

Signal Processing Element Users' Reference Manual

WILLIAM R. SMITH AND JOHN P. IHNAT

*Information Systems Group
Office of Associate Director of Research for Electronics*

September 5, 1972



**NAVAL RESEARCH LABORATORY
Washington, D.C.**

REPORTS IN THIS SERIES

**"Microprogramming Control Unit (MCU)–Programming Reference Manual,"
John D. Roberts, Jr., NRL Report 7476, Aug. 15, 1972**

CONTENTS

Abstract	iii
Problem Status	iii
Authorization	iii
1. INTRODUCTION	1
1.1 Functional Description	1
2. MCU ORGANIZATION	2
2.1 Control Store	2
2.1.1 Addressing	2
2.1.1.1 Control Store Address Register (CSAR)	2
2.1.1.2 Alternate Control Store Address Register (ACSAR)	2
2.1.1.3 Command Register (CR)	5
2.2 Arithmetic and Logic Unit (ALU)	5
2.3 Data Storage and Manipulation	5
2.3.1 Z Register	6
2.3.2 Local Store A (LSA) and Local Store B (LSB)	6
2.3.3 Buffer Address Register A (BARA) and Buffer Address Register B (BARB)	7
2.3.4 Field Select Data Register (FSDR)	7
2.3.5 Shift Amount Register (SAR)	7
2.3.6 Counter (CTR)	7
2.4 Interrupt Control Unit (ICU)	8
2.4.1 Interrupt	8
2.4.2 Register Pushdown	8
2.4.3 Interrupt Return	9
3. BUFFER MEMORIES	9
4. STORAGE CONTROL UNIT (SCU)	9
4.1 Buffer Request	9
5. INPUT/OUTPUT	9
5.1 Organization	9
5.2 Unbuffered Channel	10

5.2.1	Z Bus	10
5.2.2	MCU-to-Z-Bus Interface	12
5.2.3	Unbuffered I/O Operation	13
5.3	Buffered Channel Controller	14
5.3.1	SCC Command and Format	14
5.3.2	SCC Status Information and Format	15
6.	SIGNAL PROCESSING ARITHMETIC UNIT (SPAU)	16
6.1	SPAU Organization	16
7.	MCU MAINTENANCE PANELS	16
7.1	MCU Switches	16
7.2	SCU Switches	20
7.3	MCU Indicators	20
7.4	SCU Indicators	21
8.	MCU MICROINSTRUCTIONS	21
8.1	Control Field Summary	22
8.2	Microinstruction Operations	22
8.2.1	Instruction Addressing and Sequencing	22
8.2.2	ALU Operations	23
8.2.2.1	ALU Input Selection	23
8.2.2.2	ALU Function	23
8.2.2.3	ALU Result Destination	24
8.2.3	Local Store and Buffer Addressing	24
8.2.3.1	Local Store Addressing	24
8.2.3.2	Buffer Store Addressing	24
8.2.4	Buffer Operation	25
8.2.5	Interrupt Operation	25
8.2.6	Input/Output Operation	25
8.3	Control Field Definitions	26
8.3.1	CØND Field	26
8.3.2	NØT Field	26
8.3.3	NEXT Field	26
8.3.4	SAUX Field	27
8.3.5	WBUF Field	27
8.3.6	FSU Field	28
8.3.7	SARA Field	28
8.3.8	SARB Field	28
8.3.9	SLSA Field	28
8.3.10	SLSB Field	29
8.3.11	ADIN Field	29
8.3.12	ADØP Field	30
8.3.13	SAAL Field	30
8.3.14	SBAL Field	30
8.3.15	DAAL Field	31
8.3.16	DBAL Field	31
8.3.17	LIT Field	31
9.	ACKNOWLEDGMENT	32

ABSTRACT

The NRL Signal Processing Element (SPE) is a high-performance signal processing facility for radar, sonar, and communication systems. It is intended to be compatible with the Navy All Applications Digital Computer (AADC). The SPE consists of four major subsystems: a Microprogrammed Control Unit (MCU), a Buffer Store and Storage Control Unit (SCU), a Signal Processing Arithmetic Unit (SPAU), and Input Output (I/O) units.

The MCU serves as system supervisor and data organizer for the SPAU and other I/O devices. The MCU includes a 64-bit Control Store, two local stores, an arithmetic element, two busses to buffer memory, an unbuffered byte channel, and a priority interrupt system. Its cycle time is 150 nsec.

The buffer store and SCU consists of eight fixed-priority Direct Memory Access (DMA), or Buffered Channels, communicating on a 150-nsec cycle basis with up to eight 32-bit by 4K memories.

The SPAU is a highly parallel, high-speed arithmetic unit capable of performing complex signal processing operations such as FFT and recursive filtering. It is microprogrammed for flexibility in adapting signal processing algorithms. Four multiplies and six additions can be performed in 300 nsec.

SPE I/O and internal communications are provided by DMA buffer data channels, an unbuffered byte channel, and by a priority interrupt system. The unbuffered byte channel called the Z bus communicates both data and control information to all I/O devices. The unbuffered channel burst data rate is 2 million 16-bit words per second.

PROBLEM STATUS

This is an interim report on one phase of the problem; work on this and other phases is continuing.

AUTHORIZATION

NRL Problems B02-06 and B02-10
Projects NAVAIR WF 15-241-601 and NAVELEX XF 21-241-015-K152

Manuscript submitted August 21, 1972

SIGNAL PROCESSING ELEMENT USERS' REFERENCE MANUAL

1. INTRODUCTION

The NRL Signal Processing Element (SPE) is a high-performance signal processing facility for radar, sonar, and communication systems. The design of the SPE provides for efficient, flexible solutions to problems suited to digital signal processing machines. The SPE is intended to be compatible with the Navy All Applications Digital Computer (AADC) system now under development.

The SPE consists of the following elements:

- Microprogrammed Control Unit (MCU)
- Buffer memories
- Storage Control Unit (SCU)
- Input/output system
- Signal Processing Arithmetic Unit (SPAU).

The SPE elements are implemented with "off-the-shelf" components. Bipolar monolithic storage devices and TTL Schottky family logic are used. Performance specifications include:

MCU basic microinstruction	150 nsec
Buffer memory cycle	150 nsec
SPAU-equivalent complex operation (four multiplications and six adds)	300 nsec

Performance is compatible with projected AADC technology, and efficient operation can be expected under stand-alone or system-integrated conditions.

1.1 Functional Description

The SPE is designed as a tool for processing digital data streams. The heart of the SPE is the MCU which serves as system supervisor and data organizer for the SPAU and other I/O devices in the system. Microprogrammed operations in the MCU process 16-bit-wide data accessed from 32-bit-wide buffer memories and control buffered and unbuffered I/O operations to and from SPE devices.

The SPAU performs special data processing operations such as FFT, recursive filtering, and correlation under direction of the MCU. Parallel organization of fast multiply and add logic units allow for high-speed execution of these functions. Interfacing between the SPAU and MCU is via buffer memories and the I/O system.

It is the responsibility of the SCU to manage accesses to buffer memories by the elements of the SPE. The MCU, SPAU, and other buffered devices in the system access buffer memories independently under their own control, and the SCU resolves conflicts for buffer cycles on a priority basis.

The I/O system is designed to allow expansion of the SPE so that multiple MCU's and SPAU's can communicate and coordinate processing of increased data bandwidths.

Figure 1 is a block diagram of the SPE.

2. MCU ORGANIZATION

The MCU is composed of four major sections. These are

- Control Store (CS)
- Arithmetic and Logic Unit (ALU)
- Associated registers and local stores
- Interrupt control unit.

Figure 2 is a block diagram of the MCU.

2.1 Control Store

The MCU control store is a read/write memory with a 150-nsec cycle time. The basic memory size is 1024 words by 64 bits, with expandability to 4096 words by 64 bits. The CS holds 64-bit microinstructions which control the operation of the MCU. Input to the CS is via an independent buffered channel by which block loads of microprogram can be made from an external bulk store. CS output is to a 64-bit command register (CR) which holds the microinstruction under execution. The address of the microinstruction to be fetched is provided by the Control Store Address Register (CSAR).

2.1.1 Addressing

The CS is directly addressed by the CSAR. The CSAR and the rest of the addressing hardware is 16 bits wide to allow for virtual addressing of CS space of up to 64K words. With the virtual address translation mechanism present in the MCU, an address to a 1024-word block not in CS will cause a load of that block, via the interrupt system, into the CS. Without virtual readdressing, the upper address bits which are outside the range of the CS have no effect on MCU operation.

The registers used for CS addressing are described in the following paragraphs.

2.1.1.1 Control Store Address Register (CSAR)

The CSAR is the primary address source for the CS. It is the only MCU register connected directly to the CS address inputs. The CSAR is a counter as well as a register and provides the incrementing function needed for normal address sequencing. The CSAR can be set directly from a 16-bit Literal field in the command register (CR), from the ACSAR, from the Z register or by the interrupt control unit in the event of interrupt operations.

2.1.1.2 Alternate Control Store Address Register (ACSAR)

The ACSAR is used to save the contents of the CSAR for subroutine returns or loop returns, or as a general means of storing and restoring the contents of the CSAR. The ACSAR can be set from the CSAR, from the CR Literal field or from the Z register.

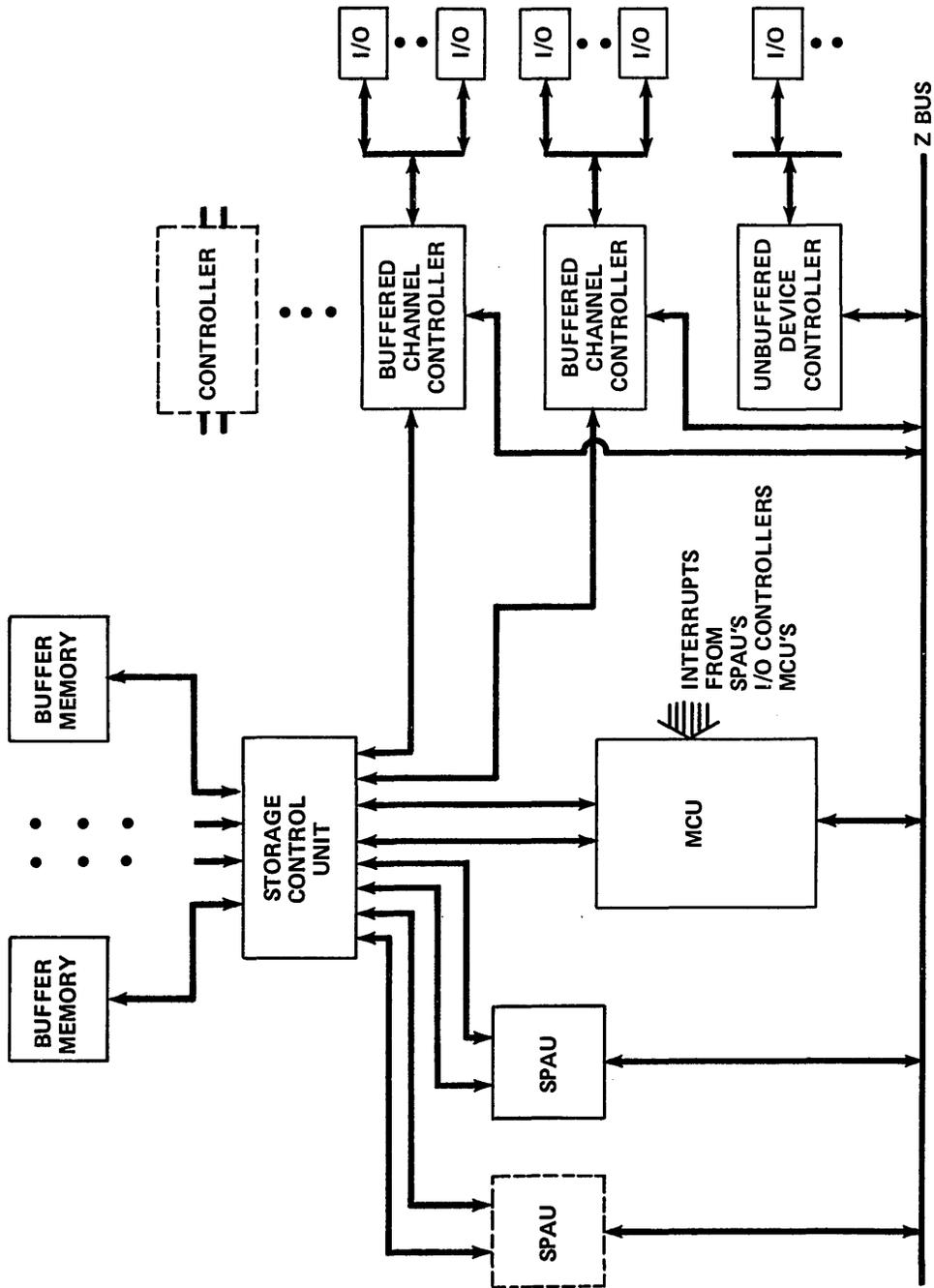


Fig. 1 - SPE system

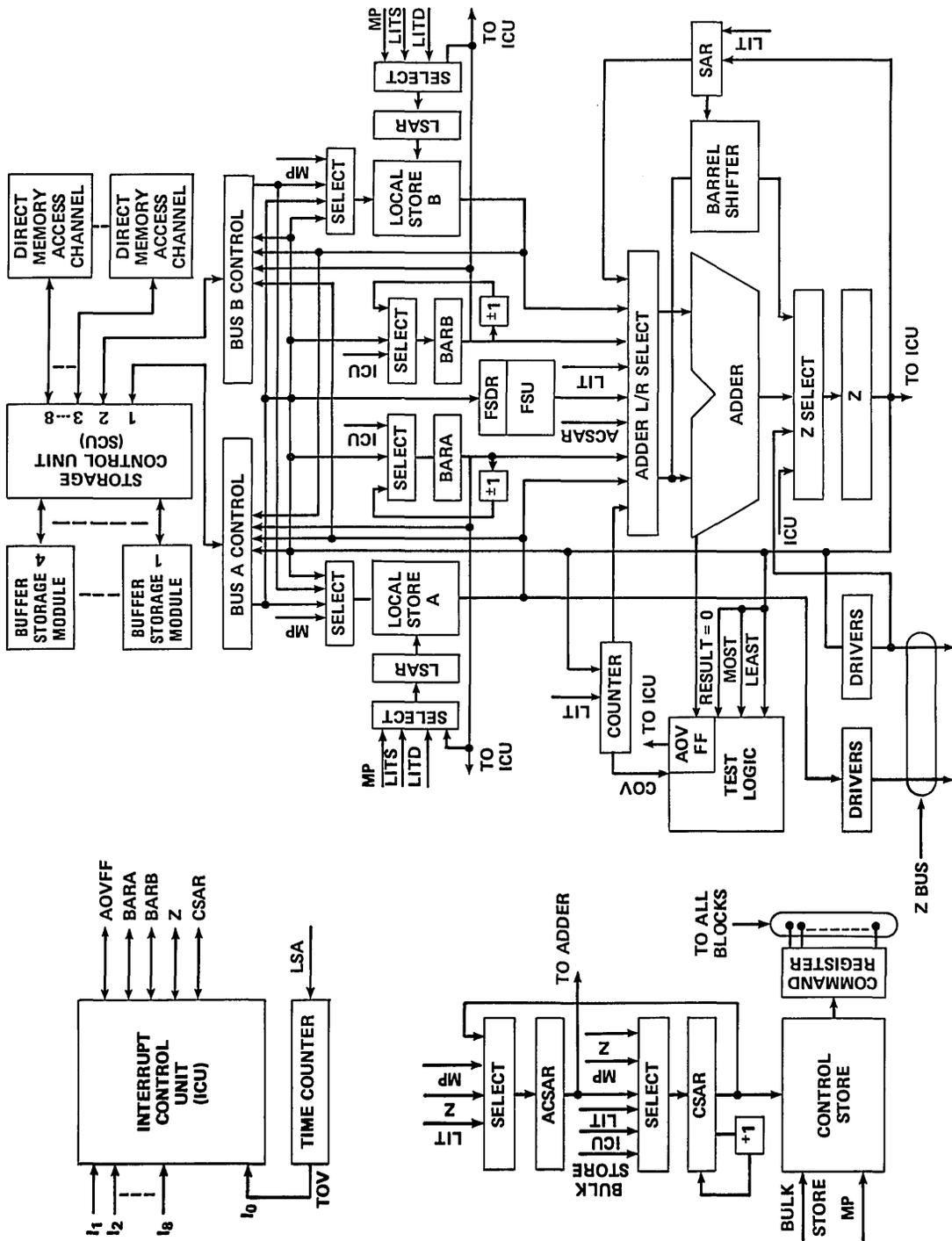


Fig. 2 - MCU

The ACSAR output can be gated to the CSAR for address restoring or to the left input of the ALU for buffer storage or for processing.

2.1.1.3 Command Register (CR)

The CR is not considered a part of the CS address mechanism but does provide certain outputs for control of address operations. The 16-bit Literal field of the CR can be gated to the CSAR or the ACSAR. Three other fields of the CR specify address successor and conditions for address operations and are described in Section 8 of this manual.

2.2 Arithmetic and Logic Unit (ALU)

The ALU provides the arithmetic, logic, and shifting capabilities of the MCU. This unit can perform a binary operation on two 16-bit words, one on the left input (L) and one on the right input (R) or unary operations on a single input.

The operation performed is specified by the ADOP field of the instruction word; the output is delivered to the 16-bit Z register which is the result register and intermediate buffer for gating ALU results to other MCU registers or to the I/O system. An ALU operation always sets Z but it is not necessary to transfer the contents of Z to any other place on any instruction.

The contents of Z and the ALU overflow register are made available to the test logic for instruction sequencing.

The ALU provides the following operations:

NOP	Z unchanged
L+R→Z	
L-R→Z	
L+1→Z	
L-1→Z	
L →Z	
\bar{L} →Z	One's complement
R →Z	
O →Z	Zero
L OR R→Z	Bit logical OR
L AND R→Z	Bit logical AND
L XOR R→Z	Bit logical Exclusive OR
L EQU R→Z	Bit logical Equivalence
Left shift L	Zero fill
Right shift L	Zero fill
Left Circular Shift L	

Arithmetic operations are in two's complement arithmetic.

Shift amounts are controlled by the SAR register.

2.3 Data Storage and Manipulation

The MCU design provides for two local storage memories and four registers intended for working data or address manipulation. Two other registers are provided for shift control and counting operations.

The primary data stores are local stores, LSA and LSB, and result register Z.

Registers BARA and BARB are intended primarily as address registers for buffer memories and local stores, but are not limited to that use.

FSDR is the only 32-bit register in the MCU organization. It receives 32-bit words direct from buffer memory for outputting to the Field Select Unit (FSU). The FSU is described later in this section.

The SAR register controls ALU shift operations and the CTR register is a counter intended for general loop or index control.

The following paragraphs describe the characteristics, operation, and uses of the data stores and registers.

2.3.1 Z Register

The Z Register is a 16-bit register intended primarily to receive the output of the ALU. It is also connected to the I/O unbuffered channel, but this function is controlled strictly by I/O commands. I/O organization and operation are described in a later section of this manual.

Z has outputs to other registers and to the memory as well as to the test logic and the I/O bus. The output from Z is not limited to a single register or memory in any one instruction but may be directed to a number of destinations simultaneously. This same ability is provided for outputs from other stores or registers.

The Z register is not itself a source to the ALU but acts as a buffer register between the ALU output and the destinations to which the result is directed. ALU input select, ALU operation, Z set, and destination set can all occur in a single instruction.

2.3.2 Local Store A (LSA) and Local Store B (LSB)

LSA and LSB are working storage files which can be addressed from the Command Register or from the buffer address registers. Each local store contains 16 words of 16 bits.

LSA and LSB serve as fast, addressable buffers between the buffer memories and the rest of the MCU. Each local store is connected to each of the two buffered memory channels serving the MCU/SCU interface. The high speed of the local stores (30-nsec access) allows efficient use of the fast ALU; each local store can be both inputs to and outputs from the ALU in a single instruction cycle.

Each local store has a 4-bit address register which can be loaded with literal addressing fields in the instruction word or with the contents of a buffer address register. Double address capability has been provided whereby different local store addresses can be specified for read and write operations in the same instruction cycle.

LSA output is connected to the I/O unbuffered channel to provide a source of computed I/O commands.

2.3.3 Buffer Address Register A (BARA) and Buffer Address Register B (BARB)

BARA and BARB are multipurpose address/data registers. They are 16-bit registers intended primarily to serve as address sources for buffer memory operations. Besides increment and decrement operations, the BAR's can be set directly with the ALU output. When used as a buffer address source, the three high-order bits specify the particular buffer module being addressed. The next 12 bits allow for addressing up to 4096 words of 32 bits each, and the least significant bit specifies the upper or lower 16-bit half of the memory word.

When used as a source for local store addressing, the four lower order bits of a BAR specify the local store location.

The BAR's have outputs to the ALU and can therefore be used as working registers.

The BAR's are not meant to be limited to a single function in any instruction cycle, and it is possible to specify a BAR for buffer memory and local store addressing as well as an ALU operation as long as the register contents are consistent with the addressing and data requirements.

2.3.4 Field Select Data Register (FSDR)

The FSDR is a 32-bit register whose input is a full 32-bit word from buffer memory. The FSDR outputs to the Field Select Unit (FSU) which outputs in turn to the ALU. The function of the FSU is to select a specified field of 16 bits or less from the FSDR, to right justify that field, and to output the result to the ALU.

The FSDR-FSU combination allows efficient unpacking of bytes or other field arrangements from full 32-bit words.

2.3.5 Shift Amount Register (SAR)

The SAR is a 4-bit register whose contents control the length of a shift operation specified in the ALU operation field of an instruction word.

The SAR can be set with constants from the lower four bits of the Literal field of an instruction word or by the lower four bits of the Z register.

The SAR has an output to the ALU for the purpose of operating on its contents or for saving them in memory.

2.3.6 Counter (CTR)

The CTR is a 16-bit counter register intended to be used to control indexing and counting operations in the MCU. The CTR can be set from the instruction word Literal field or from the Z register and is automatically loaded with the one's complement of the specified data.

The CTR is connected to the test logic of the MCU and is used to control Control Store successor operations. One test condition of the Address Successor Condition field is CTR Overflow. Each time this test is performed the counter is also incremented by one, and, if the register is full, the next Control Stores address will be that specified by

the CS Address Successor field of the same instruction word. The number of counter tests before overflow occurs is equal to the counter setting.

CTR has an output to the ALU for the purpose of saving its contents in memory.

2.4 Interrupt Control Unit (ICU)

The interrupt lines are connected such that an interrupt on one of these lines will be automatically recognized if its priority level is higher than the level at which the MCU is operating. Seven/fifteen interrupt lines are available in the MCU. Some of these lines are connected so as to allow internally generated interrupts such as address faults or I/O clock time-outs. The rest are available for use by external devices and channels.

Except for the Interrupt Return Jump code in the Control Store Address Successor field of the instruction word, the user has no control over the operation of the interrupt mechanism in the MCU.

2.4.1 Interrupt

The priority of each interrupt line is determined by its physical location at the MCU. Eight priority levels are built in. An interrupt line must be held up for at least one MCU cycle or 150 nsec in order for the interrupt to be recorded by the MCU in a special interrupt register. The recording of the interrupt does not imply its immediate recognition by the MCU unless the interrupt priority level is greater than that level being processed by the MCU. The MCU automatically sets its priority to the level of the interrupt being processed, and lower priority interrupts must wait until all recorded interrupts of higher priority have been serviced. This arrangement of unconditional interruption by higher priority requests provides for fast, low-overhead interrupt management.

When an interrupt is recognized, an address is forced into the Control Store Address Register by the ICU. This address will be equal to the number of the interrupt line at the MCU interface. Thus, the lower seven/fifteen locations of control store will normally be programmed to act as a jump table to the routines serving the different interrupt lines.

2.4.2 Register Pushdown

Since interrupt recognition is out of program control, certain MCU register contents are automatically saved in a small memory stack each time an interrupt is recognized. These registers are CSAR, Z, BARA, and BARB. Also, the ALU overflow status and the MCU priority level are saved in this memory. Saving and restoring of any other registers or storage locations are the responsibility of the programmer. Paths are provided between buffer storage and all MCU registers via the ALU for this purpose.

Following the saving of its contents by the ICU, the Z register is automatically loaded with the contents of the CSAR. This makes available to the programmer the address of the instruction about to be executed when the MCU was interrupted. This address is made available primarily for the purpose of servicing virtual-address-fault interrupts and may be ignored by the programmer if desired.

2.4.3 Interrupt Return

The last instruction of any interrupt routine will always include a jump-return-from-interrupt. This causes the information saved at the last interrupt to be restored into the MCU registers. The restoration of CSAR causes the MCU to resume processing with the instruction about to be executed when the interrupt occurred. Restoration of priority will always be to a lower level since only interrupts of higher priority than those in process can be recognized.

3. BUFFER MEMORIES

An SPE can have a maximum of eight buffer memories. Each buffer memory consists of up to 4096 words of 32 bits and has a separate 32-bit data port. The buffer memories use static, bipolar, monolithic storage devices which are compatible with TTL logic. The read/write cycle time is 150 nsec. The memories are contained on printed circuit boards which are placed in 19-in.-wide panel racks.

Each buffer memory is independently accessible through its own port. MCU's, SPAU's, and peripheral devices must contend for buffer memory access on a cycle-by-cycle basis. It is the responsibility of the Storage Control Unit (SCU) to resolve memory access conflicts.

4. STORAGE CONTROL UNIT (SCU)

All SPE devices (MCU's, SPAU's, peripherals) which require buffer memory access are interfaced to the memories through the SCU. The SCU can interface up to eight data channels with up to eight buffer memories. Any channel may access any buffer memory at any time. Whether or not the buffer cycle which is requested is granted depends on the priority of the requesting channel and the state of the other channels. Channel priority is hard wired and determined by the physical location of the channel at the SCU.

4.1 Buffer Request

Requests for a buffer cycle are made by a device raising a buffer request line along with the buffer address lines. The SCU records all buffer requests every clock cycle (150 nsec) and returns a Request Granted line to each device receiving its requested buffer cycle. If two or more devices request the same buffer on the same cycle, only the highest priority channel will receive the Request Granted line.

It is the responsibility of any device to remain idle pending a positive response by the SCU to its buffer cycle request. This is done automatically by the MCU and SPAU.

5. INPUT/OUTPUT

5.1 Organization

There are two types of input/output channels in the SPE: Direct Memory Access (DMA) buffered channels and a single unbuffered channel called the Z bus. Eight/sixteen buffered channels enable high-speed data transfer between buffer memories and system devices or MCU's. The Z bus allows direct communication under MCU control and on a word-by-word basis between the Z register of an MCU and all devices connected to the Z bus. The Z bus also enables direct MCU-to-MCU communication.

Figure 3 shows an SPE configuration with I/O system elements and interconnections.

All buffered channel communications pass through the Storage Control Unit (SCU). It is the responsibility of the SCU to manage buffer memory requests originating from MCU's, SPAU's, and system channel controllers. Devices which access buffer memory over buffered channels are interfaced to the buffered channels by Selector Channel Controllers (SCC). SCC's also interface with the Z bus and are responsible for interpreting device requests coming over the Z bus from MCU's. These requests originate in the form of MCU I/O instructions and can call upon an SCC to initiate various device I/O operations over its buffered channel interface.

The SCC's are intended to be standard I/O elements interfacing between buffered channels and Device Controllers (DC). DC's interface between SCC's and I/O devices and must be tailored to meet the interface requirements of a particular device type. DC's interface to SCC's over Z-bus-compatible connections. This allows a DC to connect directly to the Z bus for direct unbuffered communication with an MCU or to connect to an SCC for buffered channel communication.

SCC's and DC's can request MCU action via interrupt lines provided in the MCU's for such purposes. Separate SCC's or DC's sharing a single interrupt line must have hardware to resolve competition among the units for interrupt service.

An MCU generating an I/O request addressed to another MCU for the purpose of MCU-to-MCU communication causes the addressed MCU to raise an internal interrupt line. An I/O acknowledge instruction by the interrupted MCU completes the data transfer over the Z bus.

5.2 Unbuffered Channel

The MCU exchanges commands and unbuffered data with devices over a bidirectional, byte-multiplexed channel called the Z bus. The channel can be interrupt driven with information transferred from/to the Z register upon execution of an I/O instruction. There is but one Z bus regardless of the number of MCU's a system may contain. MCU's are in charge of Z-bus usage and resolve Z-bus access among themselves on a first-come, first-served basis.

Devices wishing Z-bus access must alert the MCU via an interrupt. A single interrupt line may serve many devices via a Device Controller which must resolve simultaneous requests for service. The interrupt line is deactivated upon transmission of an I/O command by the servicing MCU to the interrupting device's controller. Different interrupt lines servicing different sets of devices are subject to the priority resolution scheme built into the MCU interrupt mechanism, priority being determined by the position of the interrupt line on the MCU. The maximum burst transfer rate over the Z bus, based upon an MCU cycle time of 150 nsec, is 2 MHz.

5.2.1 Z Bus

The Z bus consists of 30 lines, 16 of which are bidirectional data lines. The remaining 14 are control lines as follows:

- a. Eight Device Address Lines (bidirectional)
- b. Input/Output Line (output)
- c. Data/Control Line (output)
- d. Continue Line (output)

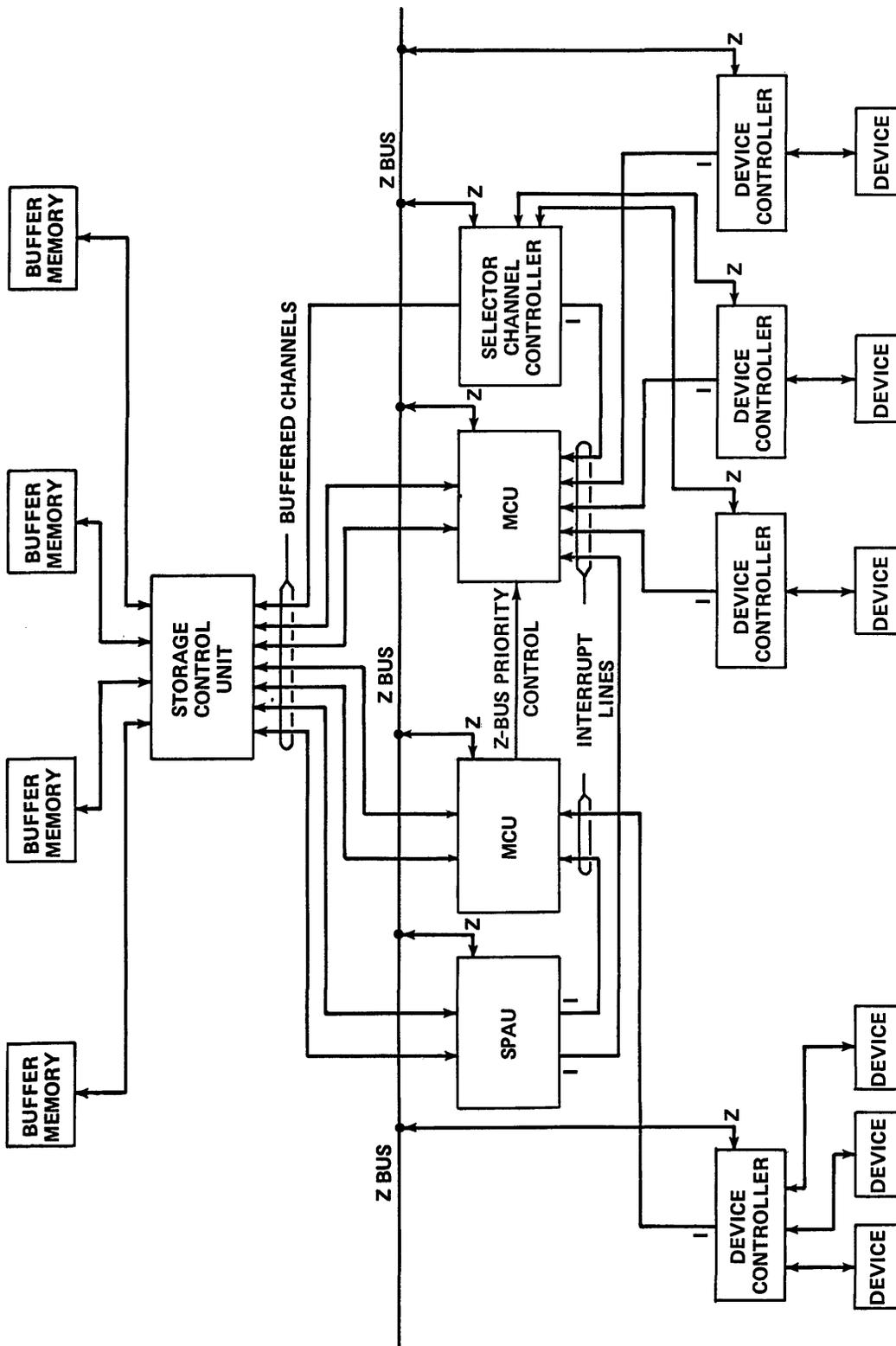


Fig. 3 - SPE configuration illustrating the I/O system

- e. Command Line (bidirectional)
- f. Acknowledge Line (bidirectional)
- g. Reject Line (input).

The Device Address Lines alert a specified device controller or channel controller to connect to the Z-bus data lines for an information transfer.

The Input/Output Line specifies the direction of information transfer.

The Data/Control Line specifies whether the transfer will involve data or control information. If control information is specified, an Input command will bring a status word from the device to the MCU; an Output command will send a command word from the MCU to the device.

The Continue Line is used to specify, where appropriate, that a device be prepared to accept another I/O word as continuation of the present I/O.

The Command Line alerts all devices on the Z bus that an I/O command has been put on the bus.

The Acknowledge Line is raised by a device to acknowledge to the MCU that the device has accepted the I/O command.

The Reject Line is raised by a device to notify the MCU that the device cannot accept the I/O command.

The Device Address Lines, Command Line, and Acknowledge Line take part in MCU-to-MCU communication as well as MCU-to-device communication and are therefore bidirectional. An MCU continually monitors the Command Line and Device Address Lines for an I/O command directed at it by another MCU. An MCU can raise the Acknowledge Line in response to an I/O command in the same manner as a device.

A separate line called the Z-Bus Priority Control Line passes from one MCU to another in a multiple-MCU system. This line allows each MCU, if it does not wish the Z bus, to pass on bus-access privilege to the next MCU in line in decreasing priority.

5.2.2 MCU-to-Z-Bus Interface

The MCU interfaces with the Z bus through its Z register and Local Store A (LSA). The Z-bus data lines input to and are driven from the Z register. The Z register must be loaded with data or command information prior to an I/O output command. An I/O input operation loads the Z register from the Z bus for subsequent use.

A control word must be set up in LSA prior to the I/O command. The LSA location containing the control word is specified by the LSA Address Field, and 12 bits of the selected word directly drive 12 control lines of the Z bus as shown in Fig. 4. The Acknowledge Line is driven indirectly through I/O control logic. The Reject Line inputs to the MCU Sequence Unit and causes a program jump to the Control Store location specified by the Literal field contents in the I/O command word.

Four bits of the command word set up in LSA do not go out over the Z bus but go to a clock time-out counter in the MCU. These bits allow the programmer to specify a time interval in powers of 2 from 2^0 to 2^{15} clock cycles. This interval limits the time that the MCU will wait inhibited in the I/O state for the return of an Acknowledge signal from an addressed device. In the event of a time out, an internal interrupt is generated and the MCU leaves the I/O state to process that interrupt.

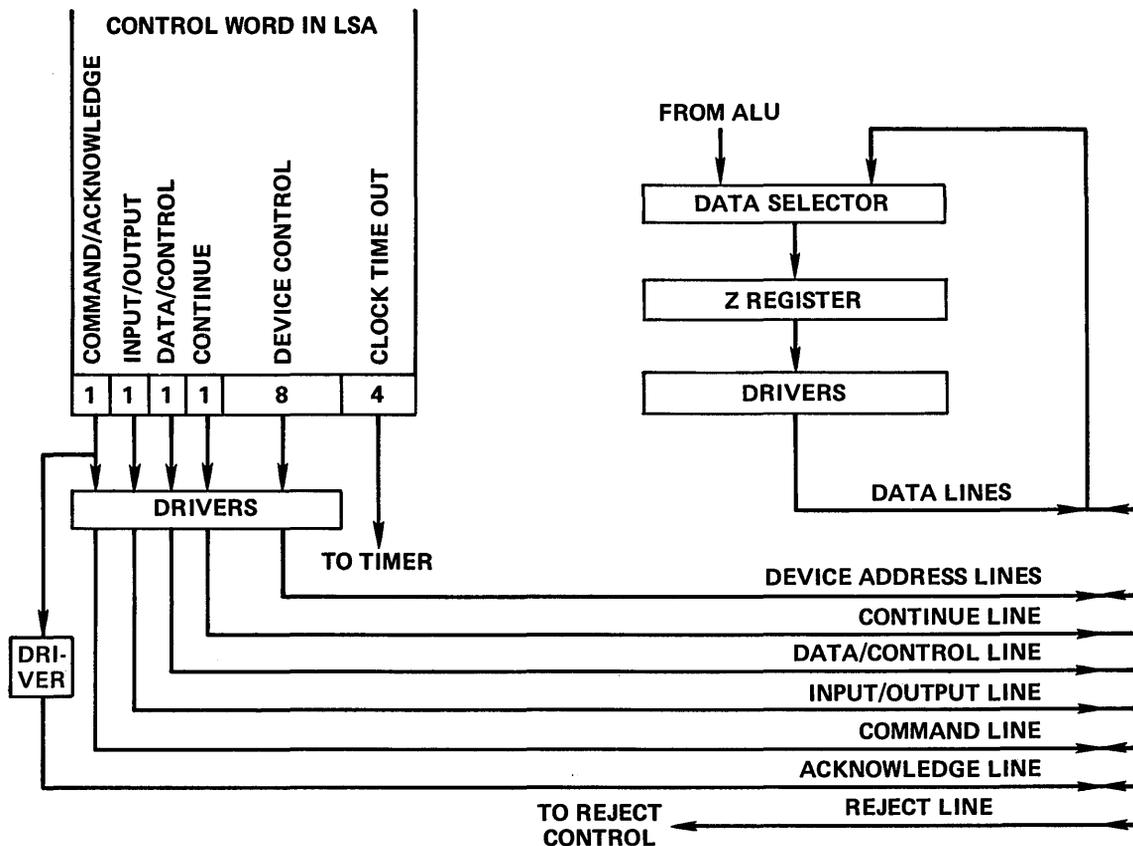


Fig. 4 - MCU-to-Z-bus interface

5.2.3 Unbuffered I/O Operation

I/O operations are initiated by an I/O instruction in an MCU. Devices cannot initiate I/O operations directly but do so by appealing to an MCU through the interrupt system. An I/O command is specified by a code point of 7 in the command word successor field. Prior to the I/O instruction, the I/O control word must be set up in LSA and, if the transfer is to be an output, data or command information must be set up in the Z register. Also, the programmer must specify in the LSA address field the location of the I/O control word in LSA, and the address of the Reject jump must be specified in the Literal field.

The MCU Command Register is loaded every cycle with a new instruction. If an I/O command is specified, the MCU checks whether it can get access to the Z bus. If the Z bus is available, the MCU initiates the I/O cycle. Absence of the Z-bus Available signal inhibits MCU operation until the bus becomes available at a later cycle.

An I/O command causes the control word drivers from LSA to drive the Z bus, and, if the command is an output, the data drivers from the Z register are also activated. Further MCU operation is suspended until the receipt of the Acknowledge Line signal which deactivates all Z-bus drivers. If the I/O command is an input, the information on the Z-bus data lines is set into the Z register. The MCU then proceeds with normal operation.

If the I/O operation is an Acknowledge, the Z-bus data lines are clocked into the Z register and the MCU continues with normal operation.

If the Reject Line is raised in response to an I/O Command, the Reject Line signal terminates the I/O cycle and sets the contents of the CS Literal field into the CSAR. This causes the MCU to resume normal processing at the I/O Reject jump address.

5.3 Buffered Channel Controller

The Buffered Channel Controller can take several forms. In the SPE system a Selector Channel Controller is specified as a generalized Direct Memory Access Controller.

The SCC interfaces with the Storage Control Unit (SCU), the Z bus, and up to eight Device Controllers (DC). Each DC may be attached to up to four SCC's.

The SCC interfaces with the Z bus to receive commands from and to transmit status information to an MCU. The SCC, in addition to the Z-bus lines, has an interrupt line to alert an MCU of a change in device status.

The SCC-DC interface is identical to the Z-bus interface except that the data width is 32 lines. Eight, 16, or 32 data-line DC's may be connected to the SCC. The particular width for an SCC-DC operation is determined by a 2-bit field (WDC) in the MCU I/O Command to the SCC.

The SCC contains a 32-bit assembly register at the SCC-DC interface and a 32-bit data-exchange register at the SCU-SCC interface. Two incrementing address registers and a word counter are contained in the SCC.

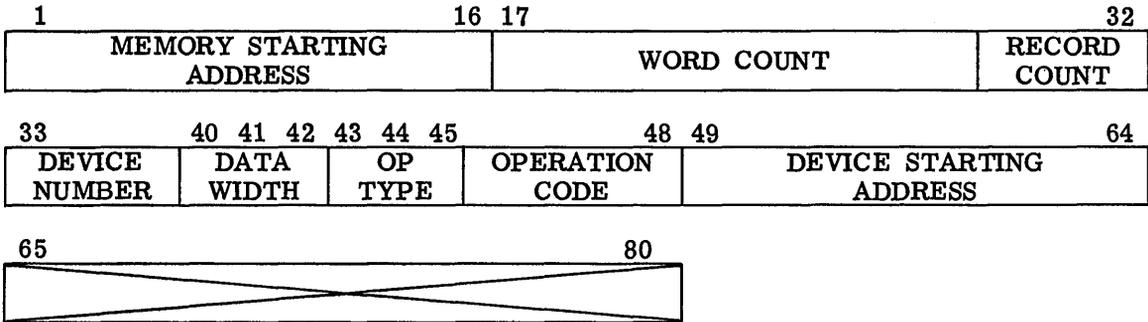
If an SCC is connected to the highest priority SCU position, a transfer rate to buffer memory of 192 Mbits/sec is possible. The maximum transfer rate is dependent on the round-trip line delay between the SCC and DC.

5.3.1 SCC Command and Format

The SCC Command Format includes the following fields:

- a. Buffer Memory Starting Address (15 bits)
- b. Word Count (12 bits)
- c. Record Count (4 bits)
- d. Device Number (8 bits)
- e. SCC-DC Data Width (2 bits)
- f. Operation Type (2 bits)
- g. Operation Code (4 bits)
- h. Device Starting Address.

SCC COMMAND FORMAT

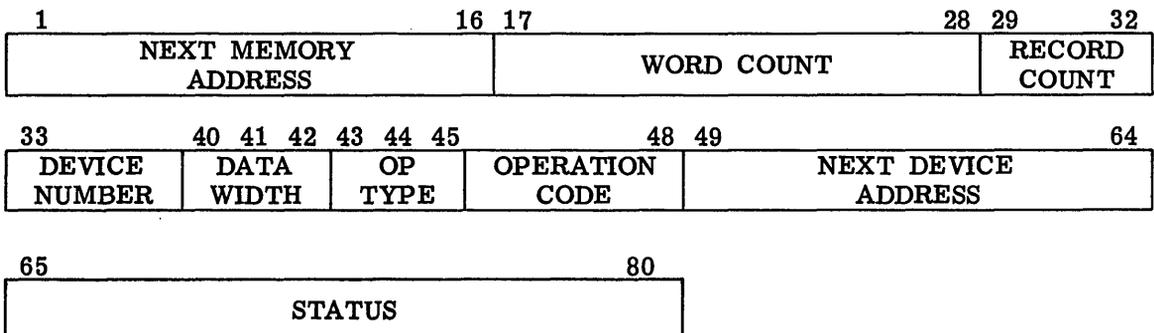


5.3.2 SCC Status Information and Format

The SCC status field includes the following:

- a. Next Memory Address (15 bits)
- b. Word Count (12 bits)
- c. Record Count (4 bits)
- d. Device Number (8 bits)
- e. SCC-DC Data Width (2 bits)
- f. Operation Type (2 bits)
- g. Operation Code (4 bits)
- h. Next Device Address (16 bits)

SCC STATUS FORMAT



6. SIGNAL PROCESSING ARITHMETIC UNIT (SPAU)

The SPAU operates under direction of the MCU. It is a special-purpose hardware device designed to provide very high-speed processing of FFT, recursive filter, and other signal processing algorithms. Its performance is indicated by a time of 300 nsec (two MCU cycles) to complete an SPAU-equivalent complex operation (four multiplications and six additions).

6.1 SPAU Organization

Two major sections provide the processing functions of the SPAU. These are the Arithmetic and Control Section (ACS) and the Address Generator and Control Section (AGCS). Both sections operate under microprogram control from read-only or read-mostly memories.

The ACS contains four high-speed multipliers (185 nsec) and six high-speed adders (25 nsec) which can operate in various parallel or serial configurations as governed by the microprogram control. Direct access to SPE buffer memories is provided via two buffered data channels allowing high data throughput in the SPAU.

The AGCS contains adders, counters, and other logic elements and provides the function of computing addresses needed by the ACS to access buffers and internal stores containing data used by the signal processing operations.

The SPAU is under development at the time of this report, and full detailed description can be found in the forthcoming SPAU Technical Reference Manual.

7. MCU MAINTENANCE PANELS

There are MCU and SCU maintenance panels (MP), (Figs. 5 and 6). The MP's contain control switches and indicators for operation and maintenance. The SCU maintenance panel interface shall be pluggable and identical to a DMA channel.

7.1 MCU Switches

The maintenance panel can be used to modify a microprogram sequence, stop the machine, and initialize the machine. The control switches to perform these operations are as follows:

1. **START:** Starts the microprogram sequence after a machine stop.
2. **RUN, SINGLE CYCLE, SINGLE PULSE:** A key switch with three positions. In the RUN position, the MCU runs at normal speed. In the SINGLE CYCLE position, the MCU advances a cycle at a time (P/1→P/2→P/3→P/4). In the SINGLE PULSE position, the MCU advances a pulse at a time whenever the SINGLE PULSE switch is depressed.
3. **EXECUTE, SEQUENCE:** A key switch with two positions. In the execute position, the command register outputs are enabled and displayed. In the sequence position, the CS output bus is displayed and the Command Register load is inhibited.
4. **DISABLE INTERRUPTS:** A two-position switch. In one position, the MCU operates normally. In the other position, all interrupts are disabled.

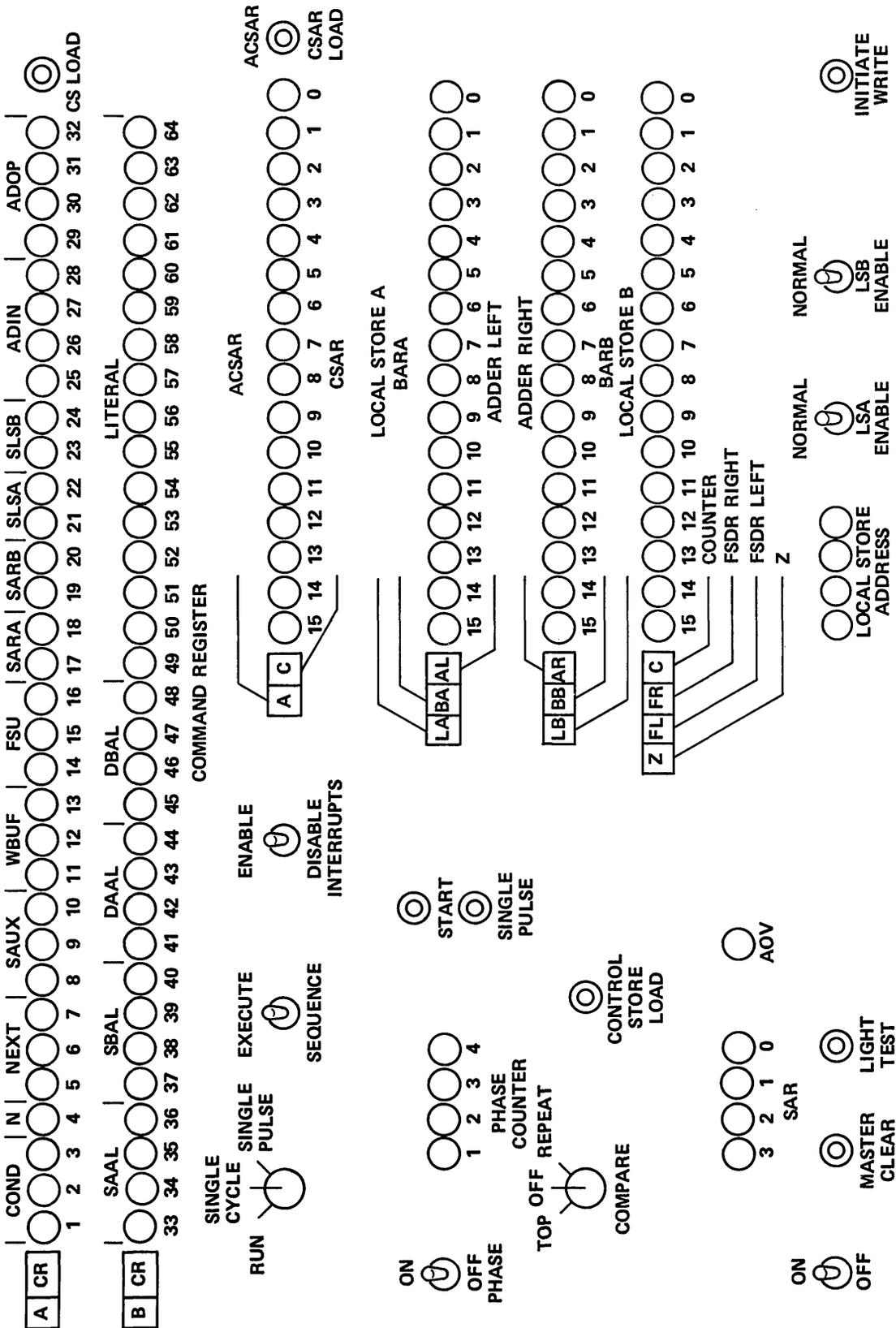


Fig. 5 - MCU maintenance panel

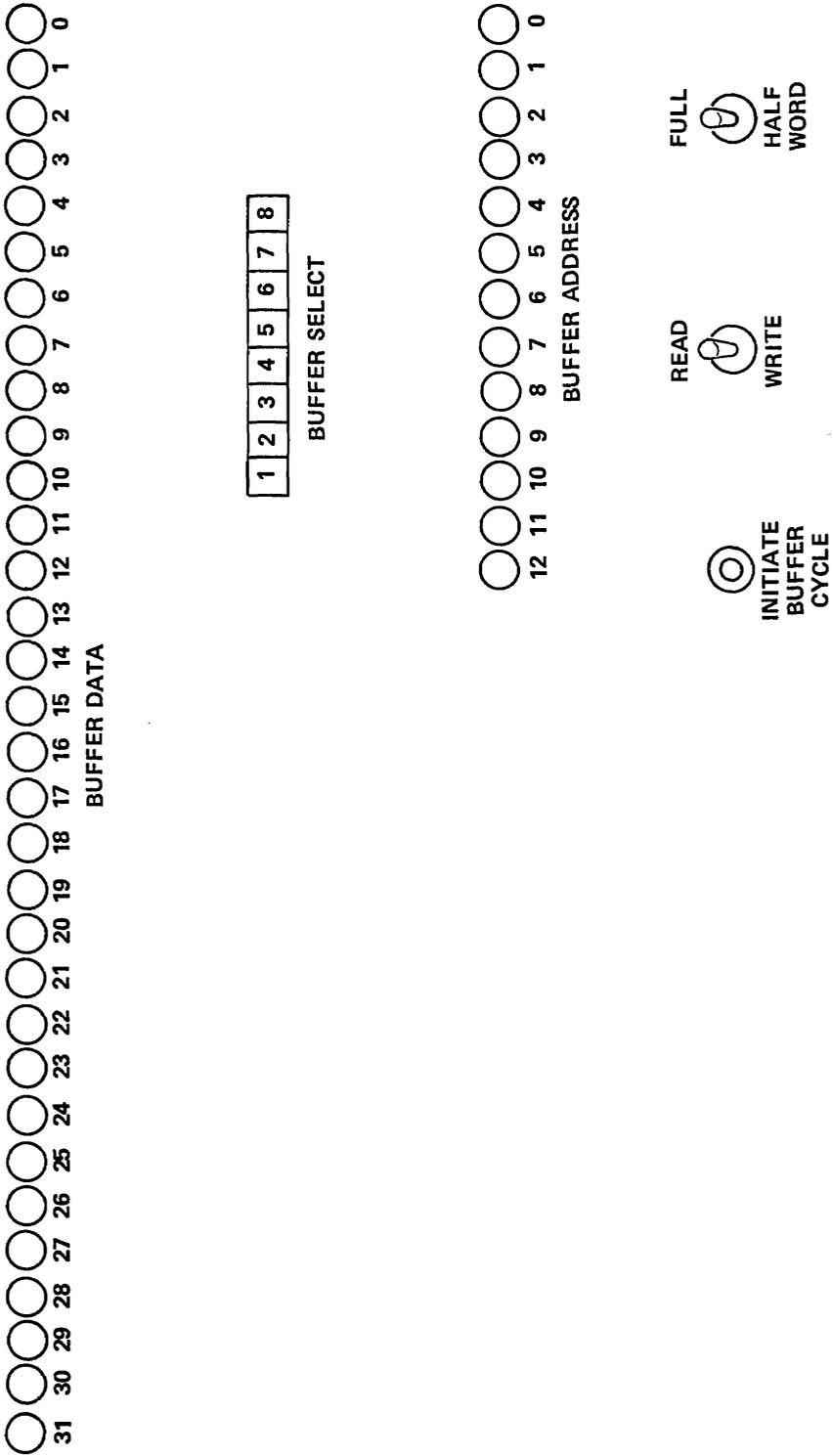


Fig. 6 - SCU maintenance panel

5. **PHASE ON, OFF:** A key switch with two positions. IN the ON position, the phase counter is permitted to advance; in the OFF position, the advance of the phase counter is inhibited. Pulses are taken from the single-pulse switch.

6. **SINGLE PULSE:** A pushbutton that produces a single pulse at each depression.

7. **COMPARE:**

a. CS STOP causes the machine to stop when the CSAR matches the value set in the 12 low-order data keys.

b. CS REPEAT causes the machine to repeat the address set in the address keys.

8. **CS LOAD:** Initiates the control store load sequence. This switch can only be activated in the STOP state.

9. Six sets of pushbuttons are used to select for display registers as described under Section 7.2. They are:

A. 1. A (BUS A)

2. CR (High-Order Command Register).

B. 1. B (BUS B)

2. CR (Low-Order Command Register).

C. 1. A (ACSAR)

2. C (CSAR)

D. 1. LA (LOCAL STORE A)

2. BA (BARA)

3. AL (ADDER LEFT SELECT)

E. 1. LB (LOCAL STORE B)

2. BB (BARB)

3. AR (ADDER RIGHT SELECT)

F. 1. Z

2. FL (FIELD SELECT DATA REGISTER - LEFT)

3. FR (FIELD SELECT DATA REGISTER - RIGHT)

4. C (COUNTER).

10. **ADDRESS KEYS:** 16 keys/indicators used to specify addresses entered into the MCU.

11. **DATA KEYS:** 64 keys/indicators used to specify data entered into the MCU.

12. **ACSAR, CSAR LOAD:** A pushbutton used to load the Alternate and Control Store Address Registers with the value in the address keys. The desired register is selected by depressing pushbuttons A and C. This switch can only be activated in the Stop state.

13. **CS LOAD KEY:** Causes the 64 data keys to be entered into the Control Store word specified by the CSAR. These switches can only be activated in the Stop state.

14. **LS ADDRESS KEYS:** 4 keys/indicators used to specify Local Store Addresses.

15. **LSA ENABLE:** A two-position switch used to select the LS ADDRESS KEYS as input to the LSA address register. This switch can only be activated in the Stop state.

16. **LSB ENABLE:** A two-position switch used to select the LS ADDRESS KEYS as input to the LSB address register. This switch can only be activated in the Stop state.

17. **INITIATE WRITE:** A pushbutton used to load the data specified in the low-order data keys into the Local Store word specified by the LS Address Keys. This switch is active only in the Stop state and if either LSA ENABLE or LSB ENABLE is active.

18. **MASTER CLEAR:** A pushbutton that when depressed clears the MCU to initial conditions.

19. **LAMP TEST:** A pushbutton switch which activates all display lights for test purposes.

20. **POWER ON, OFF:** Sequences power on or off.

7.2 SCU Switches

1. **BUFFER DATA KEYS:** 32 switches/indicators used to specify data to be entered into a buffer memory.

2. **BUFFER SELECT:** Eight pushbuttons used to select buffers. These buttons supply the high-order buffer address bits.

3. **BUFFER ADDRESS KEYS:** 13 switches/indicators used to specify the buffer memory low-order address bits.

4. **INITIATE BUFFER CYCLE:** A pushbutton which initiates buffer memory cycles.

5. **READ, WRITE KEY:** A two-position switch that determines whether retrieval or storage cycle is executed when the INITIATE BUFFER CYCLE pushbutton is depressed.

6. **FULL, HALF WORD KEY:** A two-position switch. In the FULL position 32 bits of information are transferred to or from Buffer Memory. In the HALF position, 16 bits of information are transferred to or from Buffer Memory. The high- or low-order 16 bits selected are determined by the state of the low-order address bit.

7.3 MCU Indicators

The MCU maintenance panel provides display capability in the MCU. Displayed are the following:

1. The 64-bit Command Register (CR). The high-order, 32 CR bits can be selected to display the Bus A outputs. The low-order, CR 32 bits can be selected to display the Bus B outputs.

2. The 12-bit Alternate Control Store Address Register (ACSAR) or the Control Store Address Register (CSAR) can be selected for display in a set of indicators.

3. Buffer Address Register A (BARA) or the left adder input can be selected for display in a set of indicators.

4. Buffer Address Register B (BARB) or the right adder input can be selected for display in a set of indicators.

5. Register Z, Field Select Data Register (FSDR) Left, FSDR Right, and Counter (CTR), all 16 bits, share the same indicators. The desired register contents are displayed by depressing the respective pushbutton.

6. The 4-bit Shift Amount Register (SAR) is displayed.

7. The 4-bit Phase Counter is displayed. The Phase Counter indicates which of the pulses P/1, P/2, P/3, or P/4 of the four-phase clock has occurred.

8. Adder Overflow (AOV) bit is displayed.

7.4 SCU Indicators

The SCU maintenance panel provides display capability. Displayed are the following:

1. 32 bits of Buffer Data are displayed.
2. 13 bits of Buffer Address are displayed.

8. MCU MICROINSTRUCTIONS

MCU microinstructions consist of 64-bit words read from the control store to the command register. The bits of each instruction word are numbered 1 to 64 from left to right. There is only one microinstruction format, shown below.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
COND			NOT		NEXT		SAUX			WBUF			FSU		
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
SARA		SARB		SLSA		SLSB		ADIN				ADOP			
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
SAAL				SBAL				DAAL				DBAL			
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
LIT															

8.1 Control Field Summary

The control fields of the microinstruction word are summarized in the list below. The 17 control fields provide the ability to specify several concurrent data and addressing operations in each MCU cycle.

<u>Field</u>	<u>CR Bits</u>	<u>Function</u>
COND	1-3	Condition test select
NOT	4	True or false of condition
NEXT	5-7	Next control store address selection
SAUX	8-10	Set auxiliary register
WBUF	11-13	Buffer write operation
FSU	14-16	Definition field for the Field Select Unit
SARA	17-18	Source for Buffer Address Register A (BARA)
SARB	19-20	Source for Buffer Address Register B (BARB)
SLSA	21-22	Source for Local Store A input
SLSB	23-24	Source for Local Store B input
ADIN	25-28	Adder input selection
ADOP	29-32	Adder/shifter operation selection
SAAL	33-36	Source Address Literal for LSA
SBAL	37-40	Source Address Literal for LSB
DAAL	41-44	Destination Address Literal for LSA
DBAL	45-48	Destination Address Literal for LSB
LIT	49-64	Literal (all-purpose data, address)

8.2 Microinstruction Operations

This section describes the operations of MCU microinstructions. The control fields allow a number of operations to be specified and to take place concurrently in a single microinstruction cycle. The MCU is designed so that, with few exceptions, any combination of control field functions may be specified without conflict. By being aware of the operation of microinstructions, the programmer can make optimum use of the parallelism in the MCU.

8.2.1 Instruction Addressing and Sequencing

The MCU control store is organized in 64-bit words. Each 64-bit word is an addressable microinstruction. The selection of the next microinstruction is made by the addressing mode specified in the NEXT field of the microinstruction and may be a sequential instruction obtained from the next sequential address.

Another sequencing method is a skip sequence where the next instruction is obtained from the second sequential address.

A third method is by a jump to the address specified in the Literal (LIT) field of the instruction word.

A fourth method is by a jump to the address specified by the contents of the Z register.

A fifth method is by a jump to the address specified by the contents of the ACSAR. This is the normal method of performing a return jump, since a call jump automatically saves the current address plus one into the ACSAR.

A final method of sequencing is the interrupt return jump, which is a jump to the address of the instruction about to be executed when the MCU was last interrupted.

Whatever address sequencing method is specified in the NEXT field, the performance of that sequence depends upon the presence of the condition specified in the COND and NOT fields of the microinstruction. If the specified condition is not present, the sequencing mode is always a sequential step.

8.2.2 ALU Operations

The ALU performs binary and unary operations on the L and R inputs for delivery to the Z register and various destinations. The programmer specifies three control functions in order to complete an ALU operation. These are ALU input selection, ALU function, and destination of result.

The MCU instruction cycle can be thought of as divided into two parts, the source phase and the destination phase. The source phase is defined as the early part of the instruction cycle, before the ALU operation, during which registers and local stores (which are data sources) are being read. The destination phase is defined as the later portion of the instruction cycle, after the ALU operation, during which destination registers and local stores are being written into. This concept will be referred to again in later sections of the microinstruction operation description.

8.2.2.1 ALU Input Selection

ALU input selection is specified by the ADIN field of the microinstruction. This field allows the programmer to select from among 16 combinations of L (left) and R (right) inputs to the ALU. The combinations provided allow for most frequently required operations on dual or single data inputs. Data transfer paths in the MCU provide for data movement among any registers or memory locations allowing complete flexibility in bringing data to the ALU for processing.

8.2.2.2 ALU Function

The ALU can perform arithmetic, logic, and shifting operations. The programmer has a choice of 16 operations in total, one of which he specifies in the ADOP field of the microinstruction.

All ALU binary operations except subtract are commutative operations, and in those cases, the order of selected inputs at L and R is not important.

With one exception, all unary operations are on the L input, and any register or local store is available to the L input by proper choice of the ADIN combination. It should be

noted that shifting is a unary operation on the L input. One unary operation on the R input is provided primarily for passing the contents of the LIT field of the microinstruction through the ALU; the LIT field is available only to the R input.

All ALU operations except NOP (no operation) cause the ALU result to be set into the Z register. An NOP specification leaves Z unchanged from the previous instruction.

8.2.2.3 ALU Result Destination

The ALU result is always set into the Z register. The programmer specifies which other registers or local stores will be destinations for the result. The user will note that there is no microinstruction field for explicitly specifying the destination of the ALU result. Rather, there are fields which allow him to specify the data source for each register and local store, and in this case, he would specify Z as the source. These fields are SAUX, WBUF, SARA, SARB, SLSA, and SLSB, and they provide for up to six ALU result destinations in a single instruction cycle not counting a Z jump address sequence specified by the NEXT field.

8.2.3 Local Store and Buffer Addressing

Each time local stores or buffer stores are specified as an input or output for a data transfer, the programmer must also specify address sources for these stores.

8.2.3.1 Local Store Addressing

Up to four local store addresses can be specified in a single microinstruction. Each local storage can be double addressed; that is, one location specified as a data source and a different location as a data destination in the same local store in the same instruction cycle. Fields SAAL, SBAL, DAAL, and DBAL are provided for this purpose. Each of these fields provides explicit addressing of 15 local store locations.

One address (zero) in each field, instead of being used for explicit addressing, specifies that that local store address is to be taken from the lower four bits of a Buffer Address Register (BAR). This provides a source of computable addresses for local stores, and, with the increment/decrement capability of the BAR's, facilitates use of the local stores as register stacks.

8.2.3.2 Buffer Store Addressing

Buffer stores can be addressed only by BARA and BARB. The user should note that the timing of data transfers to and from buffer store is such that the desired buffer addresses must be put into the BAR's at least one instruction before the instruction in which the buffer transfer is specified. This is so because BAR loading is a destination operation occurring in the latter part of the instruction cycle, whereas buffer operations require their addresses during the early part of the cycle. This does not prevent the programmer from specifying a buffer operation in one instruction while in the same instruction changing BAR contents for a buffer operation in a following instruction.

The BAR's provide 16 bits of address for specifying up to 64K 16-bit words of buffer store. There can be up to eight buffer modules of 8K words each, and the upper three bits of the BAR's specify the particular module being addressed. The two BAR's allow two buffer locations to be referenced in the same instruction as long as they are in different buffer modules. Violation of this rule will cause an MCU internal interrupt.

A buffer read or write operation always implies addressing from BARA and/or BARB. Microinstruction fields WBUF, SLSA, SLSB, and FSU allow the programmer to specify his choice of BARA or BARB as addresses along with buffer source and destination.

8.2.4 Buffer Operation

The MCU design provides for data transfers between buffer memories and other storage elements. These other storage elements are LSA and LSB (Local Stores A and B), FSDR (FSU Data Register), and Z. Buffer operations are not as flexible as other data transfers in the MCU in that a data transfer to or from buffer memory requires a full instruction cycle. As a result, a given buffer module can be read or written but not both in one instruction. This does not prevent a read from one buffer addressed by one BAR and a write into another buffer addressed by the other BAR in a single instruction. Similarly, two buffer reads or two buffer writes can be specified.

A buffer write operation, the buffer address source(s), and data source(s) can be specified explicitly by the WBUF microinstruction field. The source data involved in a buffer write operation will be the contents of the Z register or local store as left by the previous instruction.

A buffer read operation, the buffer address source(s), and data destination(s) are specified separately for the three possible destinations, LSA, LSB, and FSDR. The applicable fields are SLSA, SLSB, and FSU. The user will note that a buffer read into FSDR can be addressed only by BARA. A buffer read operation will load the destination register(s) in the latter part of the instruction cycle allowing the same register(s) to be used as data sources for other operations in the same instruction cycle.

8.2.5 Interrupt Operation

As mentioned in Section 2.4 of this manual, the programmer can specify no interrupt-associated operations except an interrupt-return jump in the NEXT field. However, the user must be aware that a recognized interrupt will unconditionally cause control store address sequencing to jump to one of locations 1 through 16 in the control store depending on the number of the interrupt line recognized. It is the responsibility of the programmer to provide instructions in these locations with address jumps to the routines in control store servicing the interrupts.

An interrupt will also leave in Z the control store address about to be executed when the interrupt was recognized. The user may or may not wish to make use of that data.

An interrupt-return jump can be specified in the microinstruction field NEXT. It is important to recognize that this operation causes Z, BARA, and BARB to be restored with their preinterrupt contents, and any attempt to specify data transfers into these registers in the same instruction will result in possible program errors.

8.2.6 Input/Output Operation

An I/O operation is specified by a special code (7) in the microinstruction field COND. If an I/O operation is specified, all other fields of the microinstruction have no effect on MCU operation except for SAAL and LIT. SAAL specifies the local store location containing the I/O control word and LIT contains the control store address to which control will jump if a reject signal is received from the external device addressed. See Section 5 of this manual for a description of I/O control word format.

It is the programmer's responsibility to set up the proper I/O control word in LSA and data in Z prior to issuing an I/O instruction. If the I/O instruction is an input command, the input data will be left in the Z register following the I/O instruction.

If the I/O instruction is an acknowledge, no data need be set up in Z prior to the instruction, but input data will be left in Z following the instruction.

8.3 Control Field Definitions

8.3.1 CØND Field

The CØND field is a 3-bit field defining an MCU hardware condition which controls the operation of the control store address successor specified in the NEXT field.

<u>Code Point</u>	<u>Symbol</u>	<u>Definition</u>
0	—	No condition
1	MST	Most significant bit of ALU result true
2	LST	Least significant bit of ALU result true
3	AØV	Adder overflow
4	CØV	Counter Register overflow*
5	ZERO	ALU result = 0
6	—	Spare
7	I/O	Indicates that the rest of the control word is to be treated by the MCU as an I/O instruction

8.3.2 NØT Field

The NØT field is a 1-bit field indicating whether the true or false value of the condition specified in the CØND field is to control the control store successor operation.

<u>Code Point</u>	<u>Symbol</u>	<u>Definition</u>
0	T	Specifies CØND
1	F	Specifies CØND

8.3.3 NEXT Field

The NEXT field is a 3-bit field specifying the next control store address to be formed if the condition specified by the CØND and NØT fields is satisfied.

*The counter register is incremented by one each time its condition is checked by the CØND field.

<u>Code Point</u>	<u>Symbol</u>	<u>Next CSAR</u>	<u>Next ACSAR</u>
1	SKIP	CSAR+2	NC
2	SAVE	CSAR+1	CSAR+1
3	CALL	CSLit	CSAR+1
4	JU MPL	CSLit	NC
5	JUMPA	ACSAR	NC
6	JUMPZ	Z	NC
7	INTRET	Interrupt Stack*	NC

If the condition specified is not satisfied, the default successor is a STEP.

8.3.4 SAUX Field

The SAUX field is a 3-bit field which specifies the 16-bit CS LIT field or Z register as source for a transfer into an auxiliary register: Counter (CNTR), Shift Amount Register (SAR), Alternate Control Store Address Register (ACSAR).

<u>Code Point</u>	<u>Transfer</u>	<u>Code Point</u>	<u>Transfer</u>
0	None	4	Z to SAR
1	LIT to ACSAR	5	LIT to CTR
2	Z to ACSAR	6	Z to CTR
3	LIT to SAR	7	Spare

8.3.5 WBUF Field

The WBUF (Write Buffer) field is a 3-bit field which designates the MCU source to be transferred to Buffer Storage Modules specified by BARA or BARB.

<u>Code Point</u>	<u>Transfer</u>
0	No Write
1	BUF(BARA)←LSA
2	BUF(BARB)←LSA
3	BUF(BARA)←LSB
4	BUF(BARB)←LSB
5	BUF(BARA)←Z
6	BUF(BARB)←Z
7	BUF(BARA)←LSA, BUF(BARB)←LSB

*See Section 2.4.

8.3.6 FSU Field

The FSU (Field Select Unit) field is a 3-bit control word field used as a data field selector. A "0" code point causes a transfer over Bus A of a Buffer word into the FSU Data Register (FSDR). Any other FSU code point will cause a masked select and right shift of the selected field in the FSDR to be made available, right justified, at the FSU output.

<u>Code Point</u>	<u>Definition</u>	<u>Code Point</u>	<u>Definition</u>
0	FSDR←BUF(BARA)	4	FSDR Field 4
1	FSDR Field 1	5	FSDR Field 5
2	FSDR Field 2	6	FSDR Field 6
3	FSDR Field 3	7	FSDR Field 7

8.3.7 SARA Field

The SARA (Source for buffer A Address Register A (BARA)) field is a 2-bit field specifying the source for BARA.

<u>Code Point</u>	<u>Source</u>
0	None
1	-1, decrement BARA by 1
2	+1, increment BARA by 1
3	Z

8.3.8 SARB Field

The SARB (Source for buffer B Address Register B (BARB)) field is a 2-bit field specifying the source for BARB.

<u>Code Point</u>	<u>Source</u>
0	None
1	-1, decrement BARB by 1
2	+1, increment BARB by 1
3	Z

8.3.9 SLSA Field

The SLSA (Source for Local Store A) field is a 2-bit field specifying the source for LSA input.

<u>Code Point</u>	<u>Source</u>
0	None
1	BUF(BARA)
2	BUF(BARB)
3	Z

8.3.10 SLSB Field

The SLSB (Source for Local Store B) field is a 2-bit field specifying the source for LSB input.

<u>Code Point</u>	<u>Source</u>
0	None
1	BUF(BARA)
2	BUF(BARB)
3	Z

8.3.11 ADIN Field

The ADIN field is a 4-bit field which specifies the Left (L) and Right (R) inputs to the ALU.

<u>Code Point</u>	<u>L Input</u>	<u>R Input</u>
0	LSA	LIT
1	LSA	FSU
2	LSA	BARB
3	LSA	LSB
4	LSB	LIT
5	LSB	FSU
6	LSB	BARB
7	LSB	BARA
8	BARA	LIT
9	BARA	LSA
10	BARB	LIT
11	BARB	BARA
12	FSU	LIT
13	CNTR	LIT
14	ACSAR	LIT
15	SAR	LIT

8.3.12 ADØP Field

ADØP is a 4-bit field which specifies the operation that the ALU is to perform on the L and R inputs. The ALU result goes into the Z register.

<u>Code Point</u>	<u>Symbol</u>	<u>Definition</u>
0	NOP	No operation (Z remains unchanged)
1	L+R→Z	L plus R
2	L-R→Z	L minus R
3	L+1→Z	L plus 1
4	L-1→Z	L minus 1
5	L→Z	L
6	\bar{L} →Z	L complement
7	R→Z	R
8	0→Z	Zero
9	L OR R→Z	Bit logical OR of L and R
10	L AND R→Z	Bit logical AND of L and R
11	L XOR R→Z	Bit logical EXCLUSIVE-OR of L and R
12	L EQU R→Z	Bit logical EQUIVALENCE of L and R
13	LEFT L→Z	Left shift L, zero fill (shift controlled by SAR)
14	RIGHT L→Z	Right shift L, zero fill
15	CIRC L→Z	Left circular shift L

8.3.13 SAAL Field

The SAAL field is a 4-bit field specifying the address of the register in Local Store A which is to be a data source.

<u>Code Point</u>	<u>Definition</u>
0	LSA address = contents of BARA
1	LSA address = 1
2	LSA address = 2
.	.
.	.
.	.
15	LSA address = 15

8.3.14 SBAL Field

The SBAL field is a 4-bit field specifying the address of the register in Local Store B which is to be a data source.

<u>Code Point</u>	<u>Definition</u>
0	LSB address = contents of BARB
1	LSB address = 1
2	LSB address = 2
.	.
.	.
.	.
15	LSB address = 15

8.3.15 DAAL Field

The DAAL field is a 4-bit field specifying the address of the register in Local Store A which is to be a data destination.

<u>Code Point</u>	<u>Definition</u>
0	LSA address = contents of BARA
1	LSA address = 1
2	LSA address = 2
.	.
.	.
.	.
15	LSA address = 15

8.3.16 DBAL Field

The DBAL field is a 4-bit field which specifies the address of the register in Local Store B which is to be a data destination.

<u>Code Point</u>	<u>Definition</u>
0	LSA address = contents of BARB
1	LSA address = 1
2	LSA address = 2
.	.
.	.
.	.
15	LSA address = 15

8.3.17 LIT Field

The LIT field is a 16-bit field serving as a literal source of data and/or addresses to various devices in the MCU.

9. ACKNOWLEDGMENT

Work leading to the SPE System Reference Manual was performed by a design team which included the authors, John D. Roberts, Jr., Bruce Wald, and Y. S. Wu, all of the Information Systems Group.

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)		2a. REPORT SECURITY CLASSIFICATION	
Naval Research Laboratory Washington, D.C. 20390		Unclassified	
3. REPORT TITLE		2b. GROUP	
SIGNAL PROCESSING ELEMENT USERS' REFERENCE MANUAL			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates) An interim report on a continuing project.			
5. AUTHOR(S) (First name, middle initial, last name) William R. Smith and John P. Ihnat			
6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS	
September 5, 1972	37		
8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)		
NRL Problems B02-06 and B02-10	NRL Report 7488		
b. PROJECT NO.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)		
NAVAIR WF 15-241-601 and			
c.			
NAVELEX XF 21-241-015-K152			
d.			
10. DISTRIBUTION STATEMENT			
Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY	
		Department of the Navy (Naval Air Systems Command) Washington, D.C. 20360	
13. ABSTRACT			
<p>The NRL Signal Processing Element (SPE) is a high-performance signal processing facility for radar, sonar, and communication systems. It is intended to be compatible with the Navy All Applications Digital Computer (AADC). The SPE consists of four major subsystems: a Microprogrammed Control Unit (MCU), a Buffer Store and Storage Control Unit (SCU), a Signal Processing Arithmetic Unit (SPAU), and Input Output (I/O) units.</p> <p>The MCU serves as system supervisor and data organizer for the SPAU and other I/O devices. The MCU includes a 64-bit Control Store, two local stores, an arithmetic element, two busses to buffer memory, an unbuffered byte channel, and a priority interrupt system. Its cycle time is 150 nsec.</p> <p>The buffer store and SCU consists of eight fixed-priority Direct Memory Access (DMA), or Buffered Channels, communicating on a 150-nsec cycle basis with up to eight 32-bit by 4K memories.</p> <p>The SPAU is a highly parallel, high-speed arithmetic unit capable of performing complex signal processing operations such as FFT and recursive filtering. It is microprogrammed for flexibility in adapting signal processing algorithms. Four multiplies and six additions can be performed in 300 nsec.</p> <p>SPE I/O and internal communications are provided by DMA buffer data channels, an unbuffered byte channel, and by a priority interrupt system. The unbuffered byte channel called the Z bus communicates both data and control information to all I/O devices. The unbuffered channel burst data rate is 2 million 16-bit words per second.</p>			

14.

KEY WORDS

Signal Processing Element
 Microprogrammed Control Unit (MCU)
 MCU
 Microprogramming
 Signal Processing
 All Applications Digital Computer (AADC)

LINK A

LINK B

LINK C

ROLE

WT

ROLE

WT

ROLE

WT