

Special T&D modes can be set in which the Interval Timer is made to count microinstruction executions, and to halt on Interval Timer Runout. These modes are described in paragraph 2.1.1.2(3).

2.1.6 Error Detection

The following error detection features will be incorporated into the MPC.

- Byte parity is carried and checked on all data transfers and data storage within the MPC.
- Byte parity is carried on all microinstructions in control store, and is checked as microinstructions are pulled from control store.
- The arithmetic function network is duplicated, and the results of each network are compared for equality. Parity on the result is generated.
- Byte parity is maintained on the Interval Timer.
- Byte parity is carried on all data and control busses to the Device Adapter Interface, and checked on all data and control busses from the Device Adapter Interface.
- Byte parity is carried on all address and data transfers to main memory, and checked on all data transfers from main memory.
- Interval Timer is checked for reloading following runout.
- Detection is made of attempted access to non-existent control store.
- A timeout is made on all asynchronous operations which involve waiting for an external signal, and on all microinstruction executions. This includes waiting for an RPI signal from the Device Adapter Interface during a DAI instruction; and waiting for a response from main memory during a memory microinstruction. The duration of this timeout will be determined during design, but will be no less than 8 microseconds and no more than 20 microseconds.

All error detections result in an Error Interrupt, as defined in paragraph 2.1.2.1. At the time the Error Interrupt is generated the Error Data Register will be set to indicate the specific error causing the interrupt.

2.2 CONTROL STORE

The control store provides storage for microinstructions.

The physical implementation of the control store depends on technology developments, and may change during the life of the MPC product.

Two distinct storage functions are defined for the MPC: main store and control store. The MPC has a <u>separate interface</u> to each of these functions, regardless of whether the storage subsystems are separate, or implemented as one volatile read/write subsystem.

For the purpose of this specification the term control store refers to that storage subsystem or portion of a storage system from which microinstructions are read via the "Read-Only" Memory Interface. (Interface "a" in diagram, paragraph 2.2.3.)

2.2.1 Format

The basic control store word is the microinstruction, which is formatted as 16 data bits plus two parity bits, one per byte, for a total instruction word width of 18 bits.

2.2.2 Size/Modularity

As a design target, the basic control store modularity will be 512 microinstructions, with a maximum control store size of 8K microinstructions.

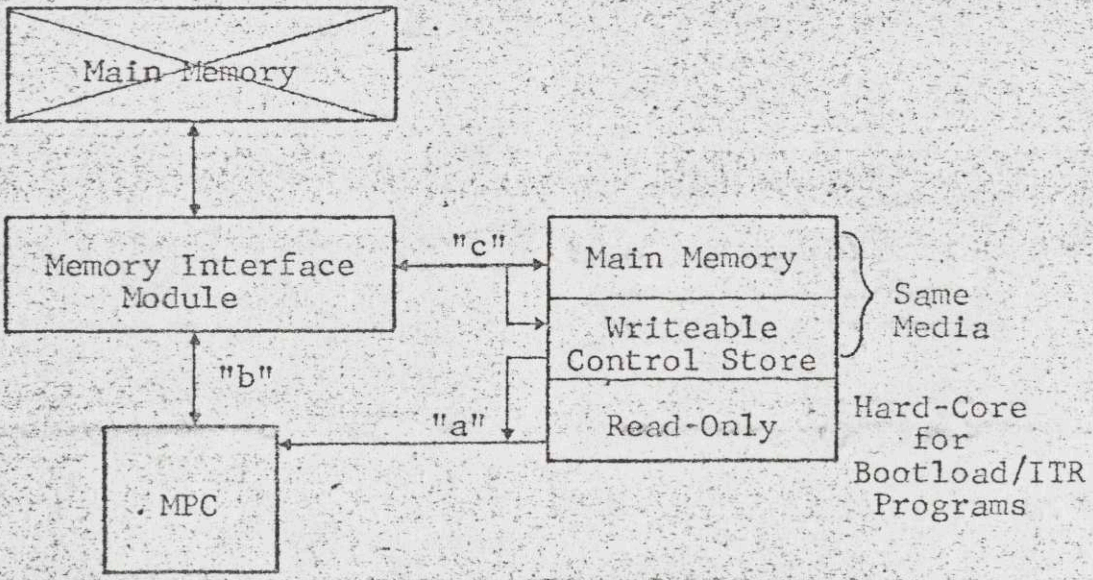
The MPC will be designed to allow modular expansion of control store in the field by a simple board plug-in procedure.

2.2.3 Writeable Control Store Capabilities

The design of the MPC will provide for the utilization of a writeable control store, read only control store and mixtures of the two technologies.

The design of the MPC should allow for MPC configurations which utilize all writeable control store, with no conventional main memory. (This configuration will include a hard core of nonwriteable control store for bootload/ITR purposes.) The microinstruction repertoire will not change in this configuration, and the MPC will still offer the flexibility of control store for microinstruction storage and main storage (but now physically implementated as an extension of the control store) for data storage.

The following diagram illustrates an acceptable implementation of this capability:



2.2.3.1 MPC Configuration Using all Writeable Control Store

In the configuration shown, the main memory is not present, and the main memory function is now controlled by a modified "memory interface module" over interfaces "b" and "c."

Interface "b" is the original main memory interface.

Interface "c" is an interface, controlled by the "memory interface module," which provides access into a second port in the writeable control store.

Main memory microinstructions are executed as before, but now access this second port in the writeable control store, which serves for main memory data storage.

Interface "a" is the standard read-only interface used to pull microinstructions from control store.

The "writeable control store" is written through interface "b" under control of a microprogram which resides in the hard core "read-only" portion of control store.

2.3 MAIN MEMORY

Main memory can be added to the MPC on an optional basis. As a design goal, it should be possible to remove all logic and circuitry pertaining solely to the main memory in MPC configurations not requiring this memory.

The interface to main memory is unique from the interface to the control store used to pull microinstructions.

2.3.1 Memory Modularity

The following numbers are to be interpreted as design targets. They may require change, depending on the economics of implementation.

2.3.1.1 Minimum Memory Size

Minimum memory size (when included) will be 2 kilobytes.

2.3.1.2 Standard Memory Sizes

Optional memory sizes greater than the minimum will be:

4 kilobytes
8 kilobytes
16 kilobytes
24 kilobytes
32 kilobytes
48 kilobytes
64 kilobytes (maximum memory size)

Main memory will be designed to allow modular expansion in the field by a simple board plug-in procedure.

2.3.2 Memory Width

The basic width of a memory word shall be two bytes. Each byte will include a parity bit, making the basic memory word width 18 bits.

2.3.3 Memory Accessibility

Memory will be accessible under microinstruction control on either a word or byte basis. (See definition of memory access microinstructions, Appendix A.)

The basic memory access will be on a word (2 byte) basis. However, it will be possible, under microinstruction control, to access either the high or low order byte of an addressed memory word.

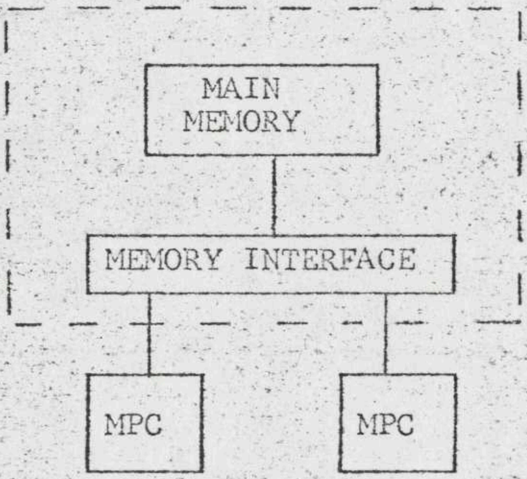
For specific memory accessibility requirements see Appendix A, definition of memory access microinstructions.

2.3.4 Memory Speed

The basic memory cycle time shall be no slower than 900 nanoseconds.

2.3.5 Shared Memory Capability

The MPC will be capable of sharing main memory with a second MPC.



In a shared memory configuration, the maximum memory capacity is 64 kilobytes.

By means of a Configuration Switch (see paragraph 4.2.8) the "Shared Memory" state can be manually defined to exist. The state of the switch is testable by microinstructions.

When accessing main memory, the "Start Read Memory Cycle" and "Start Write Memory Cycle" microinstructions can use the state of this switch in forming the memory address. This allows automatic memory address modification as a function of the main memory configuration.

2.4 MICROINSTRUCTION REPERTOIRE

The microinstruction repertoire provides the capability of manipulating data between registers within the processing structure of the MPC, as well as between the processing structure, main memory, the Device Adapter Interface, and the Operator/Maintenance Panel.

The following gross capabilities are provided by the microinstruction repertoire.

- Arithmetic Operations

  Single byte, binary add, subtract between registers and accumulators.

- Logical Operations

  AND, OR, Exclusive OR between registers and accumulators.

- Immediate Value Operations

  Arithmetic and Logical operations, with immediate value constant contained in microinstruction.

- Shift Operations

  Shift by 1 or 8 bits, accumulators and registers.

- Double Byte Operations

  Sets of 2-byte registers transferred as 16 bit fields.

- Read/Write Memory Operations

- Interrupt Mechanism Operations

  Control Interrupt functions.

- Device Adapter Interface Operations

  Data transfer to and from the DAI.

- Branch Instructions

  Conditional, Segment, Absolute Branch operations within control store.

The microinstruction repertoire is defined in detail in Appendix A.

3. OPERATIONAL DESCRIPTION

The following paragraphs define pertinent operational characteristics of the Basic MPC.

3.1 INITIALIZATION

There are four ways in which the MPC can be set to the Initialize mode:

- Actuation of Initialize Switch on Operator's Panel.
- An Initialize Signal from a Link Adapter.

  This must be a Link Adapter connected to either Device Adapter port 2 or port 3. (See paragraph 4.1 for description.)
- Execution of the "Change Interrupt Mechanism Condition" microinstruction.
- Power Up Sequence.

3.1.1 Initialized State

The Initialization procedure causes the MPC to set itself to a known operating state. This state is defined as follows:

- RBA registers are not initialized. They must be loaded by microinstruction before being read, or an error interrupt may result.
- Following registers are reset to zero, but with good parity. They may be read without causing error interrupt:

  Indicator Registers
  Interval Timer
  "LA" and "Level" Fields only
  Error Data Register
  DA Number Register
  Branch Test Register
  Trace Register
  Accumulators A,B
  AUXAR

- DA Control Register, "Mask Field" set.
- T&D Register is set, with exception of the "Ignore Errors" bit which is reset.
- Active RBA Pointer is reset to point to the Lower RBA Half.
- The "Inhibit All EN-1's" mode is reset.
- ROSAR is reset with good parity if the initialization was executed from the Operator Panel, Link Adapter, or Power Initialization.
- ROSAR is left unchanged if the Initialization was executed via the "Change Interrupt Mechanism Condition" microinstruction.
- A reset signal will be sent to each connected Device Adapter over the "Initialize" line of the Device Adapter Interface.

3.1.2 Next Normal Procedure

When the Initialization function is initiated from the Operator Panel or via a Power-Up sequence, the MPC will be left in the initialized state, and halted.

The next normal sequence is the actuation of the Start Switch on the Operator's Panel. The Control Store Address Register, ROSAR, will have been cleared to zero. Microinstruction execution will therefore, resume from absolute Control Store Location 0.

If the Initialization was executed via the "Change Interrupt Mechanism Condition" microinstruction, the MPC will be initialized, and then will automatically resume microinstruction execution. In this case, microinstruction execution will resume from location 0 of the 256 microinstruction segment which contained the "Initialize" microinstruction.

If the Initialization was executed via a signal from the Link Adapter (see paragraph 4.1.1), the MPC will be initialized, and then will automatically resume microinstruction execution, starting at absolute Control Store Location 0.

3.2 BOOTLOAD

The Bootload operation is normally preceded by an Initialize operation generated from either the Operator's Panel or a Link Adapter.

The Initialize operation will leave the MPC in a reset state, ready to start executing microinstructions, starting at Control Store Location 0, when the Start Switch is actuated, (or automatically, in the case of the Link Adapter Initialize).

The Bootload operation itself must be accomplished by a microprogram. This microprogram must either start at absolute Control Store Location 0, or be branched to/from a program which starts at Location 0.

The Configuration Switches on the Operator's Panel (paragraph 4.2) will be set to indicate required Bootload parameters, and will be accessed by the Bootload routine.

3.3 INTERRUPT MECHANISM SETUP

It is mandatory that the various registers associated with the interrupt mechanism and RBA Pointers be set up before the MPC begins running application microprograms. This set up must be done by a set up microprogram.

The normal startup sequence of events is as follows:

- Initialization by Operator Panel Switch or Link Adapter signal. This leaves the MPC registers and Interrupt Mechanism in a defined state (paragraph 3.1), and the MPC halted if initialized by Operator Panel Switch.

- If initialized from Operator's Panel, subsequent actuation of the Start Switch on Operator Panel starts microinstruction execution from absolute ROS location 0.

- If initialized from Link Adapter, microinstruction execution begins automatically from absolute ROS location 0 upon the trailing edge of the Initialize signal.

At this point, the microprogram located at location 0 must set up the Interrupt Mechanism as required by the MPC configuration, microprograms to be run, etc.

The following registers and modes must be set up:

- Device Adapter Control Register
- Test and Diagnostic Register
- Dual Mode Bit
- Normal RBA Pointer
- Inhibit EN-1
- Inhibit Macro Interrupt

See paragraph 2.1 for the definition of these registers and modes.

3.4 MICROINSTRUCTION ACCESS

The normal mode of operation in the MPC is the execution of microinstructions out of control store. However, capability exists in the MPC to allow microinstruction execution to be performed out of main memory, or to execute microinstructions received via the Device Adapter Interface or the A,B accumulators. These capabilities are described in the following paragraphs.

3.4.1 Microinstruction Access from Control Store

Microinstructions are retrieved from control store two at a time. Each pair of microinstructions is referred to as an "even/odd" pair, the even microinstruction being the lower address instruction, and the first to be executed.

In order to accomplish the concurrent branch requirement, all branch microinstructions must be contained in control store as the "odd" microinstruction of the even/odd pair.

3.4.2 Microinstruction Access from Main Memory

By means of the "Read Memory Data Register" microinstruction (see Appendix A), the contents of a memory word can be loaded directly into an internal microinstruction execution register. As a result, the contents of the memory word will be executed as the "odd" microinstruction of the even/odd pair following the even/odd pair containing the Read Memory Data Register microinstruction.

This is illustrated as follows:

Control Store Address	Even Microinstruction	Odd Microinstruction	
A	X	RMDR	
A + 2	X	X -------------→	Replaced by contents of memory word
A + 4	X	X	

Assume a microprogram in control store is as shown above.

The "even/odd" pair at control store address "A" is pulled.

The "even" microinstruction is executed.

The "odd" microinstruction is executed. In this example, this microinstruction is the RMDR microinstruction which transfers the contents of a previously addressed memory word into the internal execution register. (It is assumed that the above example was preceded by a "Start Read Memory Word" microinstruction.)

Now the "even" microinstruction at control store address "A + 2" is executed.

The contents of the memory word read by the RMDR microinstruction are now executed as the "odd" microinstruction of the even/odd pair at control store location "A + 2," replacing the odd microinstruction pulled from control store.

The next microinstruction pair executed will be the even/odd pair located at control store location "A + 4."

In order to use this capability to execute microprograms which reside in main memory, an "execution" microprogram must reside in control store.

This "execution" microprogram must execute the following functions:

- Maintain the Memory Address used to access the microinstructions in memory.
- Execute the "Start Read Memory Cycle" and "Read Memory Data Register" microinstructions to retrieve each microinstruction from memory.

The RMDR microinstruction which transfers the memory word to the internal execution register, must be in an "odd" location in the control store, or an "even" location with a Branch instruction in the "odd" location.

Special consideration is required when a Branch Start Read Memory Cycle or Start Write Memory Cycle is executed out of memory.

3.4.2.1 Start Read/Start Write Memory Cycle Microinstruction

When either of these microinstructions are executed out of memory, hardware will detect this occurrence, and force the next "even" microinstruction to be either a "Read Memory Data Register" or "Write Memory Data Register" microinstruction. The source or destination register must be accumulators A, B in this case.

This is illustrated below:

Control Store Address	Even Instruction	Odd Instruction
A	X	RMDR
A + 2	X	X
A + 4	X	X

Replaced by contents of memory word, and is a Start Read or Start Write Memory Cycle microinstruction

The microinstruction read from memory will be executed instead of the odd microinstruction of the "A + 2" even/odd pair. Hardware will detect that the microinstruction is a Start Memory type, and will force the next even microinstruction pulled from control store, location A + 4, to be a Read or Write Memory Data Register (to or from AB), as required. This microinstruction must be encoded as a "no-op" in the control store execution microprogram.

3.4.2.2 Branch Microinstruction

The execution of a branch microinstruction out of memory must be handled by interpretive routines included as a part of the "execution" control store microprogram.

One method would be to include, in the memory-resident microprogram, branch instructions which, when executed, will cause a branch within the execution control store program to the applicable interpretive routine for the type branch to be made.

It is also possible for a microprogram which is to reside in main memory to execute branches within itself by arithmetically modifying the hardware register being used as the main memory address (actually the microinstruction address for the microprogram in main memory). With this method, actual branch microinstructions would not be required.

3.4.3 Microinstruction Access from DAI

The Device Adapter Interface microinstruction allows the option of transferring the DAI "Data-In" lines into the internal microinstruction execution register.

This provides the capability of executing microinstructions received from an external source via the DAI.

The capability of loading and executing microinstructions in this manner is similar to that described in paragraph 3.4.2 for microinstructions loaded from main memory.

3.4.4 Microinstruction Access from Accumulators A,B

The Load Interrupt Mechanism microinstruction allows the contents of the A,B accumulators to be loaded directly into the internal microinstruction execution register.

This allows the same capabilities of noncontrol store microinstruction access and execution as described in paragraphs 3.4.2 and 3.4.3, above.

3.5 FIXED CONTROL STORE OPERATIONAL ADDRESSES

There are several operational control store addresses which are fixed, and not changeable.

These are defined as follows:

3.5.1 Initialize State

When the Initialization procedure (defined in paragraph 3.1) is executed as a result of either the Initialize Switch on the Operator's Panel, or a signal from a Link Adapter, subsequent microinstruction execution begins from absolute Control Store Address 0.

Cont. on Page 52

3.5.2 Low-Level EN-1 Interrupt

When a Low-Level EN-1 interrupt is executed, the MPC will automatically jump to one of the following locations:

- Decimal control store location 32 if not in the T&D mode.
- Decimal control store location 32 + 128 if in the T&D mode.

(T&D mode set as described in paragraph 2.1.1.2(3))

3.5.3 Error Interrupt

When an error interrupt is executed, the MPC will automatically jump to one of the following locations:

- Decimal control store location 64 if not in the T&D mode.
- Decimal control store location 64 + 128 if in the T&D mode.

4. INTERFACE REQUIREMENTS

4.1 DEVICE ADAPTER INTERFACE

The Device Adapter Interface, as defined in Specification 43A177876 is used between the MPC and connected Device Adapters. The MPC will allow connection of up to four Device Adapters.

The processing structure of the MPC will provide the registers and data paths to control and communicate over this interface. Specific data path requirements are defined in the description of the DAI microinstruction, Appendix A. Requirements with respect to interpretation of DAI interrupt lines, are defined in paragraph 2.1.2, Interrupt Mechanism.

Some special considerations concerning the Device Adapter Interface are described in the following paragraphs.

4.1.1 Remote Initialize Function from Link Adapter

The initialize MPC function, as described in paragraph 3.1, can be initiated from an external subsystem, such as a central processor. However, there is no unique line in the Device Adapter Interface to cause such a function.

Therefore, a special "Remote Initialize" line is defined to exist between a Link Adapter and the MPC. This line is not a standard interface line, but exists only on Device Adapter ports 2 and 3, which are the normal ports for the Link Adapter connection.

By means of this line, a Link Adapter can set a signal which will cause the MPC to execute the Initialize function. During the duration of this signal, the MPC will be held in the initialized, halt state. When the signal drops, the MPC will begin microinstruction execution, starting at absolute control store address 0. The Initialize signal, as received from a Link Adapter, must be a minimum of 300 nanoseconds, to ensure the initialization of the MPC.

The Initialize signal will be passed directly to the four Device Adapter ports (over the "initialize out" DAI line). It is the responsibility of the Link Adapter (and/or the external system to which it is connected) to ensure that the width of the Initialize signal is adequate to initialize all connected Device Adapters.

The "Remote Initialize" signal can be inhibited by Maintenance Panel Switch. The signal is also inhibited if the "T&D Mode" bit of the T&D Register is set <u>and</u> the MPC is <u>not</u> in the "halt" state.

4.1.2 Operational-In-Line

This line is a standard Device Adapter Interface line.

If, during the execution of any Device Adapter Interface microinstruction, this line indicates a "nonoperational" state of the Device Adapter, the MPC will complete the execution of the microinstruction independent of a signal from the DA on the "Response-In" DAI line. (The DAI microinstruction can optionally specify a "wait for response-in" before execution completion.)

The eight Status Lines from the DAI will be forced to all ones in this case, and the 16-data-in lines from the DAI will be forced to all ones with correct parity.

4.1.3 Operational-Out Line

This line is a standard Device Adapter Interface line.

This line will reflect the operational state of the MPC as follows:

- Line set = MPC is running, that is, executing microinstructions.
- Line reset = MPC is in a halt state, that is, the MPC clock is stopped.

Cont. on Page 55

4.1.4 Initialize Line

This is a standard DAI line.

When the Initialize function (as defined in paragraph 3.1) is executed, a signal is sent on the Initialize line to each of the four Device Adapter ports.

This signal is intended to cause each connected Device Adapter to initialize itself to a reset state.

4.1.5 Power Supply Contact Lines

These are not standard DAI lines.

There are four relay contacts (or equivalent) which are made available to Device Adapter ports 2 and 3 (normal ports for Link Adapter connection). Two contacts are made available via backpanel connection to each of these DA ports, representing a total of four backpanel connections.

These contacts will be open if d-c power is off, or if the MPC is set to the "Halt" state (internal clock not running).

Upon power up, these contacts are guaranteed not to close until d-c power is stabilized and the MPC internal clock is running.

When a power down sequence originates via the Power-Off switch, these contacts are guaranteed to open before d-c power loses regulation. If a power down sequence originates from detection of a malfunction or out of tolerance condition, the contacts will open, but cannot be guaranteed to open before d-c regulation is lost.

4.1.6 Control Reset Line

This line to the Device Adapter is used to cause the Device Adapter to reset selected control logic.

The intent of this line is not to cause d-c initialization of the entire DA. The defipition of "Selected Control Logic" is a function of each individual Device Adapter type.

4.2 OPERATOR PANEL

This panel will contain those switches and indicators which must be accessed by normal system operating personnel.

The panel should be physically accessible from the outside of the MPC cabinet, without opening a door.

The Operator Panel, and its indicators and switches, will conform to Group Standards B01.9.

The following switches and indicators will be included on the Operator Panel.

4.2.1 Power On/Power Off Switches

These are two separate pushbutton switches. Each action of the switch complements its state.

The Power On switch is labeled "POWER ON". When the switch is actuated, d-c power is applied to the MPC and the switch becomes illuminated.

The Power Off switch is labeled "POWER OFF". When the switch is actuated, d-c power is dropped from the MPC and the POWER ON switch goes dark.

4.2.2 Initialize Switch

This switch is labeled "INITIALIZE." Actuation of this pushbutton switch causes the execution of the Initialize function, as described in paragraph 3.1. The switch is not illuminated.

4.2.3 Start Switch

This switch is labeled "START." Actuation of this pushbutton switch, causes the MPC to begin execution of microinstructions. The switch has no effect unless the MPC is in the "Halt" state. The switch is not illuminated.

4.2.4 Auto/Manual Switch

This is an alternate action pushbutton switch. Each actuation complements its state.

The switch contains a split field indicator labeled "AUTO" in the upper field and "MANUAL" in the lower field. When actuated to the "MANUAL" state, the switch causes the Manual Mode condition to be generated in the MPC. (See paragraph 2.1.2.3.) This condition is testable under microinstruction control.

When actuated to the "AUTO" state, the Manual Mode condition is removed. The split field indicator is always lit in the field defining the state of the switch.

4.2.5 Operator Interrupt Switch

This pushbutton switch is labeled "OPER INT.". Actuation of this pushbutton switch, generates the Operator Interrupt condition, as described in paragraph 2.1.2.3. This condition is testable under microinstruction control. Upon setting of this condition, the switch will become illuminated. The switch will stay illuminated until the interrupt condition is reset by the execution of a "Store Interrupt Mechanism Register" microinstruction.

4.2.6 Error Reset Switch

This pushbutton switch is labeled as follows:

ROS	EXT
INTERNAL	

The switch will be illuminated in the field defining the category of error condition being registered in the Error Data Register (paragraph 2.1.1.2(4)):

ROS - error detected during the access of a microinstruction from control store.

EXT - error detected during a Device Adapter Interface or main memory operation.

INTERNAL - error detected internal to the processing structure of the MPC (internal busses, etc.)

Actuation of this pushbutton switch results in the resetting of the Error Data Register.

4.2.7 Data Switches

This is a set of 16 two-position toggle switches. These switches can be read by the "Store Interrupt Mechanism Register" microinstruction.

These switches are labeled 0-15, with switch 0 the most significant switch.

4.2.8 Configuration Switches

This is a set of 16 switches, labeled 0-15. The state of these switches can read by the "Store Interrupt Mechanism Register" microinstruction.

The intent of these switches is to provide configuration data required for the conditioning of various control and bootload microprograms, as well as certain MPC hardware functions. These functions are described in subsequent paragraphs.

Since these switches will be used for sensitive configuration data, and will be changed only when physical subsystem changes are to be made, they should be protected from accidental modification by means of a cover or lock arrangement.

4.2.8.1 Hardware Control Functions

The following switches are used to control hardware functions.

- Switch 4 - This switch, is set, defines the existence of a "shared memory configuration."

  The "Start Read Memory Cycle" microinstruction (paragraph A9.1) and "Start Write Memory Cycle" microinstruction can use the state of this switch in forming the memory address. This capability allows automatic memory address modification as a function of the memory configuration.

- Switch 0,1,2,3 - Device Adapter On/Offline

  These four switches are used to establish the online/offline state ~~of~~ Device Adapters.

Switch 0 is associated with DA Port 0, Switch 1 with DA Port 1, etc.

When a configuration switch is set to offline, any Device Adapter microinstruction addressed to that DA Port will be inhibited. The "Select" line to the Device Adapter will not be enabled, which will cause the 8 Status Lines and the 16 Data-In Lines from the DAI Port to be forced to all ones. The EN-1 and EN-2 lines will also be inhibited.

4.2.8.2 Microprogram Conditioning Functions

The remaining Configuration Switches do not directly affect the execution of hardware functions, but are testable by microinstruction, and can be used to affect the execution of microprograms.

Typical of the conditions to be set into these switches are the following:

● Bootload Control Store (Program 1 or 2)	(1 switch)
● Bootload DA Number	(2 switches)
● Bootload Device Number (if required)	(3 switches)
○ Basic Memory Module Size (1K or 8K words)	(1 switch)
● Number of Memory Modules	(2 switches)

Firm definition of the exact allocation of these switches will be contained in related application specifications for MPC peripheral controller configurations, Isolation Text Routine procedures, etc.

4.2.9 Address Switches

These switches may be set to provide bits 3-14 of a control store address where bit 15 (least significant) of the address is assumed zero. This address can be used as a branch address by one option of the Absolute Branch microinstruction.

These switches may be assigned other maintenance functions as required.

4.2.10 Control Reset Switches

This switch resets the control logic of the MPC, but does not cause the "initialize" function to occur.

Following the execution of the reset function initiated by the switch, the MPC will execute an Absolute Branch to the control store location specified by the Address Switches.

The Control Reset signal is sent to all four DA Ports over the individual "Control Reset" DAI line. (See paragraph 4.1.6)

4.2.11 Data Lights

This set of 16 indicators will illuminate the current state of the main MPC data bus, and therefore, the contents of the last register or register pair upon which a microinstruction was executed.

By means of the "Change Interrupt Mechanism Condition" microinstruction (which allows an MPC halt to be programmed), register displays can be micorprogrammed.

These indicators are labeled 0-15, with indicator 0 the most significant indicator positions.

4.3 MAINTENANCE PANEL

The Maintenance Panel will contain the switches and indicators necessary to maintain the MPC to the standards prescribed in paragraph 6 of this specification.

This panel will not be accessible to normal system operating personnel, and should be concealed except during the execution of maintenance functions.

The following functions can be executed from the Maintenance Panel:

- Display/Load all registers.
- Single Step control of microinstruction execution.
- Selective control of error ignore, error halt.
- Selective halt on specific control store address.

- Selective halt on specific main memory address.
- Simulate microinstruction branch address from switches.
- Simulate micrqinstruction from switches.
- Simulate main memory data from switches.
- Simulate DAI data from switches.
- Display Isolation Test Routine outputs required for "Fault Dictionary Lookup" (see paragraph 6.1.2).

4.4 MAIN MEMORY AS INTERFACE BETWEEN MPC'S

As defined in paragraph 2.3.5, two MPC's can share a common main memory. This shared memory capability can serve as an MPC to MPC link in configurations requiring MPL-to-MPL communication.

5. GENERAL DESIGN REQUIREMENTS

5.1 CONFORMANCE REQUIREMENT

The MPC will conform to the requirements of 43A177851, General Design Requirements for GE-655 and GE-355 systems.

5.2 PHYSICAL PACKAGING

The MPC is to be considered a physical entity which is packaged into a physical controller configuration. Therefore, packaging requirements are governed by the specific requirements of each individual controller configuration which utilizes the MPC. These requirements will be contained in the individual controller configuration specifications.

5.3 POWER SHARING/DISTRIBUTION

The MPC and the Device Adapters with which it is combined to form a physical controller configuration are to share common power supplies.

Power supplies for devices, and device oriented electronics may be included in the physical controller configuration, depending on the particular controller configuration requirements. This will be specified in the individual controller specifications.

Cont. on Page 64

6. RELIABILITY AND MAINTAINABILITY REQUIREMENTS

6.1 GENERAL

The requirements of this section assume that the General Design Requirements of Section 5 are fulfilled. In addition, a continuing program of product improvement is assumed during development, design and production of the product.

6.1.1 General Maintainability Requirements

The MPC hardware will be designed to allow meeting the maintainability requirements of the Advanced Product Line, as defined in APL Line System Specification, dated March 2, 1970, paragraph 5.2.

6.1.2 Isolation Test Routine Capability

The basic method of diagnosing the MPC will be via "Isolation Test Routines." These routines consist of a series of microprogrammed test sequences and associated control routines. These routines can be executed out of either main memory or control store.

To fully utilize this capability, all possible failure modes of the MPC must be pre-analyzed and cataloged into a hard-copy Fault Dictionary.

The maintenance procedure, therefore, involves the manual initiation and execution of the ITR routines. The action taken by the ITR routine upon detection of a fault will depend on the type of fault, and status of the MPC at the time the fault is detected. If desired, and if enough of the hardware has been proved operational, the ITR routine may attempt to log out Fault Dictionary information to the Link Adapter. Otherwise, the ITR routine may chose to simply halt the MPC, with Maintenance panel lights indicating the fault symptoms.

The normal procedure will be to run the ITR sequences immediately upon manual startup, following initialization.

In controller configurations utilizing the MPC, ITR routines can also be included to diagnose connected Device Adapters.

6.2 DEFINITIONS

Reliability and Maintainability definitions shall be per Group Standard B03.1 except as otherwise defined in this specification.

6.3 MEASURING DEVICES

6.3.1 Instrumentation

Power-on meter and other instrumentation shall be designed into all MPC's which are designated as factory or field evaluation units. The installation and removal of such instrumentation shall be possible as a routine operation. Power supplies will be designed considering these instrumentation requirements. The addition of instrumentation shall not degrade the reliability performance of the controller or interfere with its operation.

6.3.2 Unit Identification

Positive identification of all logic cards, control stores, power supplies and other critical assemblies shall be through direct-attachment identification labels, embossing, silk-screening or stamping. Such identification shall be used for determining the date of replacement and the number of repair cycles.

6.4 OFFLINE REPAIRABILITY

The replacement of single components of piece parts, which are known to be in the failed state, shall not exceed ten minutes. Such replacement to be performed by the use of standard hand tools given in the standard tool catalog wherever possible.

6.5 INTERCHANGEABILITY

Parts and assemblies, designed and manufactured to the same specification and revision level, shall be electrically, mechanically, and functionally interchangeable.

All spare parts shall be interchangeable to the Optimum Replaceable Unit level. If adjustments are required, the time to adjust shall be an integral part of the time to repair.

6.6 ADJUSTMENTS

Field adjustments shall be reduced to an absolute minimum. Where adjustments are unavoidable, they shall:

- Require no scheduled field adjustment.
- Be capable of objective calibration.
- Be secured such that they will not change positions under site conditions.
- Require no interdependent adjustment between to or more individual adjustments.
- Require no more than one man to accomplish.
- Be adjustable to specified values within the tolerance of standard field instrumentation or built in calibration standards, wherever possible.

An out-of-tolerance condition or any adjustment, shall not be destructive to any circuitry or component affected by the adjustment or process of adjusting.

6.7 PREVENTATIVE MAINTENANCE (PM)

PM for the Basic MPL shall consist only of changing air filters as provided for in standard Field Engineering PM schedules.

6.8 DUTY CYCLES

The average ratio of channel busy time to system use time is assumed to 0.7.

The minimum number of cold starts per week is 1. A cold start is defined as a start-up after a power-off period of greater than four hours.

The minimum number of power-up, power-down sequences per week is 1.5.

6.9 STEP 4 REQUIREMENTS

Prior to first shipment, and when tested in a maximum nonredundant configuration, the MPC shall have demonstrated the following reliability and maintainability characteristics to a 50 percent confidence level.

PROCESSOR CONFIGURATION	MTBF	MTTR
Logic (Processing Structure)	22,000 hr.	20 min.
Power Supply	20,000 hr.	20 min.
Control Store (8K x 18 bits)	12,000 hr.	20 min.
Main Memory (32K x 18 bits)	8,000 hr.	20 min.
Diagnostic Hardware and Maintenance Panel	50,000 hr.	30 min.
Total MPC	3,090	25 min.

MTTR is measured from the beginning of the ITR run until the faulty component has been repaired and the subsystem has been re-initialized and successfully run through the ITR phase.

Conformance to these requirements is to be measured through specification 43A228294, "PED In-House Data Collection Plans and Procedures."

6.10 STEP 5 REQUIREMENTS

Prior to unrestricted production, the MPC shall have demonstrated the following availability, mean-time-between-system-interrupt and mean-time-of-system interrupt characteristics to a 90 percent confidence level.

Availability	- 98.5 percent
MTBSI	- 100 hours
MTOSI	- 1.5 hours

Conformance to these requirements to be measured through a Field Evaluation Data Collection Plan.

6.11 UNRESTRICTED PRODUCTION

One year after installation of the first production MPC, the following best estimates are expected (design goals only):

Availability	=	99.6 percent
MTBSI	=	100 hours
MTOSI	=	0.5 hours

Conformance to these goals shall be measured through the Field Reporting System.

6.12 FIELD PERFORMANCE MEASUREMENT

6.12.1 Incident Reports

Reports shall be submitted from designated sites whenever maintenance or operator action is required to restore the system to useful status. Each incident report will contain at least the following:

- Complete identification of the major unit (serial and model numbers).
- Elapsed power-on time at incident occurrence.
- Reading of the "channel busy meter" at each incident.
- Description of diagnosed fault.
- Corrective action taken.
- Time required to restore the system to operational status.

6.12.2 Failure Analysis

All fully components or subassemblies shall be identified with the incident report and returned to Reliability Engineering for failure analysis.

6.12.3 Data Reduction and Storage

Data from all incident reports will be correlated with results of failure analyses, reduced to standard format and stored for later retrieval.

6.12.4 Equipment Selection

All MPC's manufactured through Step 4 releases shall be monitored. This includes all engineering and prototype models and pilot MPC's manufactured for initial shipments. After release for unrestricted production (Step 5), selected MPC's will be designated for additional monitoring. A selection plan will be developed which will depend upon the results on previous MPC's through the Step 4 pilots.

6.12.5 Evaluation Period

Performance monitoring shall continue until the data shows conformance to R/M requirements of this specification.

6.12.6 Performance Reports

Performance reports shall be issued periodically as soon as evaluation tests begin on the Engineering Model. Final performance reports shall be issued immediately prior to Step 4 and Step 5 Product Reviews. A final performance evaluation report shall be issued at the completion of the Evaluation Period (section 6.12.5).

APPENDIX A

A1. MICROINSTRUCTION REPERTOIRE

A1.1 REGISTER NOTATION

The following registers are referred to in the description of the microinstruction repertoire:

1. Accumulator A -- Eight-bit accumulator, bits designated 0-7, bit 0 = most significant bit.

2. Accumulator B -- Eight-bit accumulator, bits designated 0-7, 0 = most significant bit.

3. Accumulators A,B -- The contents of the two 8-bit accumulators are treated as one 16-bit field, with the contents of accumulator A providing the high-order bits of the field; also referred to as (AB)0-15.

4. RBA Register -- Any one of the sixteen 8-bit registers of an RBA half. The sixteen registers are designated R0 - R15.

5. RBA Even/Odd Register Pair -- The contents of a pair of RBA registers are treated as one 16-bit field, designated 0-15. The register pair will always be made up of an even and an odd register, R0/R1, R2/R3,...R14,R15.

   The contents of the even RBA register provides the high-order bits of the combined 16-bit field.

6. Upper Indicator Register (Upper IR) -- Four-bit indicator register.

7. Lower Indicator Register (Lower IR) -- Four-bit indicator register.

8. Interrupt Mechanism Status -- This is a four-bit field tested during certain branch operations.

   Bit 0: Manual Mode
   Bit 1: Operator Interrupt
   Bit 2: Interval Timer Runout
   Bit 3: Macro Interrupt

9. TRACE Register -- Thirteen-bit address storage register. Contents referred to as (TRACE) 3-15.

10. ROSAR -- Twelve-bit Control Store Address Register. Contents referred to as ROSAR_{3-14}. Contents of this register are always even.

11. AUXAR -- Twelve-bit Auxiliary Control Store Address Register. Contents referred to as AUXAR_{3-14}. Contents of this register are always even.

12. Device Adapter Number Register -- Two-bit register specifying a DAI port.

13. Branch Test Register -- Eight-bit register, used for result testing. Referred to as BT_{0-7}.

14. Test and Diagnostic Register -- Eight-bit register used to control the operating state of the MPC. Referred to as $\text{T\&D}_{0-7}$.

15. Error Data Register -- Sixteen-bit register used to indicate specific error occurrences. Referred to as EDR_{0-15}.

A2. ARITHMETIC OPERATIONS

A2.1 BINARY ADD

Arg. 1: can be either accumulator A or accumulator B.

Arg. 2: can be any one of RBA registers, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry

Microinstruction Options:

1. Result of (Arg. 1) + (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.

2. Result of (Arg. 1) + (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Lower IR affected.

3. Result of (Arg. 1) + (Arg. 2) + (Previous Carry) replaces either (Arg. 1) or (Arg. 2).

   Previous Carry as held in Upper IR. Zero condition propagated in Upper IR. (Operation result "ANDed" with previous state of Zero Indicator.)

   Only Upper IR affected.

4. Result of (Arg. 1) + (Arg. 2) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A2.2 BINARY SUBTRACT

Arg. 1: can be either accumulator A or accumulator B.

Arg. 2: can be any one of RBA registers, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry (Borrow = 0 if Carry = 1)

Microinstruction Options:

1. Result of (Arg. 1) - (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.

2. Result of (Arg. 1) - (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Lower IR affected.

3. Result of (Arg. 1) - (Arg. 2) - (Previous Borrow) replaces either (Arg. 1) or (Arg. 2).

   Previous Borrow as held in Upper IR
   Zero condition propagated in Upper IR
   Only Upper IR affected

4. Result of (Arg. 1) - (Arg. 2) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A2.3 ADD CARRY

Arg. 1: can be accumulator A, accumulator B or any one of RBA Registers R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry

Microinstruction Options:

1. Result of (Arg. 1) + (Carry from Upper IR) replaces (Arg. 1).

   Only Upper IR affected.

2. Result of (Arg. 1) + (Carry from Lower IR) replaces (Arg. 1).

   Only Lower IR affected.

3. Result of (Arg. 1) + (Carry from Upper IR) replaces (Arg. 1).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Result of (Arg. 1) + (Carry from Lower IR) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A2.4 SUBTRACT BORROW

Arg. 1: can be accumulator A, accumulator B or any one of RBA registers R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry (Borrow = 0 if Carry = 1)

Microinstruction Options:

1. Result of (Arg. 1) - (Borrow from Upper IR) replaces (Arg. 1).

   Only Upper IR affected.

2. Result of (Arg. 1) - (Borrow from Upper IR) replaces (Arg. 1).

   Only Lower IR affected.

3. Result of (Arg. 1) - (Borrow from Upper IR) replaces (Arg. 1).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Result of (Arg. 1) - (Borrow from Lower IR) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

12.5 NEGATE

Arg. 1: can be any RBA register, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry

Microinstruction Options:

1. Replace (Arg. 1) with 2's complement of (Arg. 1).

   Only Upper IR affected.

2. Replace (Arg. 1) with 2's complement of (Arg. 1).

   Only Lower IR affected.

3. Replace (Arg. 1) with 2's complement of (Arg. 1) (Carry from Upper IR).

   Only Upper IR affected
   Zero condition propagated in Upper IR

4. Change Lower IR to reflect result of 2's complement of (Arg. 1).

   Arg 1 not affected
   Only Lower IR affected

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A2.6 COMPLEMENT

Arg. 1: can be any RBA register, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

Microinstruction Options:

1. Replace (Arg. 1) with 1's complement of (Arg. 1).

   Only Upper IR affected.

2. Replace (Arg. 1) with 1's complement of (Arg. 1).

   Only Lower IR affected.

3. Replace (Arg. 1) with 1's complement of (Arg. 1).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Change Lower IR to reflect result of 1's complement of (Arg. 1).

   Arg. 1 not affected.
   Only Lower IR affected.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A3. LOGICAL OPERATIONS

A3.1 Logical AND

Arg. 1: can be either accumulator A or accumulator B.

Arg. 2: can be any one of RBA registers, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

Microinstruction Options:

1. Result of (Arg. 1) ANDed with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.

2. Result of (Arg. 1) ANDed with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Lower IR affected.

3. Result of (Arg. 1) ANDed with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Result of (Arg. 1) ANDed with (Arg. 2) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to be also placed in the Branch Test Register.

A3.2. Logical OR

Arg. 1: can be either accumulator A or accumulator B.

Arg. 2: can be any one of RBA registers, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

Microinstruction Options:

1. Result of (Arg. 1) OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Lower IR affected.

2. Result of (Arg. 1) OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.

3. Result of (Arg. 1) OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Result of (Arg. 1) OR'd with (Arg. 2) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A3.3 Exclusive OR

Arg. 1: can be either accumulator A or accumulator B.

Arg. 2: can be any one of RBA registers, R0-R15.

Upper/Lower Indicators reflect the following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

Microinstruction Options:

1. Result of (Arg. 1) EXC. OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Lower IR affected.

2. Result of (Arg. 1) EXC. OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.

3. Result of (Arg. 1) EXC. OR'd with (Arg. 2) replaces either (Arg. 1) or (Arg. 2).

   Only Upper IR affected.
   Zero condition propagated in Upper IR.

4. Result of (Arg. 1) EXC. OR'd with (Arg. 2) used to change Lower IR only.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A4. SINGLE BYTE LOAD/STORE OPERATIONS

Arg. 1: can be any one of RBA registers, R0-R15.

Upper/Lower Indicator Registers reflect the following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

A4.1 STORE A

Contents of accumulator A are stored in register specified by Arg. 1.

A4.2 STORE B

Contents of accumulator B are stored in register specified by Arg. 1.

A4.3 LOAD A

Contents of accumulator A are replaced by contents of register specified by Arg. 1.

A4.4 LOAD B

Contents of accumulator B are replaced by contents of register specified by Arg. 1.

A4.5 STORE BRANCH TEST REGISTER

Contents of Branch Test Register are stored in register specified by Arg. 1.

A4.6 LOAD BRANCH TEST REGISTER

Contents of Branch Test Register are replaced by contents of register specified by Arg. 1.

Microinstruction options for A4.1 through A4.6:

1. Execution of microinstruction affects only Upper IR.

2. Execution of microinstruction affects only Lower IR.

3. Execution of microinstruction affects only Upper IR. Zero condition is propagated in Upper IR.

4. Execution of microinstruction affects only the Lower IR. No other register is affected.

In all cases the microinstruction can optionally cause the result to also be placed in the Branch Test Register.

A5. INDICATOR REGISTER OPERATIONS

A5.1 LOAD INDICATOR REGISTER

Arg. 1: can be any one of RBA registers, R0-R15.

Microinstruction Options:

1. Contents of Upper IR replaced by contents of four most-significant bits of register specified by Arg. 1.

   Contents of Lower IR not affected.

2. Contents of Lower IR replaced by contents of four least-significant bits of register specified by Arg. 1.

   Contents of Upper IR not affected.

3. Contents of Upper IR replaced by contents of four most-significant bits of register specified by Arg. 1.

   Contents of Lower IR replaced by contents of four least-significant bits of register specified by Arg. 1.

In all cases the microinstruction can optionally cause the contents of Argument 1 to be loaded into the Branch Test Register:

$$(\text{Arg. } 1) \longrightarrow (\text{Branch Test Register})_{0-7}$$

A5.2 STORE INDICATOR REGISTER

Arg. 1: can be any one of RBA registers, R0-R15.

Microinstruction Options:

1. Contents of Upper IR replace most-significant four bits of register specified by Arg. 1.

2. Contents of Lower IR replace least-significant four bits of register specified by Arg. 1.

In all cases the microinstruction can optionally cause the resulting contents of both Indicator Registers to be placed in the Branch Test Register.

$$(\text{Upper IR})_{0-3}\ (\text{Lower IR})_{4-7} \longrightarrow (\text{Branch Test Register})_{0-7}$$

A6. LOGICAL/ARITHMETIC IMMEDIATE VALUE OPERATIONS

These microinstructions will contain a four-bit constant which can be used to operate on either the least-significant or most-significant four bits of any one of the following registers:

Accumulator A
Accumulator B
Any RBA Register, R0-R15
Upper Indicator Register
Lower Indicator Register
Branch Test Register

All microinstructions in this class can optionally cause the resulting conditions to be loaded into either the Upper or Lower Indicator Register. (If an Indicator Register is the result register named in the microinstruction, the result conditions are not loaded.)

Resulting conditions can also be optionally loaded into the Branch Test Register.

In the case of arithmetic immediate operations to be executed on the low-order four bits of the designated register, a "borrow" or "carry" will be propagated into the high-order four bits of the designated register.

In all other cases only the specific four-bit field of the designated register will be affected.

A6.1 ADD IMMEDIATE

Immediate value constant is added to the indicated field.

Upper/Lower Indicator Registers reflect following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry

A6.2 SUBTRACT IMMEDIATE

Immediate value constant is subtracted from the indicated field.

Upper/Lower Indicator Registers reflect following conditions:

Most Significant Bit
Overflow
Zero Condition
Carry

A6.3 AND IMMEDIATE

The immediate value constant is "ANDed" with the indicated field.

Upper/Lower Indicator Registers reflect following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

Cont. on Page 84

A6.4 OR IMMEDIATE

The immediate value constant is "OR'd" into the indicated field.

Upper/Lower Indicator Registers reflect following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

A6.5 EXCLUSIVE OR IMMEDIATE

The immediate value constant is "EXCLUSIVE OR'd" into the indicated field.

Upper/Lower Indicator Registers reflect following conditions:

Most Significant Bit
All One's Condition
Zero Condition
Least Significant Bit

A7. <u>IMMEDIATE VALUE LOAD OPERATION</u>

A7.1 LOAD IMMEDIATE VALUE

This microinstruction will contain an 8-bit constant which can optionally be loaded into any one of the following registers:

Accumulator A
Accumulator B
Branch Test Register
Any RBA Register, R0-R15
Upper/Lower Indicator Registers (high-order four bits of constant loaded into Upper IR, low-order four bits of constant into Lower IR)

Indicator Registers are not affected by this microinstruction unless specified as the result register.

A8. SHIFT OPERATION

Two types of shift operations are possible: single register shifts (single byte), and double register shifts (double byte).

A8.1 SHIFT ONE BIT, SINGLE

This microinstruction causes a one-bit shift, either left or right, to be executed on the byte contents of any one of the following registers:

Accumulator A
Accumulator B
Any RBA Register, R0-R15

The Upper Indicator Register will not be affected.

The Lower Indicator Register will always be set to zero except for the "shifted out" bit.

Note: The "shifted out bit" is saved in the Lower IR in the following cases:

High-order bit shifted left out of accumulator A, or any even RBA register (R0, R2, etc.)

Low-order bit shifted right out of accumulator B, or any odd RBA register (R1, R3, etc.)

In the following cases the "shifted out bit" is not saved:

High-order bit shifted left out of accumulator B, or any odd RBA register (R1, R3, etc.)

Low-order bit shift right out of accumulator A, or any even RBA register (R0, R2, etc.)

Microinstruction Options:

1. Shift Left One

   Zero is shifted into low-order bit position of register.

2. Shift Left One and Insert

   Contents of "Shift" bit of Lower IR shifted into low-order bit position of register. (Accumulator B or "odd" RBA register only.)

3. Shift Left One, Rotate

   Bit shifted out of most-significant bit position of register is shifted into least-significant bit position of register. (Accumulator B or "odd" RBA register only.)

4. Shift Right One

   Zero is shifted into high-order bit position of register.

5. Shift Right One and Insert

   Contents of "Shift" bit of Lower IR shifted into high-order bit position of register. (Accumulator A or "even" RBA register only.)

6. Shift Right One, Rotate

   Bit shifted out of least significant bit position of register is shifted into most-significant bit position of register.

   (Accumulator A or "even" RBA register only.)

SHIFT ONE BIT, DOUBLE

This microinstruction causes a one-bit shift, either left or right, to be executed on any of the following pairs of registers:

Accumulators A,B
Any even/odd pair of RBA registers, R0/R1;
R2/R3; etc.

For purposes of this operation the contents of the two registers are considered to be one contiguous 16-bit data field.

The Upper Indicator Register is not affected.

The Lower Indicator Register will always be set to zero, except for the "Shifted Out Bit."

Shifted Out Bit = High-order bit of most significant byte on a left shift.

Low-order bit of least significant byte on right shift.

Microinstruction Options:

1. Shift Left One, Double

   Zero shifted into least-significant bit of least-significant byte.

2. Shift Left One, Double and Insert

   Contents of "Shift" bit of Lower IR shifted into least-significant bit of least-significant byte.

3. Shift Left One, Double, Rotate

   Bit shifted out of most-significant bit of most-significant byte is shifted into least-significant bit of least-significant byte.

4. Shift Right One, Double

   Zero shifted into most significant bit of most significant byte.

5. Shift Right One, Double and Insert

   Contents of "Shift" bit of Lower IR shifted into most-significant bit of most-significant byte.

6. Shift Right One, Double, Rotate

   Bit shifted out of least-significant bit of least-significant byte is shifted into most-significant bit of most-significant byte.

7. Shift Right One, Double, Sign Smear

   Most significant bit of shifted field is not changed.

NOTE: In all cases the "shifted out bit" is always saved in the Lower IR.

A8.3 SHIFT EIGHT, DOUBLE

This microinstruction causes an 8-bit shift, either left or right, to be executed on any of the following pairs of registers:

Accumulators A and B
Any even/odd pair of RBA Registers (R0/R1, R2/R3, etc.)

The Indicator Registers are not affected.

Microinstruction Options:

1. Shift Left Eight, Accumulator B

   Accumulator B is shifted into accumulator A; accumulator B is set to zero.

2. Shift Left Eight, RBA

The odd RBA register specified in the instruction is shifted into the adjacent even RBA register. The odd RBA register is set to zero.

3. Shift Right Eight, Accumulator A

Accumulator A is shifted into accumulator B; accumulator A is set to zero.

4. Shift Right Eight, RBA

The even RBA register specified in the instruction is shifted into the adjacent RBA register. The even RBA register is set to zero.

5. Exchange Accumulators

Accumulator A is exchanged with accumulator. B.

6. Exchange RBA

The even RBA register specified in the instruction is exchanged with the adjacent odd RBA register.

7. Exchange Special, Store A

Accumulator A is exchanged with accumulator B; accumulator A is also stored in the odd RBA register specified in the microinstruction.

8. Exchange Special, Store Upper RBA

The even/odd RBA register pair specified in the instruction are exchanged. In addition, the even RBA register is stored in accumulator B.

9. Exchange Special, Store B

Accumulator A is exchanged with accumulator B; accumulator B is also stored in the even RBA register specified in the microinstruction.

10. Exchange Special, Store Lower RBA

The even/odd RBA register pair specified in the instruction are exchanged. In addition, the odd RBA register is stored in accumulator A.

A9. TWO-BYTE LOAD/STORE OPERATIONS

A9.1 STORE AUXAR

This microinstruction causes the contents of the Auxiliary Address Register to be saved in the pair of registers specified.

Microinstruction Options:

1. Save AUXAR in RBA

   Contents of AUXAR are stored in the even/odd pair of RBA registers specified (R0/R1; R2/R3, etc.).

2. Save AUXAR in Accumulators

   Contents of AUXAR are stored in accumulators A,B.

For purposes of this microinstruction, the pair of registers specified to receive the contents of AUXAR are considered to be a single, 16-bit register. The 12-bit contents of AUXAR are stored in bits 3-14 of the register pair, with the three high-order bits and the least-significant bit of the register pair set to zero.

A9.2 STORE ROSAR

This microinstruction causes the contents of the Control Store Address Register to be saved in the pair of registers specified.

Microinstruction Options:

1. Save ROSAR in RBA

   Contents of ROSAR are stored in the even/odd pair of RBA registers specified (R0/R1; R2/R3; etc.).

2. Save ROSAR in Accumulators

Contents of ROSAR are stored in accumulators A,B.

For purposes of this microinstruction, the pair of registers specified to receive the contents of ROSAR are considered to be a single, 16-bit register. The 12-bit contents of ROSAR are stored in bits 3-14 of the register pair, with the three high-order bits and the least-significant bit of the register pair set to zero.

A9.3 LOAD ACCUMULATORS A,B

This microinstruction causes the contents of a specified pair of RBA registers to be loaded into accumulators A,B. The RBA pair specified must be an even/odd pair (R0/R1; R2/R3; etc.).

The contents of the even RBA register are loaded into accumulator A.

The contents of the odd RBA register are loaded into accumulator B.

Optionally, this microinstruction can cause the byte parity bits, as read out of the selected RBA registers, to be complemented before being stored in accumulators A,B. This gives the capability of loading A,B, with bad parity. A parity error is not detected on execution of this microinstruction, but will be detected on subsequent access of the data in A,B. This option is conditioned on the T&D mode (paragraph 2.1.1.2(3)) being set.

A9.4 STORE ACCUMULATORS A,B

This microinstruction causes the contents of the accumulators A,B to be stored into a specified pair of RBA registers. The RBA pair specified must be an even/odd pair (R0/R1; R2/R3, etc.).

Cont. on Page 92

The contents of accumulator A are stored into the even RBA register.

The contents of accumulator B are stored into the odd RBA register.

The contents of accumulator A can be optionally stored in the Branch Test Register.

A9.5 STORE TRACE REGISTER

This microinstruction causes the contents of the TRACE register to be saved in the pair of registers specified.

1. Save TRACE in RBA

   Contents of TRACE are stored in the even/odd pair of RBA registers specified (R0/R1; R2/R3; etc.).

2. Save TRACE in Accumulators

   Contents of TRACE are stored in accumulators A,B.

For purposes of this microinstruction, the pair of registers specified to receive the contents of TRACE are considered to be a single, 16-bit register. The 13-bit contents of TRACE are stored right-justified in the register pair, with the three high-order bits of the register pair set to zero.

A10. MAIN MEMORY OPERATION

A10.1 START READ MEMORY CYCLE

This microinstruction causes the initiation of a read cycle to the main memory. The contents of the addressed memory location are not obtained directly with this microinstruction, and must be transferred from memory to an MPC register with a subsequent Read Memory Data Register microinstruction.

The Start Read Memory Cycle microinstruction will specify the register source of the memory address, and will also specify the zone control to be used on the actual data readout.

Memory Address Source Options:

The microinstruction will allow several options in specifying the register source to be used as the memory address.

In the following list of options, reference to an RBA pair implies an even/odd RBA register pair (R0/R1; R2/R3; etc.), in which the contents of the register pair is considered to be a single 16-bit field, with the even RBA register representing the high-order 8-bits of the field.

Reference to accumulators A,B implies the contents of accumulators A and B are considered to be a single 16-bit field, with accumulator A representing the high-order 8-bits of the field.

1. Contents of the specified RBA pair are to be used as the memory address.

2. Contents of accumulators A,B are to be used as the memory address.

3. Contents of accumulators A,B are to be used as the memory address. In addition, the contents of A,B are to be stored in the RBA register pair specified in the microinstruction.

4. Contents of a specified RBA pair are to be used as the memory address.

   In addition, the contents of the odd RBA register (which represents the low-order eight bits of the address) are to be incremented by one and restored. The Lower Indicator Register is adjusted as defined for the add microinstruction.

5. Low-order eight bits of memory address determined by any of the above options, 1-5. The next most-significant bit (2^8 bit) is determined by the state of the Shared Memory Configuration Switch (see paragraph 2.3.5). The most-significant seven bits of the address are forced to zero.

   This option allows addressing a fixed 256 word memory block, represented by memory location 0-255, or 256-511, depending on the state of the switch.

Zone Control Options:

The microinstruction provides the following options in controlling how data read from the addressed memory location is to be restored in that memory location.

1. Memory word restored as read.
2. High-order byte of memory word restored as zero with good parity. Low-order byte restored as read.
3. Low-order byte of memory word restored as zero with good parity. High-order byte restored as read.
4. Entire memory word restored as zero, with good parity.

A10.2 START WRITE MEMORY CYCLE

This microinstruction causes the initiation of a write cycle on the main memory. The data to be written is not transferred with this microinstruction, but must be transferred to memory by the next sequential microinstruction, which must be a Write Memory Data Register microinstruction.

Cont. on Page 95

The Start Write Memory Cycle microinstruction will specify the register source of the memory address, and will also specify the zone control to be used on the subsequent write operation.

Memory Address Source Options:

The microinstruction will allow the same memory address source options as defined for the Start Read Memory Cycle microinstruction.

Zone Control Options:

The microinstruction provides the following options in controlling how data will be subsequently written into memory.

1. Only the low-order byte of the addressed memory location will be changed.

2. Only the high-order byte of the addressed memory location will be changed.

3. The entire addressed memory location will be changed.

A10.3 START DIAGNOSTIC MEMORY CYCLE

This microinstruction is used to start a pseudo memory cycle which allows access to internal memory registers for diagnostic purposes.

The microinstruction will specify the particular internal memory register to be accessed, and whether the register is to be written into or read. A subsequent Read or Write Memory Data Register microinstruction will accomplish the actual read or write operation.

An actual memory cycle is not initiated by the microinstruction.

This microinstruction will allow the same memory address source options as defined for the Start Read Memory Cycle microinstruction. However, since an actual memory cycle is not initiated by

Cont. on Page 96

this microinstruction, the memory address will be used only to determine which memory module is to be accessed, for the case where the MPC is connected to two memory modules.

A10.4 START SHARED MEMORY CYCLE

This microinstruction is used for those configurations where a common main memory is shared by two MPC's.

Execution of this microinstruction allows one MPC to set a "Shared Memory Connect" macro interrupt condition in the sharing MPC. The interrupt condition is detected as described in paragraph 2.1.2.3, Macro Interrupt.

This microinstruction has the following options:

1. Set Shared Memory Connect Interrupt in other MPC.
2. Reset Shared Memory Connect Interrupt in other MPC.
3. Set local Shared Memory Connect interrupt.
4. Reset local Shared Memory Connect interrupt.

A10.5 READ MEMORY DATA REGISTER

This microinstruction is used following a Start Read Memory Cycle microinstruction to transfer the memory data from the internal memory register to the specified MPC register(s). This microinstruction does not have to immediately follow the Start Memory Cycle microinstruction.

This microinstruction is also used following a Start Diagnostic Memory Cycle microinstruction to transfer diagnostic information to the specified MPC register(s).

This microinstruction will allow the following options in specifying the data transfer from the internal memory register to the MPC register(s);

1. High-order byte of memory data to any even RBA register (R0, R2, etc.).

2. Low-order byte of memory data to any odd RBA register (R1, R3, etc.).

3. High-order byte of memory data to any even RBA register and the Branch Test Register.

4. Low-order byte of memory data to any odd RBA register. High-order byte of memory data to the Branch Test Register.

5. Memory data (bits 0-15) to the specified even/odd RBA register pair (R0/R1, R2/R3; etc.).

6. Same as (5) except high-order byte of memory data is also placed in Branch Test Register.

7. High-order byte of memory data to accumulator A.

8. Low-order byte of memory data to accumulator B.

9. Memory data (bits 0-15) to accumulators A,B.

10. Memory data (bits 0-15) to the accumulators A,B; and also to the even/odd RBA register pair specified in the instruction.

11. Memory data (bits 0-15) to the internal register which normally holds the next odd microinstruction to be executed.

    This will result in the 16-bit memory data word being interpreted as the next odd microinstruction to be executed.

A10.6 WRITE MEMORY DATA REGISTER

This microinstruction must immediately follow a Start Write Memory Cycle microinstruction, and results in the transfer of the data to be written into read/write memory.

The microinstruction allows the following options in specifying the source register to be used for the data transfer. In all cases, a 16-bit data field is transferred.

Where RBA register pair is named, it must be an even/odd pair (R0/R1, R2/R3; etc.). The register pair is considered a 16-bit data field with the even RBA register containing the most-significant byte.

Where the accumulators A,B are named, they are treated as single 16-bit field, with accumulator A containing the most-significant byte.

1. Data transferred from any even/odd RBA register pair.

2. Data transferred from accumulators A,B.

3. Data transferred from accumulators A,B. Data is also stored in the RBA register pair specified in the microinstruction.

4. Data transferred from any even/odd RBA register pair. In addition, the contents of the odd RBA register are incremented and restored.

A11. INTERRUPT MECHANISM OPERATIONS

A11.1 LOAD INTERRUPT MECHANISM REGISTER

This microinstruction provides the capability of loading various registers associated with the interrupt mechanism and the Interval Timer.

The microinstruction allows the following options:

1. Load Interval Timer

   The contents of the A,B accumulators, bits 0-15, are loaded into the Interval Timer. Bit 0 must be loaded as "0," or an Interval Timer Overflow Condition will be generated.

2. Load Device Adapter Control Register

   The contents of accumulator B are loaded into the Device Adapter Control Register.

3. Load Device Adapter Number Register

   Bits 6,7 of accumulator B are loaded into the DA Number Register.

4. Load T&D Register

   The contents of accumulator B are loaded into the T&D Register.

5. Load Next Microinstruction from Accumulators

   The contents of the A,B accumulators are loaded into the internal register which normally holds the next odd microinstruction to be executed.

   This results in the establishment of the 16-bit field in accumulators A,B as the next odd microinstruction to be executed.

A11.2 STORE INTERRUPT MECHANISM REGISTER

This microinstruction provides the capability of storing various registers associated with the interrupt mechanism and the Interval Timer.

The microinstruction allows the following options:

1. Store Interval Timer

   The 16-bit contents of the Interval Timer are stored in accumulators A,B. (Note the high-order bit of the Interval Timer is the "runout" bit.)

2. Store Device Adapter Number Register

   The 2-bit contents of the DA Number Register are stored, right-justified, in accumulator B. High-order six bits of B are set to zero.

3. Store T&D Register

   The contents of the T&D register are stored in accumulator B.

4. Save Error Data Register

   The contents of the Error Data Register are saved in accumulators A,B. The Error Data Register is cleared to zero.

5. Store Interrupt Conditions

   The Macro Interrupt Conditions are stored in accumulator B. These conditions included the following:

EN-2 (one for each device adapter)	(Accumulator B)$_{0-3}$
Operator Interrupt Condition	(Accumulator B)$_{4}$
Shared Memory Connect	(Accumulator B)$_{5}$
Interval Timer Runout	(Accumulator B)$_{6}$
Manual Mode	(Accumulator B)$_{7}$

The microinterrupt conditions are stored in accumulator A_{0-7}:

$EN\text{-}1_{0-3}$, EN-1 Inhibit, Simulate EN-1, Inhibit Macro Interrupt, Dual Channel Mode Bit.

6. Save Configuration Switches

The state of the 16 configuration switches (see paragraph 4.2.8) are stored in accumulators A,B.

7. Save Data Switches

The state of the 16 data switches (see paragraph 4.2.7) are stored in accumulator A,B.

A11.3 CHANGE INTERRUPT MECHANISM CONDITIONS

This microinstruction allows the following interrupt and control conditions to be set or reset:

1. Inhibit all EN-1 Interrupts

When set, this condition will cause all EN-1 interrupt requests to be ignored.

2. Simulate EN-1 Interrupt

Setting of this condition will cause an artificial EN-1 interrupt request to be generated for all four DA ports. The interrupt request will exist until this condition is reset.

3. Inhibit all Macro Interrupts

When set, this condition will inhibit the detection of a Macro Interrupt request or Interval Timer overflow. The interrupt requests will be detected and executed when the inhibit condition is reset if still present.

The microinterrupt conditions are stored in accumulator A_{0-7}:

EN-1_{0-3}, EN-1 Inhibit, Simulate EN-1, Inhibit Macro Interrupt, Dual Channel Mode Bit.

6. Save Configuration Switches

The state of the 16 configuration switches (see paragraph 4.2.8) are stored in accumulators A,B.

7. Save Data Switches

The state of the 16 data switches (see paragraph 4.2.7) are stored in accumulator A,B.

A11.3 CHANGE INTERRUPT MECHANISM CONDITIONS

This microinstruction allows the following interrupt and control conditions to be set or reset:

1. Inhibit all EN-1 Interrupts

When set, this condition will cause all EN-1 interrupt requests to be ignored.

2. Simulate EN-1 Interrupt

Setting of this condition will cause an artificial EN-1 interrupt request to be generated for all four DA ports. The interrupt request will exist until this condition is reset.

3. Inhibit all Macro Interrupts

When set, this condition will inhibit the detection of a Macro Interrupt request or Interval Timer overflow. The interrupt requests will be detected and executed when the inhibit condition is reset if still present.

4. Simulate Error Interrupt (Set Only)

Setting of this condition will cause an artificial Error Interrupt to be executed. The Error Data Register will be set to indicate the interrupt was simulated.

This condition is reset automatically when the Error Data Register is read out during the Error Interrupt service.

5. Normal RBA Pointer

This condition controls which RBA half is addressed, when the MPC is in the Normal level of operation.

When this pointer is set, the "upper" RBA half is addressed by microinstructions.

The state of this pointer can be changed only when the MPC is operating in the "Normal" level of operation.

6. Dual Channel Mode

The state of this bit is used during interrupt service to determine in which RBA half high-level interrupt service for DA Port Number 1 is to take place. (See paragraph 2.1.3.2.)

7. Halt Mode (Set Only)*

When this condition is set, the MPC clock is stopped, and the MPC assumes a halt state. The condition is reset by Operator Panel switch.

8. Initialize (Set Only)*

When set, this mode causes the d-c initialization of the MPC, and then starts the MPC clock. The condition is automatically reset during the initialization operation.

*Can only occur if MPC is in the manual or T&D modes, or if an Error Interrupt is in process of being serviced.

A12. DEVICE ADAPTER INTERFACE OPERATIONS

A12.1 DEVICE ADAPTER INTERFACE DATA TRANSFER

This microinstruction controls the transfer of data between the MPC and any of the Device Adapter ports.

The microinstruction allows the following options:

DA Port to be Addressed

1. DA port defined by contents of Device Adapter Number Register.

2. DA port defined by "Link Adapter" field of DA Control Register.

Microinstruction Execution

1. Delayed until signal detected on RPI line from Device Adapter Interface.

2. Executed independent of RPI line.

Reading of DAI Data-In Lines

1. Data-in lines (0-15) are loaded into accumulators A,B.

2. Data-in lines (8-15) are loaded into accumulator B. (Single byte transfer.)

3. Data-in lines are not sampled.

4. Data-in lines (0-15) are loaded into the internal register which normally holds the next odd microinstruction to be executed.

   This results in the 16-bit data field from the data-in lines being interpreted as the next odd microinstruction to be executed.

8-Bit Address/Control Field

This immediate value field is contained in the microinstruction, and is transferred to the Address/Control bus of the DAI during the microinstruction execution.

DAI Status Lines

1. Information on the DAI Status lines is loaded into the Branch Test register.

2. DAI Status lines are not sampled.

Data-Out Lines

The data-out lines always reflect the current contents of accumulator A,B.

A13. BRANCH OPERATIONS

Because of the concurrent branch capability, all branch type microinstructions will physically reside at odd control store address locations, and will cause branches to even control store address locations.

A Branch microinstruction in an even control store location is treated as a NOP by the MPC.

A13.1 CONDITIONAL BRANCH

This microinstruction allows the testing of any bit of any of the following registers:

Upper Indicator Register
Lower Indicator Register
Accumulator A
Accumulator B
Branch Test Register

The microinstruction will specify whether the bit is to be tested for a "true" or a "false" condition. The condition tested will always be as it existed before the concurrent execution of the "even" address microinstruction associated with the "odd" branch microinstruction.

If the condition tested for is found to exist, a branch will be made to the even ROS location specified by the microinstruction. This even location can be anywhere within the 256 word control store segment in which the branch instruction is located.

If the condition tested for is found to not exist, the next sequential microinstruction is executed.

A13.2 DIRECT SEGMENT BRANCH

This microinstruction allows a direct branch to be made to any even location:

1. Within the 256 word control store segment in which the Branch microinstruction resides;
2. Within any one of the four 256 word control store segments immediately preceding the segment containing the Branch microinstruction;
3. Within any one of the three 256 word control store segments immediately following the segment containing the Branch microinstruction.

Direct Segment Branch and Save ROSAR

This microinstruction is identical to the Direct Segment Branch microinstruction with the exception that before executing the branch the current contents of the control store Address Register (ROSAR) are saved in the Auxiliary Address Register (AUXAR).

The current contents of ROSAR is defined to mean the even control store address which is the address of the even/odd microinstruction pair containing the Branch instruction.

A13.3 VECTOR SEGMENT BRANCH

This microinstruction allows an indexed branch to be made into a microinstruction table.

The table may be located anywhere within the 256 word control store segment containing the Vector Segment Branch microinstruction. The microinstruction will contain the segment address of the table, which must be even.

The index into the table is provided by the contents of a selected register field, as specified by the microinstruction.

The table may be either four even/odd microinstruction pairs long, or 16 even/odd microinstruction pairs long. The binary value of the selected register field is used to determine the even/odd microinstruction pair of the table to be branched to. The branch is effectively made to the even microinstruction of the even/odd pair.

The following register fields may be called out by the microinstruction to be used as the index into the specified table:

Branch Test Register,	bits 0-3
Branch Test Register,	bits 4-7
Branch Test Register,	bits 0-1
Branch Test Register,	bits 2-3
Branch Test Register,	bits 4-5
Branch Test Register,	bits 6-7
Interrupt Mechanism Status,	bits 0-3

A13.4 ABSOLUTE BRANCH

The execution of this microinstruction causes a branch to be made using the contents of a specified register as the absolute branch address.

The following registers can be specified as containing the absolute branch address:

1. AUXAR

If the 12 bits of AUXAR are designated 3-14, the absolute branch address is generated as follows:

$$\left[(\text{AUXAR})_{3-14},\right] 0 + 2 = \text{absolute branch address}$$

A carry across byte boundary is not propagated (bit 8 to bit 7).

2. Accumulator A/Immediate Value from Microinstructions

The contents of accumulator A are combined with an immediate value field from the branch microinstruction to form the absolute branch address as follows:

$(A)_{3-7}$, (7-bit immediate value), 0 = absolute branch address

3. Accumulators A,B

If the 16 bits of accumulators A,B, treated as one 16-bit field, are designated 0-15, the absolute branch address is formed as follows:

$(A,B)_{3-14}$, 0 = absolute branch address

4. Operator Panel Address Switches

With the 16 address switches designated as 0-15, the absolute branch address is formed as follows:

$(\text{Address Switches})_{3-14}$, 0 = absolute branch address

The microinstruction also allows the concurrent execution of one of the following options:

1. Safestore contents of ROSAR in AUXAR before executing the absolute branch.

2. Reset the current interrupt level in progress.

3. Execute both options (1) and (2).

*NOTE: In the event the interrupt level is reset by the execution of the absolute branch, an absolute branch address generated from the contents of AUXAR is not incremented by 2.

COMPANY CONFIDENTIAL

GENERAL ELECTRIC
Information Systems Equipment Division
Phoenix, Arizona

NUMBER: 43A177875

EPS-1 Microprogrammed Peripheral Computer

Cont. on Sh. - Sh. No. 108F

SIGN-OFF SHEET

Prepared by: Roy Johnson 6/12/70
R. Johnson Date
Advanced Systems Engineering

Approved by: C W Dix 6/26/70
C. W. DIX, Manager Date
PED Engineering

Reviewed by: CA Conover 6/25/70
C. A. Conover, Manager Date
Advanced Systems Engineering

Reviewed by: 6/23/70
F. D. Strout, Manager Date
Advanced Subsystems Design

Reviewed by: MJ Tobias 6/25/70
M. J. Tobias, Manager Date
Advanced Systems Design

Reviewed by: L. I. Wilkinson 6/26/70
L.I. Wilkinson, Manager Date
Systems Engineering

Reviewed by: 6-25-70
P.J. Scola, Manager Date
Systems T&D Engineering

Reviewed by: 6/26/70
G.R. Williams, Manager Date
Systems Design Engineering

Reviewed by:
R.F. Stevens, Manager Date
Operating Systems Engineering

Reviewed by: 6/25/70
G.B. Krekeler, Manager Date
Lang. & Data Mgmt. Sys. Eng.

Approved by: 25/70
C.F. RAINE, Manager Date
DPO Systems & Product Planning

Reviewed by: 6/26/70
J.D. Hann, Manager Date
Systems & Software Planning

Reviewed by:
D.E. Booth, Manager Date
Hardware Product Planning

Reviewed by: 6/26
R.L. Ruth, Manager, Date
Computer Design Engineering

Reviewed by:
R.L. Mynatt, Manager Date
APL Design Engineering

Reviewed by:
HAAA Epperson, Manager Date
Memories & Circuits Engineering

Reviewed by:
P.G. Smee, Manager Date
Product Engineering

Reviewed by: R F Marshall 6/26/70
R.F. Marshall, Manager Date
Systems Integration Eng.

Reviewed by: 6/23/70
K.A. Johnson, Manager Date
Reliability Engineering

GENERAL ELECTRIC

COMPANY CONFIDENTIAL INFORMATION
(ISG - Class 3 Restricted)

This document shall not be reproduced in whole or in part except as may be necessary for the conduct of the internal business of the Information Systems Group and shall not be disclosed outside of the Company without Department General Manager approval with legal concurrence.