PERKIN ELMER

MODEL 3250 PROCESSOR MICROPROGRAMMING

Reference Manual

50-004 R00

The Perkin-Elmer Corporation, Computer Systems Division 2 Crescent Place, Oceanport, New Jersey 07757

Printed in the United States of America

TABLE OF CONTENTS

CHAPTERS (Continued)

CHAPTERS (Continued)

CHAPTERS (Continued)

FIGURES

TABLES

PREFACE

This manual describes the microprogram for the Perkin-Elmer Model 3250 processor. It provides a block diagram analysis of the processor, data and instruction formats, information on source and destination registers, the microinstruction repertoire, information on the input/output (I/O) system and the interrupt system, and microprogramming restrictions.

This manual is intended to be used in conjunction with the following manuals:

MANUAL	PUBLICATION NUMBER
Model 3250 Processor Maintenance Manual	47-029
Model 3250 Processor User's Manual	50-001
32-Bit Systems User Documentation Summary	50-003

For further information on the contents of all Perkin-Elmer 32-bit manuals, see the 32-Bit Systems User Documentation Summary.

CHAPTER 1
MICROPROGRAM DESCRIPTION

1.1 INTRODUCTION

Microprogramming is a means for implementing the control logic of a digital computer and has been effectively used to maintain upward program compatibility in a family of processors whose internal hardware varies from one member to the next.

The processor is designed to execute microinstructions stored in a control store Read-Only-Memory (ROM). Each microinstruction causes one or more hardware functions to be performed, such as transferring the contents of one register to another, arithmetic or logical operations between registers, controlling input/output operations, or initiating main memory accesses.

A series of microinstructions is called a microprogram. The complete microprogram, defined as an emulator, causes the microprocessor to react to a user program in main memory and to external events. A similar processor reaction is described in the Model 3250 Processor User's Manual. Every user level instruction, interrupt handling feature, and system CRT function is simulated by some portion of the microprogram.

1.2 BLOCK DIAGRAM ANALYSIS

Refer to the block diagram in Figure 1-1.

1.2.1 System Organization

The processor is organized between three 32-bit buses. The A and B buses are used to present the first and second operand data, respectively, to the Arithmetic Logic Unit (ALU). ALU output to the appropriate destination is then transferred by the S bus. The source and destination of data on the A, B, and S buses, as well as the function performed by the ALU, is controlled by microinstructions contained in the control store memory.

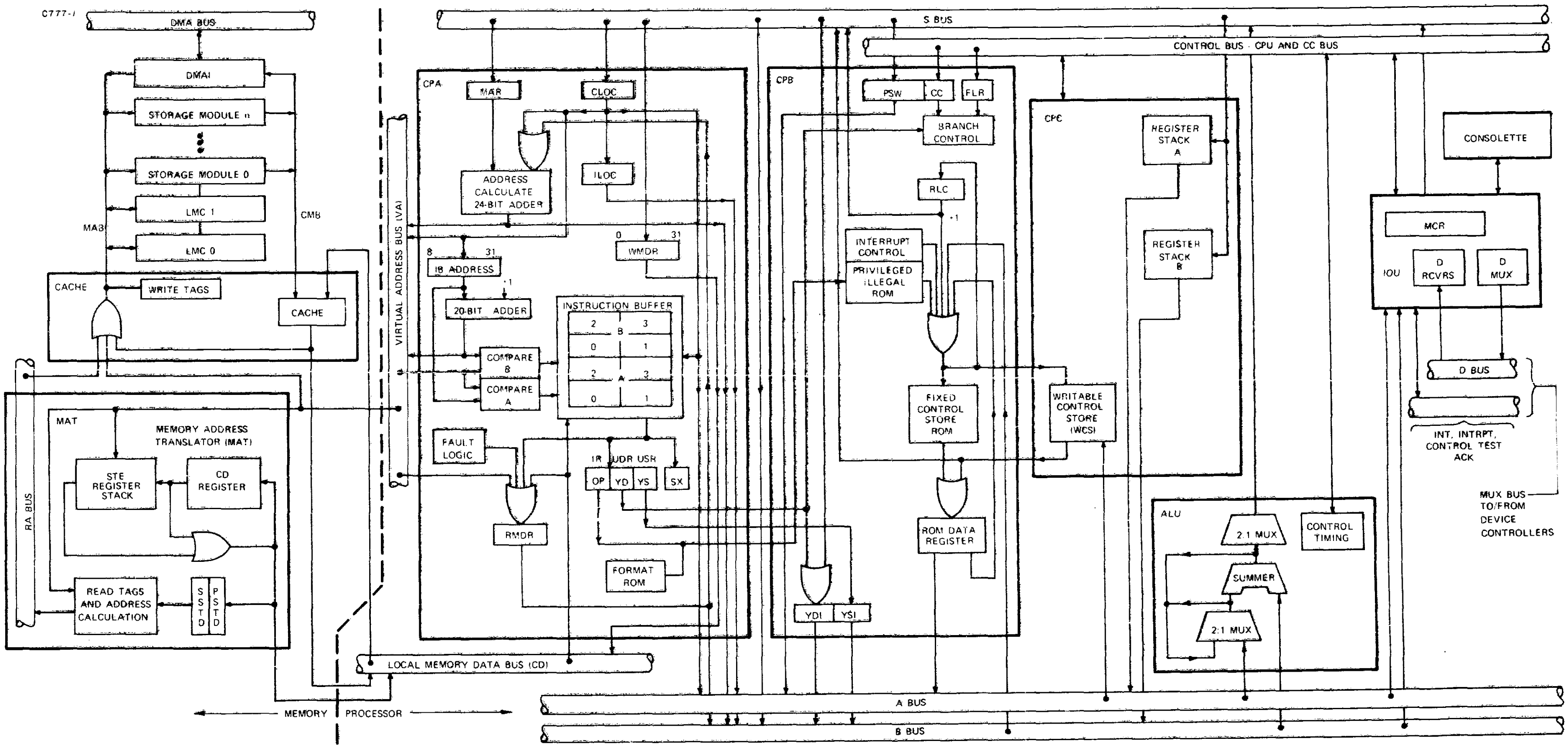


Figure 1-1 Model 3250 Processor Hardware Block Diagram

1.2.2 Control Store Memory

The control store memory is a high speed, solid-state, nondestructive readout memory organized into 16 pages of 256 words each. Each word is 32 bits long and represents one microinstruction. The first eight pages (2,048 words) in the control store memory contain the entire microprogram. Additional pages of writable control store memory can be added to the basic processor, allowing the user to supplement the standard instruction repertoire with special algorithms or application oriented functions without requiring hardware involvement.

Each microinstruction read from the control store memory is placed in the 32-bit ROM Instruction Register (RIR). Most microinstructions are executed in one machine cycle of 260 nanoseconds. At the conclusion of each microinstruction, the next one to be performed is read out and placed in the RIR. Microinstruction word bit definitions are explained later.

Locations in the control store memory are addressed by the 12-bit output from the ROM Address Gate (RAG). Inputs to the RAG may be: the ROM Location Counter (RLC) which selects the next microinstruction to be performed; certain bits of the ROM Instruction Register (RIR) for branches and transfers; the B bus for data addressing and branches; the user level operation code for entering an emulation routine; or the interrupt control logic for entering interrupt service routines.

Microinstructions are normally executed from sequential control store memory locations. After a microinstruction is read into the RIR, the RLC is loaded with the address of the next sequential instruction. When it is necessary to jump to a different program sequence, the first microinstruction in that sequence is addressed through the RAG from ROM Instruction Register (RIR) bits or B bus bits. The new address is also loaded into the RLC so that sequential instructions can again be executed.

During user instruction decoding, the user's operation code times two is presented to the RAG to address the first instruction of the sequence emulating that user's instruction. The new address is also loaded into the RLC.

In response to an interrupt, the interrupt hardware presents an address to the RAG. If the address is that of a branch and link type instruction, the hardware has time to save the current RLC value plus one in the designated link register before the RLC is updated from the RAG. In this way, the microcode could return to the interrupted sequence after servicing the interrupt, if desired.

The execute instructions are the only instructions in which the RLC is not updated. After executing the selected out-of-line instruction, the next microinstruction in the in-line microcode sequence is performed.

1.2.3 Flag Register (FLR)

The Flag Register (FLR) is a 4-bit register containing the following flags: Carry (C), Overflow (V), Greater than Zero (G), and Less than Zero (L). These flags are modified from the condition code bus at the conclusion of arithmetic, logical, and I/O operations reflecting operation results.

1.2.4 Program Status Word (PSW)

The Program Status Word (PSW) is a 32-bit register used to indicate the system status relative to the user program being emulated. Bits 0:27 of the PSW define enabled interrupts and the operational status or mode of the user-level processor. Some of the PSW bits have hardware significance, while others are of significance only to the emulator. Bits 28:31 of the PSW make up the condition code (CC) field, which reflects the results of the last executed user-level instruction. The condition code may be updated from the condition code bus, or when the PSW is the specified destination register. Bits 0:9, 12, and 15 of the PSW are not implemented.

The Location Counter (LOC) is a 32-bit extension to the PSW, holding the address in main memory of the next user instruction to be performed. During an instruction memory read, LOC is used to address main memory. For all other main memory accesses, the 32-bit Memory Address Register (MAR) is used. Only the 24 least significant bits (bits 8:31) of LOC and MAR are implemented. At the machine level, LOC consists of registers CLOC, the current location counter, and ILOC, the instruction location counter. CLOC is copied to ILOC when instruction read is started; CLOC then increments by two for every instruction halfword read.

1.2.5 Main Memory

Main memory consists of a number of 256 kb Metal Oxide Semiconductor (MOS) memory modules, providing storage for user instructions and data. Data read from or written into memory is buffered in the 32-bit Memory Data Registers (MDRs). There are separate MDRs for reading from and writing to main memory.

The microprogram initiates a main memory cycle by issuing a memory read or memory write command. After issuing a memory command, the microprogram is free to do other instructions. The memory cycle is accomplished asynchronous of other processor activity. However, if the microprogram attempts to use the contents of RMDR after a memory read, or attempts to load MAR, the processor stops until the desired function can be performed. This also occurs if the microprogram attempts to issue another memory command before the current memory cycle is complete.

After an instruction read has been issued and the read-out becomes available, bits 0:7 are placed in the OP register; bits 8:11 are placed in the YDI register; and bits 12:15 are placed in the YSI register. These three registers comprise the user's instruction register (UIR).

The OP register, containing the user's operation code, is used to address the privileged/illegal ROM and the control store memory itself. The user's operation code times two is the control store memory address of the first microinstruction of the appropriate emulation sequence. The privileged/illegal ROM is a separate ROM containing 256 4-bit words. This ROM is interrogated before entering the microsequence that emulates a user-level instruction. If the op-code is illegal, or is that of a privileged instruction and PSW bit 23 is set, or if the op-code is that of a floating-point instruction and PSW bit 13 is set, the illegal instruction interrupt is generated.

1.2.6 General Registers

The eight sets of user-level general registers each contain 16 32-bit registers. The register sets (stacks) are duplicated for the A bus and B bus. (See Figure 1-1.) Only one set of general registers is active at a time, depending upon PSW bits 25, 26, and 27. (See Table 1-1.)

The microprogram usually accesses the user's general registers without caring which of the 16 registers in the active set is used. However, when the microprogram accesses a general register for emulating a user instruction, it must be the general register specified in that user instruction. After the instruction read, the register addresses specified by the user are in the YDI and YSI registers; therefore, the microprogram can access the appropriate general register by specifying the YDI or YSI register. The hardware then selects the general register whose number is in the YDI or YSI register. The user's general registers are also directly addressable by the microprogram when it is necessary to access specific registers.

TABLE 1-1 REGISTER SET SELECTION

PSW BITS			ACTIVE REGISTER SET
25	26	27	
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	F

1.2.7 Scratchpad Registers

A single set of 16 32-bit registers is available to the microprogram as module 7. These registers may be directly addressed, or are specified by the contents of the YDI or YSI register. The ALU responds with module 7 activity.

1.2.8 Microregisters

The eight 32-bit microregisters (MRO:7) are available to the microprogram as general purpose registers.

1.2.9 Arithmetic Logic Unit (ALU)

The 32-bit A bus provides the first operand for arithmetic and logical operations. The 32-bit B bus provides the second operand. The A and B buses are input to the Arithmetic Logic Unit (ALU), which performs addition, subtraction, multiplication, division, shifting, and Boolean connect functions. The output of the ALU is the 32-bit S bus.

1.2.10 Input/Output

Input/Output (I/O) operations are accomplished by gating data from the A bus and/or B bus onto the 16-bit I/O bus, and by gating data from the I/O bus onto the S bus.

The I/O bus consists of 33 lines: 16 bidirectional data lines, 10 control lines, 6 test lines, and an initialize line. See the chapter on the I/O system.

1.2.11 Interrupt Control

The interrupt control logic provides rapid response to internal and external events. Nine priority interrupt lines are available, each with a unique trap location in the control store memory. Recognition of an interrupt causes the microinstruction at the trap location to be performed. Certain interrupts can be disabled or enabled by PSW bits. Interrupts can also be disabled or enabled as a group by the microprogram. (See Table 1-2.)

TABLE 1-2 INTERRUPT TRAPS

INTERRUPT	TRAP ADRS (HEX)	MASK	GROUP ENABLE
Floating Point Fault Data Fetch Fault(MAT, ECC, or Alignment) (MAIO Abort)	207 207	PSW13,19(FPP) PSW21 (MAT) PSW18 (ECC)	NO
Primary Power Fail	206	NONE	YES
Machine Malfunction	205	PSW18	YES
Console Attention	204	NONE	YES
External Interrupt Level 0	203		
External Interrupt Level 1	202	See	
External Interrupt Level 2	201	Table	YES
External Interrupt Level 3	200	3	
Illegal Instruction	208	NONE	N/A

PSW bits 17 and 20 define the external interrupt enable status of the processor as shown below:

PSW BITS		
17	20	
0	0	All levels disabled
0	1	Higher levels enabled
1	0	All levels enabled
1	1	Current and higher levels enabled

where the current level is a function of the currently active register set. (See Table 1-3.)

TABLE 1-3 EXTERNAL INTERRUPT ENABLE

PSW BITS					EXTERNAL INTERRUPT ENABLED			
17	20	25	26	27	LEVEL 0	LEVEL 1	LEVEL 2	LEVEL 3
0	0	X	X	X	NO	NO	NO	NO
0	1	0	0	0	NO	NO	NO	NO
0	1	0	0	1	YES	NO	NO	NO
0	1	0	1	0	YES	YES	NO	NO
0	1	0	1	1	YES	YES	YES	NO
0	1	1	0	0	YES	YES	YES	YES
0	1	1	0	1	YES	YES	YES	YES
0	1	1	1	0	YES	YES	YES	YES
0	1	1	1	1	YES	YES	YES	YES
1	0	X	X	X	YES	YES	YES	YES
1	1	0	0	0	YES	NO	NO	NO
1	1	0	0	1	YES	YES	NO	NO
1	1	0	1	0	YES	YES	YES	NO
1	1	0	1	1	YES	YES	YES	YES
1	1	1	0	0	YES	YES	YES	YES
1	1	1	0	1	YES	YES	YES	YES
1	1	1	1	0	YES	YES	YES	YES
1	1	1	1	1	YES	YES	YES	YES

1.2.12 Machine Control Register (MCR)

The 12-bit Machine Control Register (MCR) can be interrogated or cleared by the microprogram. A definition of the MCR bits is detailed in the section on control functions. (See Section 4.8.1.)

CHAPTER 2
DATA AND INSTRUCTION FORMATS

2.1 DATA FORMATS

All internal data paths except those to the Input/Output (I/O) control are 32 bits wide. The basic machine operand is, consequently, a 32-bit fullword. Positive fixed-point data is expressed in true binary form with a sign bit of zero. Negative fixed-point data is expressed in two's complement notation with a sign bit of one. Floating-point data is expressed as a signed magnitude fraction with a signed exponent. The quantity expressed is the product of the fraction and 16 raised to the power of the exponent. Each floating-point number requires a 32-bit fullword; 8 bits are used for the fraction sign and exponent, and 24 bits are used for the fraction.

Binary information is represented in hexadecimal notation (base 16) for simplicity.

2.2 INSTRUCTION FORMATS

The microinstructions for the processor can be one of six formats designated Address Link, Register Link, Register-to-Register Transfer, Register-to-Register Control, Register-to-Register Immediate, and Register Write. These instruction formats are shown in Figure 2-1.

The basic instruction format provides the microprocessor with a three-address capability, but various options of the repertoire can modify the range from two to four.

Bits 0, 1, and 2 of the microinstruction select the processor module that performs the specified function. The address link and register link microinstructions are the only ones that select module 0, the control module. The other microinstruction formats can be directed to any other module. The processor's Microcode Assembler recognizes symbolic operation codes directed to modules 0 (the control module), 1 (the ALU module), 2 (the I/O module), 6 (floating-point processor module), and 7 (scratchpad registers).

The meaning of each microinstruction word field is summarized in Table 2-1 and the following paragraphs.

ADDRESS LINK

0 2	3	4	5	6 10	11 13	14 25	26	27 31
0 0 0	1	X	T	S	F	ADDRESS	E	MC

REGISTER LINK

0 2	3	4	5	6 10	11 13	14 19	20 24	25	26	27 31
0 0 0	0	X	T	~~NULL~~ S	F		B		E	MC

REGISTER-TO-REGISTER TRANSFER

0 2	3	4	5	6 10	11 15	16 19	20 24	25	26 31
Module	0	0	I	S	A	F	B	C	PAGE ADDRESS

REGISTER-TO-REGISTER CONTROL

0 2	3	4	5	6 10	11 15	16 19	20 24	25	26	27 31
Module	0	1	I	S	A	F	B	K	E	MC

REGISTER-TO-REGISTER IMMEDIATE

0 2	3	4	5	6 10	11 15	16 19	20 31
Module	1	0	I	S	A	F	DATA

REGISTER WRITE

0 2	3	4	5	6 10	11 15	16 19	20 24	25	26	27 31
0 0 1	1	1	I	NULL	A	F	B	K	E	MC

Figure 2-1 Instruction Formats

TABLE 2-1 INSTRUCTION WORD FIELDS

FIELD	MEANING
A	Selects register to be used as first operand
B	Selects register to be used as second operand
C	If set, transfer is conditional.
E	Enable setting of condition code
F	Specifies function of addressed module
I	B bus data addresses actual data in control store.
K	F field extension
MC	Memory control field
S	Selects register to receive the result
T	If set, item F must be true for transfer.
X	Execute

The F field in all formats specifies the function that the selected module is to perform. The X bit in the address link and register link formats distinguishes Execute and Link instructions from Branch and Link instructions. The T bit specifies whether the true or false state of the condition F is to be tested.

The S field selects the S bus register to be loaded. The A field selects the first operand (A bus) register. The B field selects the second operand (B bus) register.

Setting the I bit causes the operand developed on the B bus to be taken as a control store memory address. The fullword contents of the addressed location replace the original B bus data. This function adds 160 nanoseconds to the execution time of the instruction.

Setting the C bit on Register-to-Register Transfer instructions causes the transfer to occur only if no predefined signal is returned from the addressed module. For the ALU module and the scratchpad module, the signal is Carry, meaning that no transfer occurs if a carry is generated. For the I/O module, the signal is Halfword (no transfer occurs if the addressed device is a halfword device). For all other modules, the signal is undefined.

The K bit is used as an extension of the F field, allowing more than 16 functions to be performed by the addressed module.

The E bit allows the Condition Code (CC) field of PSW to be updated with data on the CC bus from the addressed module. Once an instruction with the E bit set has been performed, the condition code remains connected to the CC bus until an instruction having an E bit field with the E bit reset is fetched.

The MC field controls main memory accesses, and MAR and LOC activities. MC can also enable the privileged/illegal ROM and the instruction decoding hardware. In this case, unless a branch is taken or an interrupt occurs, a user instruction emulation sequence is entered.

The most significant bit of the 12-bit immediate field on Register-to-Register Immediate instructions is propagated through the most significant 20 bits on the B bus. For example, the immediate operand '400' produces the value '0000 0400' on the B bus. The immediate operand '800' produces the value 'FFFFF800' on the B bus.

The 6-bit address field on Register-to-Register Transfer instructions can specify any address within the local 64-word page. For example, an instruction at address '131' can transfer to any other instruction from address '100' to '13F'. The incremented RLC always determines the lower and upper limits of the transfer destination. Thus, an instruction at address '13F' can transfer to any instruction from address '140' to '14F', but cannot transfer to an instruction at address '13E'.

2.2.1 Address Link

When executing the Address Link instruction, the incremented contents of RLC are placed in the selected S bus register. If the condition specified by F and T is met, the next microinstruction executed is the one at the location specified by the 12-bit address field. If the condition is not met, the next microinstruction in sequence is executed. In addition, if the condition is met, any memory control or decode options specified are suppressed.

2.2.2 Register Link

The Register Link instructions are identical to the Address Link instructions, except that the address to transfer to is taken from the register specified by B.

2.2.3 Register-to-Register Transfer

These instructions perform function F using a first operand specified by the contents of the register specified by A, and an effective second operand specified by B. The result replaces the register specified by S. If the C bit is reset or if a special signal (MODSIG) is not returned from the addressed module, the next microinstruction executed is from the current page of the control store memory address specified by the PAGE ADDRESS field. If the C bit is set and the special signal is returned from the addressed module, the next microinstruction in sequence is executed. The PAGE ADDRESS field can specify only the least significant 6 bits of a control store memory address. The remaining address bits are taken from the high order 6 bits of RLC. This means that a transfer can occur only to a location within the 64-word page defined by RLC bits 4:9. An exception is when the microinstruction is at the end of page boundary (e.g., address '23F'). In this instance, the transfer occurs to the specified address on the next sequential page (e.g., one of the addresses '240' through '27F').

The effective second operand, B_E, is the contents of the register specified by B if I=0:

$$B_E = (B)$$

or the contents of the fullword control store memory location whose address is in the register specified by B if I=1:

$$B_E = [(B)]$$

2.2.4 Register-to-Register Control

These instructions perform function F using a first operand specified by the contents of the register specified by A, and an effective second operand specified by B. The result replaces the contents of the register specified by S.

The effective second operand, B_E, is the contents of the register specified by B if I=0:

$$B_E = (B)$$

or the contents of the fullword control store memory location whose address is in the register specified by B if I=1:

$$B_E = [(B)]$$

At the conclusion of the instruction, or as soon as logically practical, any specified memory control options are performed.

2.2.5 Register to Register Immediate

The function specified by F is performed using the contents of the register specified by A as the first operand and an effective second operand specified by the data field. The result replaces the contents of the register specified by S.

The effective second operand, B_E, is the 12-bit value of the data field with the most significant 20 bits equal to bit 20 if I=0:

B_E =DATA

or the contents of the fullword control store memory location whose address is specified by DATA if I=1:

B_E =[DATA]

2.2.6 Register Write

The Register Write instruction stores the contents of the register specified by A into the Writable Control Store (WCS) location whose address is in the register specified by B. After the write, any specified memory control functions are performed.

If the processor is not equipped with WCS, only the specified options are performed.

2.3 MAIN MEMORY CONTROL

The processor's main memory is the source of user instructions and data. Control over the main memory is provided in the MC field of the Address Link, Register Link, Register-to-Register Control and Register Write microinstructions.

Table 2-2 and the following paragraphs describe the MC field options.

TABLE 2-2 MC FIELD

BITS					MNEMONIC	MEANING
27	28	29	30	31		
0	0	0	0	0	-	No action
0	0	0	0	1	DR2IB	Data Read, 2 bytes, from IB
0	0	0	1	0	IR	Instruction Read
0	0	0	1	1	DR4IB	Data Read, 4 bytes, from IB
0	0	1	0	0	RAS	Read and Set
0	0	1	0	1	RFAULT	Reset Fault, Reset RX format flip-flops
0	0	1	1	0	PR2	Privileged Read, 2 bytes
0	0	1	1	1	DR2	Data Read, 2 bytes
0	1	0	0	0	-	No action
0	1	0	0	1	-	No action
0	1	0	1	0	I1DR1	Increment MAR by 1, Data Read, 1 byte
0	1	0	1	1	DR1	Data Read, 1 byte
0	1	1	0	0	PR4	Privileged Read, 4 bytes
0	1	1	0	1	REL	Read Error Logger
0	1	1	1	0	I4DR4	Increment MAR by 4, Data Read, 4 bytes
0	1	1	1	1	DR4	Data Read, 4 bytes
1	0	0	0	0	D	Decode next user instruction
1	0	0	0	1	@LOC	Update ILOC from CLOC
1	0	0	1	0	IRD	Instruction Read and Decode
1	0	0	1	1	-	No action
1	0	1	0	0	-	No action
1	0	1	0	1	I4	Increment MAR by 4
1	0	1	1	0	PW2	Privileged Write, 2 bytes
1	0	1	1	1	DW2	Data Write, 2 bytes
1	1	0	0	0	LSSTD	Load Shared Segment Table Descriptor
1	1	0	0	1	LPSTD	Load Process Segment Table Descriptor
1	1	0	1	0	I1DW1	Increment MAR by 1, Data Write, 1 byte
1	1	0	1	1	DW1	Data Write, 1 byte
1	1	1	0	0	PW4	Privileged Write, 4 bytes
1	1	1	0	1	I4DW4	Increment MAR by 4, Data Write, 4 bytes
1	1	1	1	0	TEL	Test Error Logger (Write Error Byte)
1	1	1	1	1	DW4	Data Write, 4 bytes

D — The previously fetched user instruction is decoded. Faults occurring as a result of any memory operations, which were part of the instruction fetch, are enabled at decode time. Decode may occur only once in each instruction emulation. No MC operations may precede the D operation, with the exception of IR. The D function must be specified to allow interrupts to occur.

DR1 — One byte of data is read from the main memory location addressed by the current contents of MAR. This data replaces RMDR bits 24:31. The top three bytes of RMDR are forced to zero.

DR2 — Two bytes of data are read from the main memory location addressed by the current contents of MAR. This address must lie on a halfword boundary, or an abort sequence occurs. The data fetched from memory replaces the contents of RMDR, bits 16:31. RMDR bits 0:15 are forced to agree with bit 16.

DR2IB — Two bytes of data are read from the instruction buffer at the address pointed to by CLOC. This data replaces the contents of RMDR, bits 16:31. Bits 0:15 of RMDR are forced to agree with bit 16. If the DR2IB operation invalidates the IB by reading past the end of valid data in the IB, a buffer refill from memory is initiated. The DR2IB operation waits until the refill is complete. DR2IB advances CLOC by 2.

DR4 — Four bytes of data are read from the main memory location addressed by the current contents of MAR. This address must lie on a fullword boundary, or an abort sequence occurs. The data fetched from memory replaces the contents of RMDR.

DR4IB — Four bytes of data are read from the instruction buffer at the address pointed to by CLOC. This data replaces the contents of RMDR. If the DR4IB operation invalidates the IB by reading past the end of valid data in the IB, a buffer refill from memory is initiated. The DR4IB operation waits until the refill is complete. DR4IB advances CLOC by 4.

DW1 — The least significant byte of data in WMDR replaces the byte in main memory addressed by the current contents of MAR.

DW2 — The least significant 2 bytes of data in WMDR are written to memory at the location addressed by the current contents of MAR. This address must lie on a halfword boundary, or an abort sequence occurs.

DW4 — Four bytes of data in WMDR replace the contents of the fullword in main memory addressed by the current contents of MAR. This address must lie on a fullword boundary, or an abort sequence occurs.

I1DR1 The contents of MAR are incremented by one, and the byte in main memory addressed by the new MAR contents replaces the current contents of RMDR. The top three bytes of RMDR are forced to zero.

I1DW1 The contents of MAR are incremented by one, and the rightmost byte of data in WMDR replaces the byte in main memory addressed by the new contents of MAR.

I4 The contents of MAR are incremented by 4.

I4DR4 The contents of MAR are incremented by 4, and the fullword (four bytes) in main memory addressed by the new MAR contents replaces the contents of RMDR. This address must lie on a fullword boundary, or an abort sequence occurs.

I4DW4 The contents of MAR are incremented by 4, and the contents of WMDR replace the four bytes in main memory addressed by the new MAR contents. This address must lie on a fullword boundary, or an abort sequence occurs.

IR The user instruction pointed to by the current contents of CLOC is read, and the contents of CLOC replace the contents of ILOC. This MC option is usually followed by a D (Decode) option. CLOC is incremented by 2 for each instruction halfword read.

IRD The user instruction pointed to by the current contents of CLOC is read. The contents of CLOC are copied to ILOC, and the just-read instruction is decoded. CLOC is incremented by two for each instruction halfword read. This operation performs as the IR and D operations.

LPSTD Memory address translation is disabled. The fullword process segment table descriptor addressed by the current contents of MAR is loaded to prepare for enabling memory address translation. The STE register stacks in the MAT are invalidated. The instruction buffer is invalidated. The address of the PSTD must lie on a fullword boundary, or an abort sequence occurs.

LSSTD Memory address translation is disabled. The shared segment table descriptor addressed by the current contents of MAR is loaded to prepare for enabling memory address translation. The STE register stacks in the MAT are not invalidated; therefore, LSSTD must be followed by LPSTD before attempting MAT translation. The instruction buffer is invalidated. The address of the SSTD must lie on a fullword boundary, or an abort sequence occurs.

PR2 Memory address translation is disabled, and the halfword in main memory addressed by the current contents of MAR replaces the contents of RMDR, bits 16:31. Bits 0:15 of RMDR are forced to agree with bit 16. This address must lie on a halfword boundary, or an abort sequence occurs.

PR4 Memory address translation is disabled, and the fullword in main memory addressed by the current contents of MAR replaces the contents of RMDR. The address in MAR must lie on a fullword boundary, or an abort sequence occurs.

PW2 Memory address translation is disabled and the contents of WMDR bits 16:31, replace the contents of the halfword in main memory addressed by the current contents of MAR. This address must lie on a halfword boundary, or an abort sequence occurs.

PW4 Memory address translation is disabled, and the 4 data bytes in WMDR replace the contents of the fullword location in main memory addressed by the current contents of MAR. This address must lie on a fullword boundary, or an abort sequence occurs.

RAS The halfword in main memory addressed by the current contents of MAR replaces the contents of RMDR, bits 16:31. Bits 0:15 are forced to agree with bit 16. Bit 16 of the data is set as the data is written back to main memory. This address must lie on a halfword boundary, or an abort sequence occurs.

REL The error logger, at the address corresponding to the contents of MAR, is interrogated. Error logger data replaces the contents of RMDR.

RFAULT Any fault which may be latched in the processor is reset by this instruction. The RX format flip-flops are also reset. The instruction buffer is invalidated. RFAULT occurs at the end of the microinstruction cycle, after the destination register has been loaded.

TEL The contents of WMDR bits 24:31 replace the byte in main memory addressed by the current contents of MAR. The Error Corection Code (ECC) bits corresponding to the fullword in which the byte lies are not modified. A subsequent byte, halfword, or fullword fetch thus causes the data in the location and its ECC bits to disagree, and causes an MAIO abort if the machine malfunction interrupt is enabled by PSW18. This can be checked by an REL MC option in a subsequent instruction. TEL causes the corresponding data/instruction cache block to be invalidated, resulting in a main memory access for any subsequent read from that cache block.

@LOC The current contents of CLOC are copied to ILOC. This is useful for interrupt processing. CLOC may be the specified destination register in an instruction which also specifies @LOC: first @LOC occurs, then CLOC is loaded from the S bus.

All main memory control is conditional when used within Address Link and Register Link microinstructions. The control is only effected if the instruction does not result in a branch.

Interrupts may occur whenever the D option is specified, if armed and enabled.

Interrupts caused by faults while fetching data from memory or writing data to memory (called MAIO interrupts) are always armed, and may occur on any microinstruction if enabled. Halfword and fullword alignment fault interrupts cannot be disabled.

All increment functions are performed before the microinstruction terminates. Memory read and write functions start as soon as logically practical. However, the microprogram may use MAR or WMDR as a destination and then begin a memory read or write in the same microinstruction. I1DR1, I1DW1, I4, I4DR4, or I4DW4 may not be specified by a microinstruction which also specifies MAR as a destination register.

Following a memory read, instruction read (IR), an MAIO fault, or instruction buffer data fetch (DR2IB or DR4IB), no MC function may be specified before unloading RMDR. Any MC function may be specified simultaneously with the unloading of RMDR.

CHAPTER 3
SOURCE AND DESTINATION REGISTERS

The processor has 182 registers that are addressable by the microprogram. Most of these are available to the A, B, and S buses. Table 3-1 and the following paragraphs explain the exceptions and special cases.

TABLE 3-1 REGISTER ADDRESSES

HEX ADDRESS	S BUS	A BUS	B BUS	CATEGORY
00	0	0	0	
01	1	1	1	
02	2	2	2	
03	3	3	3	
04	4	4	4	
05	5	5	5	
06	6	6	6	USER'S
07	7	7	7	GENERAL
08	8	8	8	REGISTERS
09	9	9	9	IN SET
0A	10	10	10	SELECTED BY
0B	11	11	11	PSW 25:27
0C	12	12	12	
0D	13	13	13	
0E	14	14	14	
0F	15	15	15	
10	MR0	MR0	MR0	
11	MR1	MR1	MR1	
12	MR2	MR2	MR2	
13	MR3	MR3	MR3	MICRO-
14	MR4	MR4	MR4	REGISTERS
15	MR5	MR5	MR5	
16	MR6	MR6	MR6	
17	MR7	MR7	MR7	
18	YS	YS	YS	
19	YD	YD	YD	
1A	CLOC	YX	ILOC	SPECIAL
1B	WMDR	YDP1	RMDR	PURPOSE
1C	MAR	CLOC	MAR	
1D	PSW	PSW	YSI	
1E	YDI	ILOC	YDI	
1F	NULL	NULL	NULL	

Although the user's general registers, in the register set specified by PSW bits 25, 26 and 27, can be addressed directly by the microprogram, it is often more convenient to access the general register specified in the user's instruction without regard to its physical number. The symbolic addresses YD, YDP1, YS, and YX allow this to happen. Specifying YD causes the general register whose number appears in the YDI field of UIR (UIR bits 8:11) to be selected. Specifying YDP1 causes the odd member of the even/odd pair of general registers, one of whose number appears in the YDI field of UIR, to be selected. Designating YS causes the general register whose number appears in the YSI field of UIR (UIR bits 12:15) to be selected. Specifying YX is the same as specifying YS, except when the YSI field of UIR is zero, at which time all zeros are placed on the A bus. This automatic feature is used to develop the index value for the user-level RI1, RI2, RX, and RXRX format instructions.

When module 7 is specified, the scratchpad registers are selected in place of the user's general registers. User general registers and scratchpad registers cannot be selected by the same microinstruction.

On microinstructions that address the floating-point processor module, the corresponding floating-point register is selected instead of a general register. YDP1 may not be used as a source register if the floating-point processor is used.

Selecting YDI or YSI as a source causes the corresponding field of IR (i.e., YD or YS) to be placed on bits 28:31 of the B bus. The high-order 28 bits of the B bus are zero.

Specifying NULL as a source on the A or B bus causes the corresponding bus to be set to zero. Specifying NULL as the S bus destination causes the data to be lost.

Designating RMDR or CLCC as a source, after a memory read operation, causes the processor to wait until the memory data becomes available. Following an Instruction Read and Decode function, RMDR participates in the formation of the effective address, if the user's instruction is one of the RX formats. Specifically, until MAR is loaded, any reference to RMDR as a source causes the second level index register (SX2) to be accessed if the instruction format is RX3. Otherwise, RMDR is accessed. Refer to the chapter on instruction execution for details.

Specifying CLCC, WMDR, or MAR as a destination, when a memory access is in progress, causes the processor to wait until the memory access is completed.

Specifying PSW as a destination immediately prior to a BALT, BALF, EXLT, or EXLF instruction causes the true/false decision to be based on the setting of PSW28:31, and not on the flags resulting from the ALU operation. All other conditional BAL or EXL test the resulting ALU flags (e.g., BALNZ).

Following an Instruction Read and Decode sequence, the effective second operand address for an RX1, RX2, or RX3 user-level instruction is calculated by the following microinstruction:

```
     A MAR,YX,RMDR
```

The RX flip-flops in the machine are conditioned according to the format of the last user instruction decoded, so that the correct address is formed. Any appropriate MC function may be specified in this "calculate address" microinstruction, except RFAULT. For example, WMDR may be loaded prior to a microinstruction which calculates an RX effective second operand address and specifies a memory write.

The MC functions IR, IRD, DR2IB, and DR4IB must not be specified in a microinstruction which also specifies CLOC as a source or destination. The "calculate address" instruction must not specify any of these MC functions, or an incorrect address may result. If MAR or WMDR is specified as the destination register in a microinstruction which also specifies one of these MC functions, unnecessary clock stops result in an increase in execution time of the instruction.

PSW must not be the specified destination register in a microinstruction which also specifies IR, IRD, DR2IB, or DR4IB, unless it is known that PSW bits 10 and 11 (LVL), and 21 (R/P) are not being changed from their prior states.

Specifying any MC function, in a microinstruction following one which specifies a memory reference, causes the processor to wait until the first MC function is complete before allowing the second MC function to proceed.

If MAR, WMDR, or CLOC is specified as the destination register in a microinstruction which also specifies a memory read or memory write MC function, the MC function does not proceed until the destination register has been loaded and is ready for use. MAR and CLOC cause the greatest delay, and WMDR the least. Each of the following examples performs the same function, yet the second requires 60 nanoseconds less time to execute.

```
SLOWER   L  WMDR,YD              DATA TO STORE
         A  MAR,YX,RMDR,DW4      CALCULATE ADDRESS AND STORE

FASTER   A  MAR,YX,RMDR          CALCULATE ADDRESS
         L  WMDR,YD,DW4          STORE DATA
```

To load MAR with a known address so that a memory operation can be performed using the specified address, it is necessary to reset the RX flip-flops which are adjusted according to the format of the last user instruction decoded. This can be done by loading MAR to itself, or by specifying the RFAULT MC option.

The condition code field of PSW can be manipulated by any addressed module unless PSW is the explicit destination or a condition code change was inhibited by the E bit in a prior microinstruction.

The bits of PSW that have hardware implications are:

PSW 10,11	Program memory access privilege level
PSW 13	Floating-point disable
PSW 17,20	External interrupt priority selection
PSW 18	Machine malfunction interrupt enable
PSW 19	Floating-point underflow interrupt enable (used by Floating-Point Processor)
PSW 21	Memory address translator enable
PSW 23	Protect mode enable
PSW 25,26,27	Register set selection
PSW 28,29,30,31	Condition code

CHAPTER 4
INSTRUCTION REPERTOIRE

4.1 INTRODUCTION

The instruction repertoire has been grouped by function in this chapter. Each instruction operation is presented in the following format:

1. An instruction word chart for each instruction, including mnemonic, operation code and operand designations, in the correct assembler format. The format type, an instruction diagram with operation code, and the location of all fields is also provided.

2. A description of instruction operation.

3. A diagram showing instruction operation.

4. A chart showing the possible resultant flags.

5. The execution time in nanoseconds. On all microinstructions, add 180 nanoseconds if I=1.

6. A programming note may be provided to add pertinent or clarifying information.

The symbols and abbreviations used in the instruction descriptions are defined as follows:

( ) [ ]	Parentheses or brackets. Read as "the contents of ..."
$\leftarrow$ $\rightarrow$	Arrow. Read as "is replaced by..." or "replaces..."
A	The A field. First operand register specification.
B	The B field. Second operand register specification.
S	The S field. Destination register specification.
(0:7)	A bit grouping within a word. Read as "Bits 0 through 7 inclusive".
B_E	The effective second operand. If the instruction format is RR Control or RR Transfer, the effective second operand is the contents of the register specified by B if the Indirect (I) bit is zero:

B_E = (B) if I=0

If the I bit is set, the effective second operand is the contents of the fullword control store location whose address is contained in the register specified by B:

B_E=[(B)] if I=1

If the instruction format is RR Imediate, then the effective second operand is the 12-bit data field if the I bit is zero:

B_E=DATA if I=0

If the I bit is set, the effective second operand is the contents of the fullword control store location whose address is the data field:

B_E=(DATA) if I=1

4.2 LOGICAL INSTRUCTIONS

The instructions described in this section are:

4.2.1	L	Load
	LX	Load and Transfer
	LI	Load Immediate
4.2.2	STR	Store to WCS
4.2.3	N	AND
	NX	AND and Transfer
	NI	AND Immediate
4.2.4	O	OR
	OX	OR and Transfer
	OI	OR Immediate
4.2.5	X	Exclusive OR
	XX	Exclusive OR and Transfer
	XI	Exclusive OR Immediate

4.2.1 Load

L S,B,I,E,MC [RR CONTROL]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26	27-31
0 0 1	0 1	I	S	1 1 1 1 1	0 0 0 1	B	0	E	MC

LX S,B,ADRS,I,C [RR TRANSFER]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26-31
0 0 1	0 0	I	S	1 1 1 1 1	0 0 0 1	B	C	PAGE ADDRESS

LI S,DATA,I [RR IMMEDIATE]

0-2	3-4	5	6-10	11-15	16-19	20-31
0 0 1	1 0	I	S	1 1 1 1 1	0 0 0 1	DATA

The second operand is loaded into the register specified by S.

L,LI: (S)$\leftarrow B_E$
LX : (S)$\leftarrow B_E$
then (RLC10:15)$\leftarrow$PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero

Programming Note

The Load instruction assembles as an Add instruction with a NULL A field.

Execution Times

L,LI,LX: 260

4.2.2 Store to WCS

STR A,B,E,MC [REGISTER WRITE]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	1 1	I	1 1 1 1 1	A	0 0 0 0	B	0	E	MC

The contents of the register specified by A are stored in the control store memory location whose address is in the register specified by B.

STR: (A)→[(B)]

Execution Time

STR: 420

4.2.3 AND

N S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	0 1 0 1	B	0	E	MC

NX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	0 1 0 1	B	C	PAGE ADDRESS

NI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	0 1 0 1	DATA

The logical product of the first and second operand replaces the contents of the register specified by S. The 32-bit result is formed on a bit-by-bit basis.

N,NI: (S)←(A) AND B_E

NX : (S)←(A) AND B_E

Then (RLC10:15)←PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is nct zero
0	0	1	0	Result is nct zero

Execution Times

N,NI,NX: 260

4.2.4 OR

O S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	0 1 1 1	B	0	E	MC

OX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	0 1 1 1	B	C	PAGE ADRESS

OI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	0 1 1 1	DATA

The logical sum of the first and second operands replaces the contents of the register specified by S. The 32-bit result is formed on a bit-by-bit basis.

O,OI: (S)←(A) OR B_E

OX : (S)←(A) OR B_E

then (RLC10:15)←PAGE ADDRESS

<u>Resulting Flags</u>

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is not zero
0	0	1	0	Result is not zero

<u>Execution Times</u>

O,OI,OX: 260

4.2.5 Exclusive OR

X S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	0 1 1 0	B	0	E	MC

XX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	0 1 1 0	B	C	PAGE ADDRESS

XI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	0 1 1 0	DATA

The logical difference between the first and second operands replaces the contents of the register specified by S. The 32-bit result is formed on a bit-by-bit basis.

X,XI: (S)←(A) XOR B_E

XX : (S)←(A) XOR B_E

then (BLC10:15)←PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is nct zero
0	0	1	0	Result is nct zero

Execution Times

X,XI,XX: 260

4.3 BRANCH/EXECUTE AND LINK INSTRUCTIONS

These instructions are programmed decisions providing entry to and return from subprograms, as well as testing the results of arithmetic, logical, and other machine operations.

Most processor operations result in setting the microflag register. The state of this flag register is testable with the Branch/Execute and Link on condition instructions.

The Execute and Link instructions allow conditional execution of a single, nonsequential microinstruction. No branch is actually taken, and control returns to the instruction following the Execute and Link.

The address plus one of the Branch/Execute and Link instruction is always saved in the specified link register, even if the condition for doing the Branch or Execute is not met.

In the Register Link format, NULL must be specified as the S bus register. This code is filled in automatically by the microassembler.

The instructions described in this section are:

4.3.1	BAL	Branch and Link
	BALA	Branch and Link and Arm Interrupts
	BALD	Branch and Link and Disarm Interrupts
	BALZ	Branch and Link on Zero
	BALNZ	Branch and Link on Not Zero
	BALL	Branch and Link on Less
	BALNL	Branch and Link on Not Less
	BALG	Branch and Link on Greater
	BALNG	Branch and Link on Not Greater
	BALV	Branch and Link on Overflow
	BALNV	Branch and Link on No Overflow
	BALC	Branch and Link on Carry
	BALNC	Branch and Link on No Carry
	BALT	Branch and Link on True CC Match
	BALF	Branch and Link on False CC Match
4.3.2	EXL	Execute and Link
	EXLA	Execute and Link and Arm Interrupts
	EXLD	Execute and Link and Disarm Interrupts
	EXLZ	Execute and Link on Zero
	EXLNZ	Execute and Link on Not Zero
	EXLL	Execute and Link on Less
	EXLNL	Execute and Link on Not Less
	EXLG	Execute and Link on Greater
	EXLNG	Execute and Link on Not Greater
	EXLV	Execute and Link on Overflow
	EXLNV	Execute and Link on No Overflow
	EXLC	Execute and Link on Carry
	EXLNC	Execute and Link on No Carry
	EXLT	Execute and Link on True CC Match
	EXLF	Execute and Link on False CC Match

4.3.1 Branch and Link

F ADDRESS (LINK),E,MC [ADDRESS LINK]

0	3	5	6	11	14	26	27 31
0 0 0	1 0	T	LINK	F	ADDRESS	E	MC

F (B)(LINK),E,MC [REGISTER LINK]

0	3	5	6	11	14	20	25	26	27 31
0 0 0	0 0	T	LINK	F		B		E	MC

where F =

	F			
BALZ	0	0	0	0
BALL	0	0	0	1
BALG	0	0	1	0
BALF	0	0	1	1
BALC	0	1	0	0
BALV	0	1	0	1
BAL	0	1	1	0
BALA	0	1	1	1
BALNZ	1	0	0	0
BALNL	1	0	0	1
BALNG	1	0	1	0
BALT	1	0	1	1
BALNC	1	1	0	0
BALNV	1	1	0	1
BALD	1	1	1	1

The address of the next sequential microinstruction replaces the contents of the register specified by LINK; then a transfer is conditionally taken to the address specified. In the address link format, the address field of the instruction contains the branch address. In the register link format, the branch address is contained in the register specified by B. This format is used to return from subroutines.

Tested Condition True

(LINK)←(RLC4:15)+1

(RLC4:15)←ADDRESS [Address Link]

(RLC4:15)←(E) [Register Link]

Tested Condition False

(LINK)←(RLC4:15)+1

(RLC4:15)←(RLC4:15)+1

Programming Notes

For the BALT and BALF instructions, a logical AND is performed between each bit in the condition code field of PSW and the M1 field of the user's instruction (YDI, or IR 8:11). If any resultant bit is a one, the BALT instruction branches and the BALF instruction does not. If all resultant bits are zero, the BALF instruction branches and the BALT instruction does not.

If any memory control function is specified in the MC field, the function is performed only if the branch is not taken. Similarly, if Decode is specified, the Decode function occurs only if no branch is taken.

The BALA and BALD instructions are used respectively to arm and disarm the interrupt system. If an interrupt is to be allowed while executing the BALA instruction, Decode (D or IRD) must be specified in the MC field.

Execution Time

260

4.3.2 Execute and Link

F ADDRESS (LINK),E,MC [ADDRESS LINK]

0	3	5	6	11	14	26	27 31
0 0 0	1 1	T	LINK	F	ADDRESS	E	MC

F (B)(LINK),E,MC [REGISTER LINK]

0	3	5	6	11	14	20	25	26	27 31
0 0 0	0 1	T	LINK	F		B		E	MC

where F =

	T	F
EXLZ	0	000
EXLL	0	001
EXLG	0	010
EXLF	0	011
EXLC	0	100
EXLV	0	101
EXL	0	110
EXLA	0	111
EXLNZ	1	000
EXLNL	1	001
EXLNG	1	010
EXLT	1	011
EXLNC	1	100
EXLNV	1	101
EXLD	1	111

The address of the next sequential microinstruction replaces the contents of the register specified by LINK; then if the condition is met, the instruction at the specified address is executed. Any instruction may be executed including other execute instructions. When the executed instruction is completed, the processor continues with the microinstruction following the Execute and Link.

Tested Condition True

(LINK)←(RLC4:15)+1

Do instruction at ADDRESS [Address Link]
Do instruction at (B) [Register Link]
(RLC4:15)←(RLC4:15)+1

Tested Condition False

(LINK)←(RLC4:15)+1

(RLC4:15)←(RLC4:15)+1

Programming Notes

For the EXLT and EXLF instructions, a logical AND is performed between each bit in the condition code field of the PSW and the M1 field of the user's instruction (YDI, or IR 8:11). If any resultant bit is a one, the EXLT instruction executes the indicated instruction, and EXLF does not. If all resultant bits are zero, the EXLF instruction executes the indicated instruction, and EXLT does not.

If the EXL instructions execute an instruction which attempts to cause a branch or transfer, no branch or transfer occurs.

If any memory control function is specified in the MC field of this instruction, the function is performed only if the indicated instruction is not executed.

If an interrupt is to be allowed while executing the EXLA instruction, Decode (D or IRD) must be specified in the MC field.

Execution Time

260 + executed instruction

4.4 SHIFT/ROTATE INSTRUCTIONS

The Shift and Rotate instructions provide for arithmetic and logical use of information contained in the processor registers. Bits shifted out of the high or low end of a register are passed through the Carry flag (C). After a shift instruction, the last bit which was shifted out is contained in the Carry flag.

A shift of zero positions causes the G and L flags to be set, on the basis of the halfword or fullword result, with no alteration to the data contained in the register. The Carry and Overflow flags are zero, in this case.

The instructions described in this section are:

4.4.1	SLL	Shift Left Logical
	SLLX	Shift Left Logical and Transfer
	SLLI	Shift Left Logical Immediate
4.4.2	SLHL	Shift Left Halfword Logical
4.4.3	SRL	Shift Right Logical
	SRLX	Shift Right Logical and Transfer
	SRLI	Shift Right Logical Immediate
4.4.4	SRHL	Shift Right Halfword Logical
4.4.5	SLA	Shift Left Arithmetic
	SLAX	Shift Left Arithmetic and Transfer
	SLAI	Shift Left Arithmetic Immediate
4.4.6	SLHA	Shift Left Halfword Arithmetic
4.4.7	SRA	Shift Right Arithmetic
	SRAX	Shift Right Arithmetic and Transfer
	SRAI	Shift Right Arithmetic Immediate
4.4.8	SRHA	Shift Right Halfword Arithmetic
4.4.9	RLL	Rotate Left Logical
	RLLX	Rotate Left Logical and Transfer
	RLLI	Rotate Left Logical Immediate
4.4.10	RRL	Rotate Right Logical
	RRLX	Rotate Right Logical and Transfer
	RRLI	Rotate Right Logical Immediate

4.4.1 Shift Left Logical

SLL S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 0 0 1	B	0	E	MC

SLLX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 0 0 1	B	C	PAGE ADDRESS

SLLI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 0 0 1	DATA

The contents of the register specified by A are shifted left the number of bit positions specified by the least significant five bits of the second operand. The result replaces the contents of the register specified by S.

High order bits shifted out of position 0 are shifted through the carry flag, then lost. Zeros shift into the low order bit position.

SLL,SLLI: $S \xleftarrow[B_E(27:31)]{L} (A)$

SLLX: $S \xleftarrow[B_E(27:31)]{L} (A)$

then RLC10:15←PAGE ADDRESS if C=0 or Carry =0

RLC4:15←(RLC4:15)+1 if C = 1 and Carry = 1

Resulting Flags

C	V	G	L	
	0	0	0	Result is zero
	0	0	1	Result is less than zero
	0	1	0	Result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

Execution Times (n = number of shifts)

SLL,SLLI:	430+60n
SLLX (no transfer):	560+60n
SLLX (transfer):	430+60n

4.4.2 Shift Left Halfword Logical

SLHL S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 0 0 1	B	1	E	MC

The least significant 16 bits of the register specified by A are shifted left the number of bit positions specified by the least significant 4 bits of the second operand. The result replaces the least significant 16 bits of the register specified by S. The most significant 16 bits of the register specified by A replace the most significant 16 bits of the register specified by S. Bits shifted out of position 16 are shifted through the carry flag and then lost. Zeros shift into the low order bit position.

SLHI S0:15 ← (A0:15)

S16:31 ←L— (A16:31)
B_E(28:31)

Resulting Flags

C	V	G	L	
	0	0	0	Halfword result is zero
	0	0	1	Halfword result is less than zero
	0	1	0	Halfword result is greater than zero
0				Last bit shifted out of bit 16 was a zero
1				Last bit shifted out of bit 16 was a one

Execution Times (n = number of shifts)

SLHI: 430+60n

4.4.3 Shift Right Logical

SRL S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 0 0 0	B	0	E	MC

SRLX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 0 0 0	B	C	PAGE ADDRESS

SRLI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 0 0 0	DATA

The contents of the register specified by A are shifted right the number of bit positions specified by the least significant 5 bits of the second operand. Low order bits shifted out of position 31 are shifted through the carry flag and then lost. Zeros shift into position 0.

SRL,SRLI: $S \xleftarrow[B_E(27:31)]{R} (A)$

SRLX: $S \xleftarrow[B_E(27:31)]{R} (A)$

then RLC10:15 ← PAGE ADDRESS if C=0 or Carry=0

RLC4:15 ← (RLC4:15)+1 if C=1 and Carry=1

Resulting Flags

C	V	G	L	
	0	0	0	Result is zero
	0	0	1	Result is less than zero
	0	1	0	Result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

Execution Times (n = number of shifts)

SRL,SRLI:	430+60n
SRLX (no transfer):	560+60n
SRLX (transfer):	430+60n

4.4.4 Shift Right Halfword Logical

SRHL S,A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
0 0 1	0 1	I	S	A	1 0 0 0	B	1	E	MC

The least significant 16 bits of the register specified by A are shifted right the number of bit positions specified by the least significant 4 bits of the second operand. The result replaces the least significant 16 bits of the register specified by S. The most significant 16 bits of the register specified by A replace the most significant 16 bits of the register specified by S. Bits shifted out of position 31 are shifted through the carry flag, and then lost. Zeros shift into position 16.

SRHL: S0:15 ← (A0:15)

$$S16{:}31 \xleftarrow[B_E(28:31)]{R} (A16{:}31)$$

<u>Resulting Flags</u>

C	V	G	L	
	0	0	0	Halfword result is zero
	0	0	1	Halfword result is less than zero
	0	1	0	Halfword result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

<u>Execution Times</u> (n = number of shifts)

SRHL: 430+60n

4.4.5 Shift Left Arithmetic

SLA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 1 0 1	B	0	E	MC

SLAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 1 0 1	B	C	PAGE ADDRESS

SLAI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 1 0 1	DATA

The contents of the register specified by A are shifted left the number of bit positions specified by the least significant 5 bits of the second operand. Only bits 1:31 participate in the shift; bit 0 remains unchanged. High order bits shifted out of position 1 are shifted through the carry flag, and then lost. Zeros shift into the low order bit position.

SLA,SLAI: $S \leftarrow (A0)$

$$S1{:}31 \xleftarrow[B_E(27:31)]{L} (A1{:}31)$$

SLAX: $S0 \leftarrow (A0)$

$$S1{:}31 \xleftarrow[B_E(27:31)]{L} (A1{:}31)$$

then $RLC10{:}15 \leftarrow$ PAGE ADDRESS if C=0 or Carry=0

$RLC4{:}15 \leftarrow (RLC4{:}15)+1$ if C=1 and Carry=1

Resulting Flags

C	V	G	L	
	0	0	0	Result is zero
	0	0	1	Result is less than zero
	0	1	0	Result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

Execution Times (n = number of shifts)

SLA,SLAI:	430+60n
SLAX (no transfer):	560+60n
SLAX (transfer):	430+60n

4.4.6 Shift Left Halfword Arithmetic

SLHA S,A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
0 0 1	0 1	I	S	A	1 1 0 1	B	1	E	MC

The least significant 15 bits of the register specified by A are shifted left the number of bit positions specified by the least significant 4 bits of the second operand. The result replaces the least significant 15 bits of the register specified by S. The most significant 17 bits of the register specified by A replace the most significant 17 bits of the register specified by S. Bits shifted out of position 17 are shifted through the carry flag, and then lost. Zeros shift into position 31.

SLHA: S0:16 ← (A0:15)

$$S17{:}31 \xleftarrow[B_E(28:31)]{L} (A17{:}31)$$

<u>Resulting Flags</u>

C	V	G	L	
	0	0	0	Halfword result is zero
	0	0	1	Halfword result is less than zero
	0	1	0	Halfword result is greater than zero
0				Last bit shifted out of position 17 was a zero
1				Last bit shifted out of position 17 was a one

<u>Execution Time</u> (n = number of shifts)

SLHA: 430+60n

4.4.7 Shift Right Arithmetic

SRA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 1 0 0	B	0	E	MC

SRAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 1 0 0	B	C	PAGE ADDRESS

SRAI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 1 0 0	DATA

The contents of the register specified by A are shifted right the number of bit positions specified by the least significant 5 bits of the second operand. The result replaces the contents of the register specified by S. Only bits 1:31 participate in the shift; bit 0 remains unchanged and is propagated right into position 1 on each shift. Low order bits shift through the carry flag and are then lost.

SRA,SRAI: $S0 \leftarrow (A0)$

$$S1{:}31 \xleftarrow[B_E(27{:}31)]{R} (A1{:}31)$$

SRAX: $S0 \leftarrow (A0)$

$$S1{:}31 \xleftarrow[B_E(27{:}31)]{R} (A1{:}31)$$

then $RLC10{:}15 \leftarrow$ PAGE ADDRESS if C=0 or Carry=0

$RLC4{:}15 \leftarrow (RLC4{:}15)+1$ if C=1 and Carry=1

Resulting Flags

C	V	G	L	
	0	0	0	Result is zero
	0	0	1	Result is less than zero
	0	1	0	Result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

Execution Times (n= number of shifts)

SRA,SRAI:	430+60n
SRAX (no transfer):	560+60n
SRAX (transfer):	430+60n

4.4.8 Shift Right Halfword Arithmetic

SRHA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 1 0 0	B	1	E	MC

The least significant 15 bits of the register specified by A are shifted right the number of bit positions specified by the least significant 4 bits of the second operand. The result replaces the least significant 15 bits of the register specified by S. The most significant 17 bits of the register specified by A replace the most significant 17 bits of the register specified by S. Bit 16 is propagated right into bit position 15 on each shift. Bits shifted out of position 31 are shifted through the carry flag and then lost.

SRHA S0:16 ← (A0:16)

S17:31 $\xleftarrow[B_E(28:31)]{R}$ (A17:31)

<u>Resulting Flags</u>

C	V	G	L	
	0	0	0	Halfword result is zero
	0	0	1	Halfword result is less than zero
	0	1	0	Halfword result is greater than zero
0				Last bit shifted out was a zero
1				Last bit shifted out was a one

<u>Execution Time</u> (n = number of shifts)

SRHA: 430+60n

4.4.9 Rotate Left Logical

RLL S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 0 1 1	B	0	E	MC

RLLX S,A,B,ADRS,I,C [RR CONTROL]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 0 1 1	B	C	PAGE ADDRESS

RLLI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 0 1 1	DATA

The contents of the register specified by A are shifted left, end around, the number of bit positions specified by the least significant 5 bits of the second operand. Bits shifted out of position 0 are shifted into position 31.

RLL,RLLI: $S0{:}31 \xleftarrow[B_E(27:31)]{L} (A0{:}31)$

RLLX: $S0{:}31 \xleftarrow[B_E(27:31)]{L} (A0{:}31)$

then RLC10:15 ← PAGE ADDRESS, because Carry=0 always

Resulting Flags

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is not zero
0	0	1	0	Result is not zero

Execution Times (n = number of shifts)

RLL,RLLI,RLLX: 430+60n

4.4.10 Rotate Right Logical

RRL S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27–31
0 0 1	0 1	I	S	A	1 0 1 0	B	0	E	MC

RRLX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26–31
0 0 1	0 0	I	S	A	1 0 1 0	B	C	PAGE ADDRESS

RRLI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20–31
0 0 1	1 0	I	S	A	1 0 1 0	DATA

The contents of the register specified by A are shifted right, end around, the number of bit positions specified by the least significant 5 bits of the second operand. Bits shifted out of position 31 are shifted intc positicn 0.

RRL,RRLI: $S0{:}31 \xleftarrow[B_E(27:31)]{R} (A0{:}31)$

RRLX: $S0{:}31 \xleftarrow[B_E(27:31)]{R} (A0{:}31)$

then RLC10:15 ← PAGE ADDRESS, because Carry=0 always

<u>Resulting Flags</u>

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is nct zero
0	0	1	0	Result is not zero

<u>Execution Times</u> (n = number of shifts)

RRL,RRLI,RRLX: 430+60n

4.5 FIXED-POINT ARITHMETIC INSTRUCTIONS

The Fixed-Point Arithmetic Instructions provide for addition, subtraction, multiplication and division of fixed-point data contained in the processor registers. The instructions described in this section are:

4.5.1	A	Add
	AX	Add and Transfer
	AI	Add Immediate
4.5.2	AINC	Add and Increment
	AINCX	Add and Increment and Transfer
4.5.3	S	Subtract
	SX	Subtract and Transfer
	SI	Subtract Immediate
4.5.4	SDEC	Subtract and Decrement
	SDECX	Subtract and Decrement and Transfer
4.5.5	M	Multiply
	MX	Multiply and Transfer
	MI	Multiply Immediate
4.5.6	D	Divide
	DX	Divide and Transfer
	DI	Divide Immediate

4.5.1 Add

A S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	0 0 0 1	B	0	E	MC

AX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	0 0 0 1	B	C	PAGE ADDRESS

AI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	0 0 0 1	DATA

The second operand is algebraically added to the first operand. The sum replaces the contents of the register specified by S.

A,AI: S←(A) + B

AX: S←(A) + B

then RLC10:15←PAGE ADDRESS if C=0 or Carry=0

RLC4:15←(RLC4:15)+1 if C=1 and Carry=1

Resulting Flags

C	V	G	L	
		0	0	Sum is zerc
		0	1	Sum is less than zero
		1	0	Sum is greater than zero
	1			Overflow
1				Carry

Execution Times

A,AI: 260
AX (no transfer): 405
AX (transfer): 260

4.5.2 Add and Increment

AINC S,A,B,I,E,MC [RR CONTROL]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26	27-31
0 0 1	0 1	I	S	A	0 0 1 1	B	0	E	MC

AINCX S,A,B,ADRS,I [RR TRANSFER]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26-31
0 0 1	0 0	I	S	A	0 0 1 1	B	C	PAGE ADDRESS

The second operand is algebraically added with the first operand and a forced carry-in of one. The sum replaces the contents of the register specified by S.

AINC: S ← (A)+B_E+1

AINCX: S ← (A)+B_E+1

then RLC10:15 ← PAGE ADDRESS if C=0 or Carry=0

RLC4:15 ← (RLC4:15)+1 if C=1 and Carry=1

Resulting Flags

C	V	G	L	
		0	0	Sum is zero
		0	1	Sum is less than zero
		1	0	Sum is greater than zero
	1			Overflow
1				Carry

Programming Note

Multiple precision additicn operations require a carry forward from the least significant to the most significant operands. The following example shows a dcuble word add operation.

```
*
*  MRO AND MR1 CONTAIN THE 64-BIT FIRST OPERAND
*  MR2 AND MR3 CONTAIN THE 64-BIT SECOND OPERAND
*  THE 64-BIT RESULT IS RETURNED IN MRO and MR1
*
START   AX      MR1,MR1,MR3,SUM2,C   SUM LOW OPERANDS FIRST
*                                    TRANSFER IF NO CARRY, ELSE
*                                    FALL THROUGH, SUMMING
        AINCX   MRO,MRO,MR2,SUM3     HIGH OPERANDS THEN ADD ONE,
                                     SKIP TO SUM3.
*
SUM2    A       MRO,MRO,MR2          SUM HIGH OPERANDS
SUM3    EQU     *                    (MRO,MR1)=64-BIT RESULT
```

Execution Times

```
AINC:                  260
AINCX (no transfer):   405
AINCX (transfer):      260
```

4.5.3 Subtract

S S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	0 0 0 0	B	0	E	MC

SX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	0 0 0 0	B	C	PAGE ADDRESS

SI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	0 0 0 0	DATA

The second operand is algebraically subtracted from the first operand. The difference replaces the contents of the register specified by S.

S,SI: $S \leftarrow (A) - B_E$

SX: $S \leftarrow (A) - B_E$
then RLC10:15 ← PAGE ADDRESS if C=0 or Carry=0

RLC4:15 ← (RLC4:15)+1 if C=1 and Carry=1

<u>Resulting Flags</u>

C	V	G	L	
		0	0	Difference is zero
		0	1	Difference is less than zero
		1	0	Difference is greater than zero
	1			Overflow
1				Borrow

<u>Execution Times</u>

S,SI: 260
SX (no transfer): 405
SX (transfer): 260

4.5.4 Subtract and Decrement

SDEC S,A,B,I,E,MC [RR CONTROL]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26	27-31
0 0 1	0 1	I	S	A	0 0 1 0	B	0	E	MC

SDECX S,A,B,ADRS,I,C [RR TRANSFER]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26-31
0 0 1	0 0	I	S	A	0 0 1 0	B	C	PAGE ADDRESS

The second operand and a forced carry-in of one are subtracted from the first operand. The result replaces the contents of the register specified by S.

SDEC: $S \leftarrow (A)-B_E-1$

SDECX: $S \leftarrow (A)-B_E-1$

then $RLC10:15 \leftarrow$ PAGE ADDRESS if C=0 or Carry=0

$RLC4:15 \leftarrow (RLC4:15)+1$ if C=1 and Carry=1

Resulting Flags

C	V	G	L	
		0	0	Difference is zero
		0	1	Difference is less than zero
		1	0	Difference is greater than zero
	1			Overflow
1				Carry

Programming Note

See Add and Increment

Execution Times

SDEC: 260
SDECX (no transfer): 405
SDECX (transfer): 260

4.5.5 Multiply

M S,A,B,I,E,MC [RR CONTROL]

0	3	5 6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 1 1 0	B	0	E	MC

MX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5 6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 1 1 0	B	C	PAGE ADDRESS

MI S,A,DATA,I [RR IMMEDIATE]

0	3	5 6	11	16	20 31
0 0 1	1 0	I	S	A	1 1 1 0	DATA

The 32-bit second operand is multiplied by the contents of the first operand register. The 32 most significant product bits replace the contents of the register specified by S. The 32 least significant product bits replace the contents of the first operand register, the register specified by A. The S field must specify an even numbered register. The A field must specify the next sequential register, an odd number. The sign of the product is determined by the rules cf algebra.

M,MI: (S,A)←(A) * B_E

MX: (S,A)←(A) * B_E

then RLC10:15←PAGE ADDRESS

Resulting Flags

C	V	G	L
0	0	0	0

Execution Times

M,MI,MX: 2680/3290/3900 minimum/average/maximum

4.5.6 Divide

D S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 0 1	0 1	I	S	A	1 1 1 1	B	0	E	MC

DX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 0 1	0 0	I	S	A	1 1 1 1	B	C	PAGE ADDRESS

DI S,A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
0 0 1	1 0	I	S	A	1 1 1 1	DATA

The 64-bit dividend contained in the registers specified by S and A, an even/odd pair, is divided by the 32-bit second operand. The S field must specify an even numbered register and the A field must specify the next sequential odd register. The resulting 31-bit quotient with sign replaces the contents of the register specified by A and the 31-bit remainder with sign replaces the contents of the register specified by S. The sign of the quotient is determined by the rules of algebra; the sign of the remainder equals the sign of the dividend.

D,DI: $A \leftarrow (S,A)/B_E$

$S \leftarrow$ Remainder

DX: $A \leftarrow (S,A)/B_E$

$S \leftarrow$ Remainder

then RLC10:15 $\leftarrow$ PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Ncrmal
0	1	0	0	Divide fault

Programming Ncte

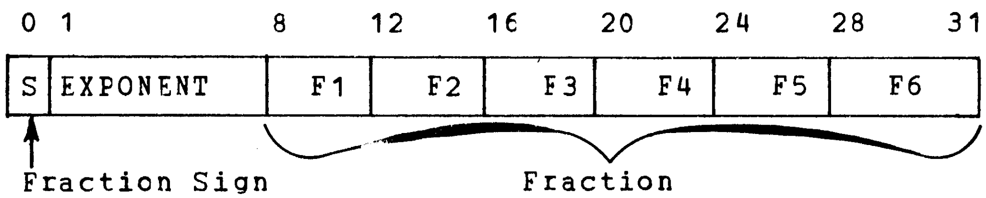
A quotient more positive than '7FFF FFFF' or more negative than '8000 0000' causes the division to be aborted with the V flag set and an unpredictable remainder in S. The register specified by A is unchanged. Attempted division by zero results in a divide fault.

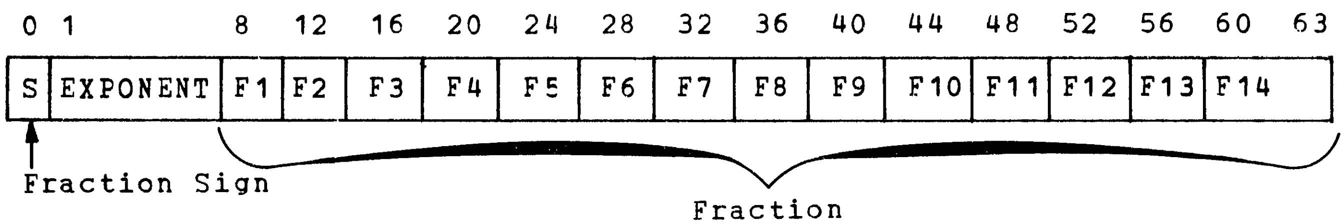
Execution Times

D,DI,DX: 4700

4.6 FLOATING-POINT INSTRUCTIONS

These instructions provide for the manipulation of single-precision and double-precision floating-point data. A floating-point quantity consists of a signed exponent and a signed magnitude fraction. The 7-bit exponent is expressed in excess 64 notation and can range in actual value from +63 through zero to -64. The value of the exponent field is that power of 16 by which the fraction field is multiplied. The 24- or 56-bit fraction is expressed as a hexadecimal number having a radix point to the left of the high order fraction digit. Bit 0 of the fullword or double word is the sign bit of the fraction.





4.6.1 Normalization

A normalized floating-point number for this processor is one in which the most significant digit of the mantissa is nonzero. In the preceding illustrations, digit F1 is nonzero if the number is normalized. If the floating-point number is not normalized, the normalization process consists of shifting the fraction field and any guard digits to the left hexadecimally (four bits at a time), until the most significant digit of the field is nonzero. The exponent is decremented by one for each shift required. Exponent underflow occurs if the exponent is already zero when it must be decremented. All floating-point arithmetic operations require normalized operands for consistent results. The result of a floating-point arithmetic operation is always normalized by the floating point processor.

4.6.2 Equalization

Equalization of two operands consists of shifting the fraction field of the operand with the smaller exponent to the right hexadecimally (four bits at a time), while incrementing the exponent of the operand by one. This process is repeated until the exponents of both operands are equal. The effect is to align the radix points of the two operands before performing an addition or subtraction. Data shifted from the lower-order digit of the operand is not lost, but is shifted into guard digits which participate in the subsequent floating point processor operation.

4.6.3 Guard Digits and R*-Rounding

When a floating-point result has been formed, it consists of a sign, an exponent, and a fraction field, as well as a number of guard digits containing the lower-order fraction digits resulting from the floating-point operation. Before the result is copied to the destination, it is rounded to provide improved accuracy.

R*-rounding is performed by the floating point processor as follows. The contents of the guard digits are tested. If the most significat guard digit is seven or less, no rounding is performed. If the most significant guard digit is eight, and all other guard digits are zero, then the least significant bit of the final result is forced to one. If the most significant guard digit is eight and another guard digit is nonzero, or if the most significant guard digit is greater than eight, one is added to the fraction field of the result to form the final result. If this addition causes a carry out of the fraction field, the exponent is incremented by one, and fraction digit F1 is set to one, while all other fraction digits are set to zero.

4.6.4 Effect of Current PSW

In the event of exponent overflow in the final result of a floating-point operation, the destination register is not modified; in effect, it did not participate in the operation. The flags returned by the floating point processor in this case include the V flag, and either the C, G, or L flag. The PSW has no effect in the case of exponent overflow.

Should exponent underflow occur in the final result of a floating-point operation, PSW bit 19 is tested. If bit 19 is zero, then zero is copied to the destination register. If bit 19 is set, the destination register is not modified; in effect, it did not participate in the operation. The floating point processor returns the V flag and no other flags in the event of exponent underflow.

The floating point processor is a standard plug-in module to the processor. A unique set of 35 microinstructions is provided to access the floating point processor (module number 6).

Figure 4-1 shows a block diagram of the floating point processor, which is situated between the 32-bit S bus and the 32-bit B bus. The A bus does not connect to the floating point processor. The floating point processor contains its own set of eight 32-bit single-precision registers and eight 64-bit double-precision registers.

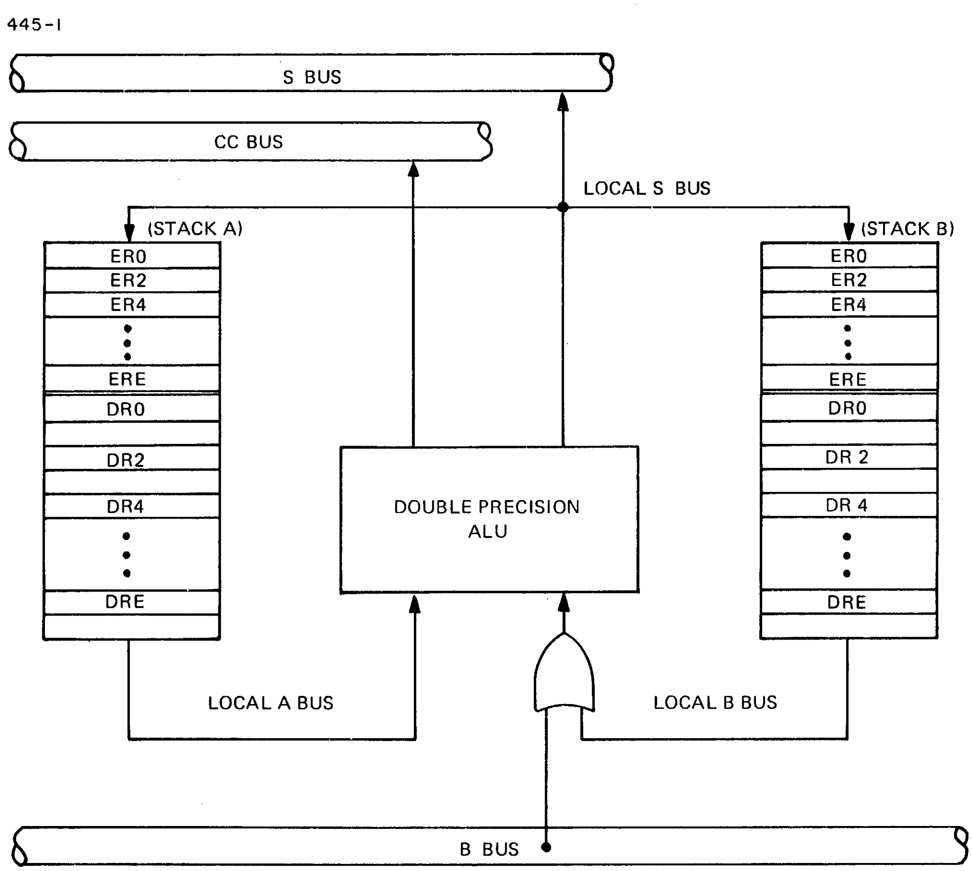


Figure 4-1 Floating-Point Processor (FPP) Block Diagram

In microinstructions directed to the floating point processor, references to the user's general registers, either directly or via the YD or YS fields of the user's instruction, cause the corresponding single-precision floating-point register or half (32 bits) of a double-precision floating-point register to be accessed. The microinstruction distinguishes whether a single-precision or a double-precision operation is to be performed. Single-precision operations can only use the single-precision registers and double-precision operations can only use the double-precision registers.

The two halves of a double-precision register are read using an even/odd addressing scheme. For example, reading double register 2 selects the most significant 32 bits of double register 2, and reading double register 3 selects the least significant 32 bits of double register 2. When writing to the double-precision registers, the least significant register address bit is ignored. The floating point processor handles the data steering, taking the first 32-bit operand to be the most significant half and the second 32-bit operand to be the least significant half of the 64-bit argument.

4.6.5 Floating-Point Processor (FPP) Autonomous Operation

The Floating-Point Processor operates in a fully autonomous mode having its own internal A, B, and S buses. Given a load, add, subtract, multiply, or divide operation, the floating-point module performs the function asynchronous of other processor activity. The microprogram is free to perform other functions while the floating-point module is finishing its task. If the microprogram attempts to test the result of a floating-point operation or begin another floating-point operation before the last one is completed, the processor stops until the prior function is completed before starting the next function.

This feature has an impact on determining execution times. The average execution time shown for divide single precision, for example, is 3395 nanoseconds. In practice, assuming the floating-point module is not busy, the microinstruction that initiates the divide takes only 260 nanoseconds. The processor immediately begins the next sequential microinstruction. The floating-point module is working independently and is busy for the next 3135 nanoseconds (3395-260). If this microinstruction does not reference the floating-point module, the instruction is performed, and the next microinstruction is begun. This continues until a microinstruction is feteched that does access the floating-point module. At that time, the processor must wait out any of the divide execution time remaining, 3135 nanoseconds, minus the execution time of all intervening microinstructions. If there was any time left at all, an additional 60 nanoseconds must be added in for resynchronization.

If the E-bit is set in the microinstruction which starts the FPP, the processor stops until the operation is complete, and the correct flags have been generated. If the IRD MC function is specified in this microinstruction, then if division by zero is attempted, or if exponent overflow or underflow occurs in the final result, a floating-point interrupt occurs. The exponent underflow interrupt does not occur if PSW bit 19 is zero. When the interrupt does occur, the information necessary to service the fault is available in the flags, condition code, ILOC, and RMDP, Refer to section 8.3, Interrupt Support.

Implementation Note:

The Floating-Point Processor is normally strapped to respond as processor module 6. Consequently, the Floating-Point Processor microinstructions assemble with a module number of 6.

In order to allow microcode to be assembled for an FPP strapped as module number 4, the common microcode assembler (MICROCAL) has a pair of special pseudo-operations that cause the module number of an FPP directed microinstruction to be switched from module 6 to module 4 and vice versa.

The assembler is normally in the FPP module 6 mode. Consequently, an AER microinstruction normally assembles with a module number of 6. The appearance in the source program of a DFU4 pseudo-operation places the assembler in the FPP module 4 mode until a DFU6 pseudo-operation is encountered. While in the FPP module 4 mode (DFU4), all microinstructions directed to the FPP (AER, for example) assemble with a module number of 4.

A pseudo-operation is an instruction only to the assembler and, as such, causes no object code to be generated.

The instructions described in this section are:

4.6.5.1	RCC	Read Condition Code
	RCCX	Read Condition Code and Transfer
4.6.5.2	LE	Load Register Single Precision
	LEX	Load Register Single Precision and Transfer
	LEI	Load Register Single Precision Immediate
4.6.5.3	RRE	Read Register Single Precision
	RREX	Read Register Single Precision and Transfer
4.6.5.4	CER	Compare Register Single Precision
	CERX	Compare Register Single Precision and Transfer
4.6.5.5	AER	Add Register Single Precision
	AERX	Add Register Single Precision and Transfer
4.6.5.6	SER	Subtract Register Single Precision
	SERX	Subtract Register Single Precision and Transfer
4.6.5.7	MER	Multiply Register Single Precision
	MERX	Multiply Register Single Precision and Transfer
4.6.5.8	DER	Divide Register Single Precision
	DERX	Divide Register Single Precision and Transfer

4.6.5.9	LW	Load Word
	LWX	Load Word and Transfer
	LWI	Load Word Immediate
4.6.5.10	LD	Load Register Double Precision
	LDX	Load Register Double Precision and Transfer
	LDI	Load Register Double Precision Immediate
4.6.5.11	RRD	Read Register Double Precision
	RRDX	Read Register Double Precision and Transfer
4.6.5.12	CDR	Compare Register Double Precision
	CDRX	Compare Register Double Precision and Transfer
4.6.5.13	ADR	Add Register Double Precision
	ADRX	Add Register Double Precision and Transfer
4.6.5.14	SDR	Subtract Register Double Precision
	SDRX	Subtract Register Double Precision and Transfer
4.6.5.15	MDR	Multiply Register Double Precision
	MDRX	Multiply Register Double Precision and Transfer
4.6.5.16	DDR	Divide Register Double Precision
	DDRX	Divide Register Double Precision and Transfer

4.6.5.1 Read Condition Code

RCC S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	S	1 1 1 1 1	0 0 0 0	B	0	E	MC

RCCX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	S	1 1 1 1 1	0 0 0 0	B	C	PAGE ADDRESS

The flags that resulted from the last single-precision or double-precision floating-point operation are collected.

RCC: CCBUS←Floating-Point Flags
RCCX: CCBUS←Floating-Point Flags
then RLC10:15←PAGE ADDRESS

Resulting Flags

Determined by previous floating-point operation

Programming Note

The S and B fields are not used and should be NULL selected.

Execution Times

RCC:	260
RCCX (no transfer):	405
RCCX (transfer):	260

4.6.5.2 Load Register Single Precision

LE A,B,I,K,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	0 0 1 0	B	K	E	MC

LEX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	0 0 1 0	B	C	PAGE ADDRESS

LEI A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
1 1 0	1 0	I	1 1 1 1 1	A	0 0 1 0	DATA

This instruction loads the single-precision floating-point register specified by A in the following manner:

If a Load Word instruction did not precede this Load Register instruction, then the second operand presented by the Load Register instruction is the most significant 32 bits of a double-precision number. The least significant 32 bits of this number are forced to zero. If a Load Word instruction did precede this instruction, then the Load Word presented the most significant 32 bits, and the data presented by this Load Register instruction is the least significant 32 bits of the double-precision argument.

This 64-bit effective second operand is normalized, if necessary, and then R*-rounded to single-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no overflow or underflow occurs, the rounded result replaces the contents of the single-precision floating-point register specified by A.

For the RR Control format, the K bit causes any normalization or rounding to be avoided. The second operand is copied directly into the floating-point register specified by A with no modification.

LE,LEI: $A \leftarrow B_E$

LEX: $A \leftarrow B_E$

then RLC10:15 $\leftarrow$ PAGE ADDRESS

Resulting Flags (if E bit is not set)

C	V	G	L	
0	0	0	1	Fraction was normalized and less than zero
0	0	1	0	Fraction was normalized and greater than zero
0	1	X	X	Fraction was not normalized

Final Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	0	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a single-precision floating-point register. If the E bit is not set, the floating-point operation is performed independent of any other processor activity. In this case, the V flag resulting from the instruction may be tested to determine whether or not the fraction was normalized. If the V flag is zero, the fraction was already normalized, and all other flags are correct, provided that there is no possibility of exponent overflow in the final result due to R*-rounding.

If the V flag is set or if there is a possibility of exponent overflow, then the flags corresponding to the final result must be collected by a read condition code microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is completed. Valid flags corresponding to the final result are produced and gated to the condition code when the operation is completed.

Because the single-precision floating-point registers are actually implemented as double-precision registers, an unnormalized load (LE with the K bit set) must be performed following power restore, to initialize the least significant 32 bits of each single-precision register. Failure to initialize these registers causes undefined data to participate in all operations using any initialized single-precision register, with unpredictable results.

Execution Times

LE, LEI:	460+130n	where n = normalize cycles
LEX (transfer taken):	460+130n	
LEX (no transfer):	605+130n	

4.6.5.3 Read Register Single Precision

RRE S,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	S	1 1 1 1 1	0 0 0 1	B	0	E	MC

RREX S,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	S	1 1 1 1 1	0 0 0 1	B	C	PAGE ADDRESS

The contents of the single-precision floating-point register specified by B are copied to the register specified by S.

RRE: S←(B)
RREX: S←(B)
then RLC10:15←PAGE ADDRESS

Resulting Flags

Not meaningful

Programming Notes

Floating-point register selection is not affected by the least significant B address bit. If an odd numbered register is specified, the next lower even numbered register is selected instead.

The S field may specify any register other than a floating-point register.

Execution Times

RRE:		410
RREX	(transfer taken):	410
RREX	(no transfer):	555

4.6.5.4 Compare Register Single Precision

CER A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	1 1 1 1 1	A	0 0 1 1	B	0	E	MC

CERX A,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	1 1 1 1 1	A	0 0 1 1	B	C	PAGE ADDRESS

The first operand is compared to the second operand. The comparison is algebraic, taking into account the sign, exponent, and fraction. The result is indicated by the resulting flags.

CER: $(A):\leftarrow B_E$

CERX: $(A):\leftarrow B_E$

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	First operand equal to second operand
1	0	0	1	First operand less than second operand
0	0	1	0	First operand greater than second operand

Execution Times

CER:	430
CERX (transfer taken):	430
CERX (no transfer):	575

4.6.5.5 Add Register Single Precision

AER A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	0 1 0 0	B	0	E	MC

AERX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	0 1 0 0	B	C	PAGE ADDRESS

The two operands are added in the following manner. The first operand is compared to the second operand. The comparison is in magnitude only, ignoring the signs of the two operands. The fraction field of the smaller of the two is shifted right hexadecimally (four bits at a time) the number of times indicated by the difference of the exponents of the two operands. This is called equalization. Note that hexadecimal digits shifted out of the low order end of the 24-bit fraction field are shifted through guard digits, which preserve the accuracy of the number to the limits of the precision of the input operand. The effect of this equalization process is to unnormalize the smaller operand so that the radix points of the two arguments are aligned. If the exponent difference exceeds seven, the smaller operand loses significance and a value of zero is substituted.

The equalized fraction with its guard digits and the fraction of the other operand with trailing zeros are then added, taking into account the signs and order of the two operands. The result fraction has an exponent equal to that of the larger operand. If the addition of fractions produces a carry, the result fraction with guards is shifted right one hexadecimal position and the result exponent is incremented by one. If no carry was produced, the result fraction with guards is normalized if necessary. The result exponent is decremented by one for each normalization cycle required.

When the result fraction has been normalized, the contents of the guard digits participate in an R*-rounding of the result to single-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the single-precision floating-point register specified by A.

AER: $A \leftarrow (A)+B_E$

AERX: $A \leftarrow (A)+B_E$

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a single-precision floating-point register.

If the second operand is larger than the first operand, this instruction requi. s an additional 100 nsec to execute. Therefore, if data is known, this penalty can be avoided by ensuring that the second operand is the smaller of two unequal operands.

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

AER: 700 + 100 (e+n) e=equalize cycles, n=normalize cycles } worst case total of e+n=6

AERX (transfer taken): Same as AER
AERX (no transfer): Same as AER, plus 145

If the B operand is greater than the A operand, add 100
If R*-rounding required, add 100

4.6.5.6 Subtract Register Single Precision

SER A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	1 1 1 1 1	A	0 1 0 1	B	0	E	MC

SERX A,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	1 1 1 1 1	A	0 1 0 1	B	C	PAGE ADDRESS

The two operands are subtracted in the following manner. The first operand is compared to the second operand. The comparison is in magnitude only, ignoring the signs of the two operands. The fraction field of the smaller of the two is shifted right hexadecimally (four bits at a time) the number of times indicated by the difference of the exponents of the two operands. This is called equalization. Note that hexadecimal digits shifted out of the low order end of the 24-bit fraction field are shifted through guard digits, which preserve the accuracy of the number to the limits of the precision of the input operand. The effect of this equalization process is to unnormalize the smaller operand so that the radix points of the two arguments are aligned. If the exponent difference exceeds seven, the smaller operand loses significance and a value of zero is substituted.

The equalized fraction with its guard digits and the fraction of the other operand with trailing zeros are then subtracted, taking into account the signs and order of the two operands. The result fraction has an exponent equal to that of the larger operand. If the addition of fractions produces a carry, the result fraction with guards is shifted right one hexadecimal position and the result exponent is incremented by one. If no carry was produced, the result fraction with guards is normalized if necessary. The result exponent is decremented by one for each normalization cycle required.

When the result fraction has been normalized, the contents of the guard digits participate in an R*-rounding of the result to single-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the single-precision floating-point register specified by A.

SER: A ← (A) - B_E

SERX: A ← (A) - B_E

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a single-precision floating-point register.

If the second operand is larger than the first operand, this instruction requires an additional 100 nsec to execute. Therefore, if data is known, this penalty can be avoided by ensuring that the second operand is the smaller of two unequal operands.

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

SER: 700+100 (e+n) e = equalize cycles
n = normalize cycles
} worst case total of e+n=6

SERX (transfer taken): Same as SER
SERX (no transfer): Same as SER, plus 145

If the B operand is greater than the A operand, add 100
If R*-rounding is required, add 100

4.6.5.7 Multiply Register Single Precision

MER A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	1 1 1 1 1	A	0 1 1 0	B	0	E	MC

MERX A,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	1 1 1 1 1	A	0 1 1 0	B	C	PAGE ADDRESS

The exponents of the first and second operands are added and the result set aside as the result exponent. The result sign is determined by the rules of algebra. The fractions of the two operands are then multiplied. If the product is zero, the entire result (sign and exponent included) is set to zero. If the product is nonzero, the fraction and guard digits resulting from the multiplication are normalized or adjusted as necessary. The sign, exponent, and result fraction are then combined.

The contents of the guard digits participate in an R*-rounding of the result to single-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the single-precision floating-point register specified by A.

MER: $A \leftarrow (A)*B_E$

MERX: $A \leftarrow (A)*B_E$

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a single-precision floating-point register.

The MER algorithm scans the second operand, searching for bit combinations which allow a reduction in the number of multiplication steps. If one operand is known to have strings of four or more contiguous one bits or contiguous zero bits, then for fastest multiplication, that operand should be used as the second operand: i.e., (π)*(2.) requires less time than (2.) * (π).

If the E bit is not set, the floating-point operation is performed autonomously, independent of other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

MER:	1425/1793/2160 BEST/AVG/WORST
MERX (transfer taken):	Same as MER
MERX (no transfer):	Same as MER, plus 145

If R*-rounding is required, add 100

4.6.5.8 Divide Register Single Precision

DER A,B,I,E,MC [RR CONTROL]

0 3	5	6 11	11 16	16 20	20 25	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	0 1 1 1	B	0	E	MC

DERX A,B,ADRS,I,C [RR TRANSFER]

0 3	5	6 11	11 16	16 20	20 25	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	0 1 1 1	B	C	PAGE ADDRESS

The exponent of the second operand is subtracted from the exponent of the first operand and the result is set aside as the exponent of the final result. The result sign is determined by the rules of algebra. The first operand (dividend) is divided by the second operand (divisor). If the quotient is zero, the entire final result (sign and exponent included) is set to zero. If the quotient is nonzero, the quotient and guard digits resulting from the division are normalized or adjusted as necessary. The sign, exponent, and result fraction are then combined.

The contents of the guard digits participate in an R*-rounding of the result to single-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the single-precision floating-point register specified by A.

DER: $A \leftarrow (A)/B_E$

DERX: $A \leftarrow (A)/B_E$

then PLC10:15 $\leftarrow$ PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero
1	1	0	0	Divisor is zero

Programming Notes

The register specified by A must be a single-precision floating-point register.

In the event of attempted division by zero, the result destination register is unchanged. The operation is aborted, and the flags returned are set to 1100 .

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and ted to the condition code.

Execution Times

DER:	3395
DERX (transfer taken):	3395
DERX (no transfer):	3540

If R*-rounding is required, add 100

4.6.5.9 Load Word

LW A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	1 0 0 0	B	0	E	MC

LWX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	1 0 0 0	B	C	PAGE ADDRESS

LWI A,DATA,I [RR IMMEDIATE]

0	3	5	6	11	16	20 31
1 1 0	1 0	I	1 1 1 1 1	A	1 0 0 0	DATA

This microinstruction is required when the second operand for a double-precision function is not resident in one of the DFU's internal registers; that is, the second operand is contained in microregisters, in main memory, or in control store.

This instruction presents the most significant 32 bits of the desired argument to the FPP. This fullword is retained in a holding register within the FPP. A subsequent floating-point microinstruction presents the least significant 32 bits of the second operand to the FPP via the B bus and the operation is performed.

LW,LWI: FPP ← B_E

LWX: FPP ← B_E

then (RLC10:15) ← PAGE ADDRESS

Resulting Flags

Unchanged

Programming Note

The A field is not used, and should be null selected.

Execution Times

LW,LWX,LWI: 260

4.6.5.10 Load Register Double Precision

LD A,B,I,E,MC,K [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	1 0 1 0	B	K	E	MC

LDX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	1 0 1 0	B	C	PAGE ADDRESS

LDI A,DATA[,I] [RR IMMEDIATE]

0	3	5	6	11	16	20 31
1 1 0	1 0	I	1 1 1 1 1	A	1 0 1 0	DATA

This instruction loads the double-precision floating-point register specified by A in the following manner: if B specifies one of the double-precision floating-point registers, then that register contains the second operand.

Otherwise, if a Load Word instruction did not precede this Load Register instruction, the second operand presented by the Load Register instruction is the most significant 32 bits of a double-precision number. The least significant 32 bits of this number are forced to zero. If a Load Word instruction did precede this instruction, then the Load Word presented the most significant 32 bits, and the data presented by this Load Register instruction is the least significant 32 bits of the double-precision argument.

This 64-bit effective second operand is normalized, if necessary. If exponent underflow occurs in the final result, the current state of the PSW must be interrogated. If no underflow occurs, the result replaces the contents of the double-precision floating-point register specified by A.

For the RR Control format, the K bit causes any normalization to be avoided. The second operand is copied directly into the floating-point register specified by A with no modification.

LD,LDI: $A \leftarrow B_E$

LDX: $A \leftarrow B_E$

then $RLC10{:}15 \leftarrow$ PAGE ADDRESS

Resulting Flags (if E bit is not set)

C	V	G	L	
0	0	0	1	Fraction was normalized and less than zero
0	0	1	0	Fraction was normalized and greater than zero
0	1	X	X	Fraction was not normalized

Final Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	0	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a double-precision floating-point register. If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the V flag resulting from the instruction may be tested to determine whether the fraction was normalized or not. If the V flag is zero, the fraction was already normalized, and all other flags are correct, provided that there is no possibility of exponent overflow in the final result due to R*-rounding.

If the V flag is set, then the flags corresponding to the final result must be collected by a read condition code microinstruction if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete. Valid flags corresponding to the final result are produced and gated to the condition code when the operation is complete.

Execution Times

LD,LDI:	560+130n	where n = normalize cycles
LDX (transfer taken):	560+130n	
LDX (no transfer):	705+130n	

4.6.5.11 Read Register Double Precision

RRD S,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	S	1 1 1 1 1	1 0 0 1	B	0	E	MC

RRDX S,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	S	1 1 1 1 1	1 0 0 1	B	C	PAGE ADDRESS

The contents of the double-precision floating-point register half specified by B are copied into the register specified by S. The flags generated by this instruction equal the result flags of the last floating-point operation.

RRD: (S)←(B)
RRDX: (S)←(B)
then (RLC10:15)←PAGE ADDRESS

Resulting Flags

Not meaningful

Programming Notes

If B specifies an even-numbered register half, the most significant half of the floating-point register is fetched. Otherwise, the least significant half is fetched.

Execution Times

RRD:	410
RRDX: (transfer taken):	410
RRDX: (no transfer):	555

4.6.5.12 Compare Register Double Precision

CDR A,B,I,E,MC [RR CONTROL]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26	27-31
1 1 0	0 1	I	1 1 1 1 1	A	1 0 1 1	B	0	E	MC

CDRX A,B,CDRS,I,C [RR TRANSFER]

0-2	3-4	5	6-10	11-15	16-19	20-24	25	26-31
1 1 0	0 0	I	1 1 1 1 1	A	1 0 1 1	B	C	PAGE ADDRESS

The first operand is compared to the second operand. The comparison is algebraic, taking into account the sign, exponent, and fraction. The result is indicated by the resulting flags.

CDR: (A):$\longleftarrow B_E$

CDRX: (A):$\longleftarrow B_E$

then RLC10:15 $\longleftarrow$ PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	First operand equal to second operand
1	0	0	1	First operand less than second operand
0	0	1	0	First operand greater than second operand

Execution Times

CDR: 455
CDRX (transfer taken): 455
CDRX (no transfer): 600

4.6.5.13 Add Register Double Precision

ADR A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27–31
1 1 0	0 1	I	1 1 1 1 1	A	1 1 0 0	B	0	E	MC

ADRX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26–31
1 1 0	0 0	I	1 1 1 1 1	A	1 1 0 0	B	C	PAGE ADDRESS

The two operands are added in the following manner. The first operand is compared to the second operand. The comparison is in magnitude only, ignoring the signs of the two operands. The fraction field of the smaller operand is shifted right hexadecimally (four bits at a time) the number of times indicated by the difference of the exponents of the two operands. This is called equalization. Note that hexadecimal digits shifted out of the low order end of the 56-bit fraction field are shifted through guard digits which preserve the accuracy of the number to the limits of the precision of the input operand. The effect of this equalization process is to unnormalize the smaller operand so that the radix points of the two arguments are aligned. If the exponent difference exceeds 14, the smaller operand loses significance and a value of zero is substituted.

The equalized fraction with its guard digits and the fraction of the other operand with trailing zeros are then added, taking into account the signs and order of the two operands. The result fraction has an exponent equal to that of the larger operand. If the addition of fractions produces a carry, the result fraction with guards is shifted right one hexadecimal position and the result exponent is incremented by one. If no carry was produced, the result fraction with guards is normalized, if necessary. The result exponent is decremented by one for each normalization cycle required.

When the result fraction has been normalized, the contents of the guard digits participate in an R*-rounding of the result to double-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the double-precision floating-point register specified by A.

ADR: A ← (A)+B_E

ADRX: A ← (A)+B_E

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a double-precision floating-point register.

If the second operand is larger than the first operand, this instruction requires an additional 100 nsec to execute. Therefore, if data is known, this penalty can be avoided by ensuring that the second operand is the smaller of two unequal operands.

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, then the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

ADR: 750+100 (e+n) e=equalize cycles, n=normalize cycles } worst case total of e+n=13

ADRX (transfer taken): Same as ADR
ADRX (no transfer): Same as ADR, plus 145

If the B operand is greater than the A operand, add 100
If R*-rounding is required, add 100

4.6.5.14 Subtract Register Double Precision

SDR A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	1 1 0 1	B	0	E	MC

SDRX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	1 1 0 1	B	C	PAGE ADDRESS

The two operands are subtracted in the following manner. The first operand is compared to the second operand. The comparison is in magnitude only, ignoring the signs of the two operands. The fraction field of the smaller operand is shifted right hexadecimally (4 bits at a time) the number of times indicated by the difference of the exponents of the two operands. This is called equalization. Note that hexadecimal digits shifted out of the low order end of the 56-bit fraction field are shifted through guard digits which preserve the accuracy of the number to the limits of the precision of the input operand. The effect of this equalization process is to unnormalize the smaller operand so that the radix points of the two arguments are aligned. If the exponent difference exceeds 14, the smaller operand loses significance and a value of zero is substituted.

The equalized fraction with its guard digits and the fraction of the other operand with trailing zeros are then subtracted, taking into account the signs and order of the two operands. The result fraction has an exponent equal to that of the larger operand. If the addition of fractions produces a carry, the result fraction with guards is shifted right one hexadecimal position and the result exponent is incremented by one. If no carry was produced, the result fraction with guards is normalized, if necessary. The result exponent is decremented by one for each normalization cycle required.

When the result fraction has been normalized, the contents of the guard digits participate in an R*-rounding of the result to double-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the double-precision floating-point register specified by A.

SDR: A ← (A) - B_E

SDRX: A ← (A) - B_E

then RLC10:15 ← PAGE ADDRESS

Resulting Flags (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a double-precision floating-point register.

If the second operand is larger than the first operand, this instruction requires an additional 100 nsec to execute. Therefore, if data is known, this penalty can be avoided by ensuring that the second operand is the smaller of two unequal operands.

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction if they are to be known.

If the E bit is set in this instruction, then the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

SDR: 750+100 (e+n) e = equalize cycles, n = normalize cycles } worst case total of e+n=13

SDRX (transfer taken): Same as SDR
SDRX (no transfer): Same as SDR, plus 145

If the B operand is greater than the A operand, add 100
If R*-rounding is required, add 100

4.6.5.15 Multiply Register Double Precision

MDR A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
1 1 0	0 1	I	1 1 1 1 1	A	1 1 1 0	B	0	E	MC

MDRX A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
1 1 0	0 0	I	1 1 1 1 1	A	1 1 1 0	B	C	PAGE ADDRESS

The exponents of the first and second operands are added and then set aside as the exponent of the final result. The result sign is determined by ıe rules cf algebra. The fractions of the two operands are then multiplied. If the product is zero, the entire result (sign and exponent included) is set to zero. If the product is nonzero, the fraction and guard digits resulting from the multiplication are normalized or adjusted as necessary. The sign, exponent, and result fraction are then combined.

The contents of the guard digits participate in an R*-rounding of the result to double-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the double-precision floating-point register specified by A.

MDR: $A \leftarrow (A) * B_E$

MDRX: $A \leftarrow (A) * B_E$

then RLC10:15 ← PAGE ADDRESS

<u>Resulting Flags</u> (after RCC, or if the E bit is set)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero

Programming Notes

The register specified by A must be a double-precison floating-point register.

The MDR algorithm scans the second operand, searching for bit combinations which allow a reduction in the number of multiplication steps. If one operand is known to have strings of four or more contiguous one bits or contiguous zero bits, then for fastest multiplication, that operand should be used as the second operand; i.e., (π)*(2.) is faster than (2.)*(π).

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction, if they are to be known.

If the E bit is set in this instruction, the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

MDR:	2410/3020/3630 BEST/AVG/WORST
MDRX (transfer taken):	Same as MDR
MDRX (no transfer):	Same as MDR, plus 145

If R*-rounding is required, add 100

4.6.5.16 Divide Register Double Precision

DDR A,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
1 1 0	0 1	I	1 1 1 1 1	A	1 1 1 1	B	0	E	MC

DDRX A,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
1 1 0	0 0	I	1 1 1 1 1	A	1 1 1 1	B	C	PAGE ADDRESS

The exponents of the first and second operands are subtracted and then set aside as the result exponent. The result sign is determined by the rules of algebra. The first operand (dividend) is divided by the second operand (divisor). If the quotient is zero, the entire final result (sign and exponent included) is set to zero. If the quotient is nonzero, then the quotient and guard digits resulting from the division are normalized or adjusted as necessary. The sign, exponent, and result fraction are then combined.

The contents of the guard digits participate in an R*-rounding of the result to double-precision accuracy. If exponent overflow or underflow occurs in the final result, the current state of the PSW must be interrogated. If no exponent overflow or underflow occurs, the rounded result replaces the contents of the double-precision floating-point register specified by A.

DDR: $A \leftarrow (A)/B_E$

DDRX: $A \leftarrow (A)/B_E$

then RLC10:15 ← PAGE ADDRESS

<u>Resulting Flags</u> (after RCC)

C	V	G	L	
0	0	0	0	Result is zero
0	0	0	1	Result is less than zero
0	0	1	0	Result is greater than zero
0	1	0	0	Exponent underflow
0	1	0	1	Exponent overflow, result is less than zero
0	1	1	0	Exponent overflow, result is greater than zero
1	1	0	0	Divisor is zero

Programming Notes

The register specified by A must be a double-precision floating-point register.

In the event of attempted division by zero, the result destination register is unchanged. The operation is aborted, and the flags returned are set to 1100_2.

If the E bit is not set, the floating-point operation is performed autonomously, independent of any other processor activity. In this case, the flags corresponding to the final result must be collected by an RCC microinstruction, if they are to be known.

If the E bit is set in this instruction, then the microprogram is not allowed to proceed until the floating-point operation is complete, and valid flags corresponding to the final result have been produced and gated to the condition code.

Execution Times

DDR:	6580
DDRX (transfer taken):	6580
DDRX (no transfer):	6725

If R*-rounding is required, add 100

4.7 BYTE HANDLING INSTRUCTIONS

These instructions use the I/O module to perform byte manipulations on the least significant 16 bits of A, B, and S bus data. The instructions described in this section are:

4.7.1	LB	Load Byte
	LBR	Load Byte Register
4.7.2	STB	Store Byte
	STBR	Store Byte Register
4.7.3	EXB	Exchange Byte

4.7.1 Load Byte

LB S,A,B,I,E,M [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	0 1 0 1	B	1	E	MC

LBR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 0 1	B	1	E	MC

Bits 24:31 of the second operand replace bits 24:31 of the register specified by S. The most significant 24 bits of S are set to zero.

LB,LBR : $S0{:}23 \leftarrow 0$

$S24{:}31 \leftarrow B_E\ (24{:}31)$

<u>Resulting Flags</u>

C	V	G	L
0	0	0	0

<u>Execution Times</u>

LB,LBR: 260

4.7.2 Store Byte

STB S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26 27	31
0 1 0	0 1	I	S	A	1 1 0 0	B	1	E	MC

STBR S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26 27	31
0 1 0	0 1	I	S	A	0 1 0 0	B	1	E	MC

Bits 24:31 of A are copied to bits 24:31 of the register specified by S. Bits 16:23 of S are set equal to bits 16:23 of B. Bits 00:15 of S are set to zero.

STB,STBR: S0:15 ← 0

S16:23 ← B_E(16:23)

S24:31 ← (A24:31)

Resulting Flags

C	V	G	L
0	0	0	0

Execution Times

STB,STBR: 260

4.7.3 Exchange Byte

EXB S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	1 0 1 0	B	1	E	MC

The two low order bytes of the second operand are exchanged and loaded into the register specified by S.

EXB: $S0{:}15 \leftarrow 0$

$S16{:}23 \leftarrow B_E(24{:}31)$

$S24{:}31 \leftarrow B_E(16{:}23)$

Resulting Flags

C	V	G	L
0	0	0	0

Execution Times

EXB: 260

4.8 CONTROL INSTRUCTIONS

These instructions allow testing and clearing the Machine Control Register, control over the console interrupt and the Consolette WAIT lamp and FAULT lamp, and the initialize relay. The instructions covered in this section are:

4.8.1	SMCR	Sense Machine Control Register
	SMCRX	Sense Machine Control Register and Transfer
4.8.2	CMCR	Clear Machine Control Register
4.8.3	LWFF	Load the Wait Flip-Flop
4.8.4	PDW	Power Down
4.8.5	BDC	Branch and Disable Console

4.8.1 Sense Machine Control Register

SMCR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 1 1	B	0	E	MC

SMCRX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 1 1 1	B	C	PAGE ADDRESS

The contents of the Machine Control Register replace the contents of the register specified by S. Bits 12:15 of the MCR become available on the CC bus and are copied into the microflag register.

SMCR: S←(MCR)
SMCRX: S←(MCR)
then (RLC10:15)←PAGE ADDRESS

Resulting Flags

C	V	G	L	
1				Module timeout
	1			Memory voltage failure
		1		Hardware CRC assist installed
			1	Early power failure

Programming Notes

The B field is not used and should be null selected.

The meanings of the MCR bits are summarized below.

MNEMONIC	BIT		MEANING
FPP	04	-	Set if FPP module is installed
-	05	-	Undefined
INIT	06	-	Set while initialize switch on the consolette is depressed
SNGL	07	-	Set while single-step switch on the consolette is on
-	08	-	Undefined
-	09	-	Undefined
CATN	10	-	Set when EXE/HLT switch on the consolette was depressed, or Instruction Read/Decode cycle completed in single-step mode
MTO	11	-	Set if optional module timeout feature is installed
STF	12	-	Set following a module timeout, i.e., no response after 35 microseconds
NVM	13	-	Set when NVMO is active from memory, indicating voltage failure
CRC	14	-	Set if hardware CRC assist option is installed
EPF	15	-	Set by early power failure detect

Execution Times

SMCR,SMCRX: 260

4.8.2 Clear Machine Control Register

CMCR S,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 1 1	B	1	E	MC

The bits in the MCR that correspond to ones in the second operand are set to zeros. The S field is not used and should be null selected.

CMCR: $MCR \leftarrow (MCR)\ AND\ \overline{B_E}$

Resulting Flags

C	V	G	L
0	0	0	0

Programming Notes

The MCR bits that are straps cannot be modified.

The first CMCR microinstruction issued following the release of system clear causes the consolette FAULT lamp to be turned off.

Execution Times

CMCR: 260

4.8.3 Load the Wait Flip-Flop

LWFF S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 1 0	B	1	E	MC

Bit 16 of the second operand is copied into the wait flip-flop. A one sets the flip-flop and turns on the console WAIT lamp. A zero resets the flip-flop and turns off the console WAIT lamp.

LWFF: WAIT $\leftarrow B_E(16)$

Resulting Flags

C	V	G	L
0	0	0	0

Execution Time

LWFF: 260

4.8.4 Power Down

PCW S,B,I,E,MC [RR CONTROL]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26	27–31
0 1 0	0 1	I	S	1 1 1 1 1	1 1 1 1	B	0	E	MC

This microinstruction may be issued by the emulator in anticipation of automatic shutdown of the processor, following the housekeeping required by the PPF interrupt. POW performs no operation, and is provided for documentation purposes only. The S, B and MC fields are not interpreted. The resulting condition code and execution time are meaningless. POW must be followed by a BALD *(NULL) microinstruction.

When the system clear relay is reenabled, microcode execution resumes at control store memory address '001'.

4.8.5 Branch and Disable Console

BDC ADRS(LINK) E,MC [ADDRESS LINK]

0	3	5	6	11	14	26	27 31
0 0 0	1 0	1	LINK	1 1 0	ADDRESS	E	MC

BDC (B)(LINK) E,MC [REGISTER LINK]

0	3	5	6	11	14	20	25	26	27 31
0 0 0	0 0	1	LINK	1 1 0		B		E	MC

Interrupts from the console (CATN or SNGL) are ignored for the interval of this instruction so that interrupts of lower priority can be detected. No branch is actually taken, so MC field functions can occur.

BDC LINK←(RLC4:15)+1

<u>Execution Time</u>

BDC: 260

CHAPTER 5
INPUT/OUTPUT SYSTEM

5.1 INTRODUCTION

This chapter discusses the Input/Output (I/O) system. There are several methods of communication between the processor and peripheral devices or other system elements. These methods vary in speed, sophistication, and the amount of attention required by the processor.

5.2 MULTIPLEXOR BUS

The multiplexor bus is a byte or halfword oriented I/O system which communicates with up to 1,023 peripheral devices. The multiplexor bus consists of 33 lines - 16 bidirectional data lines, 10 control lines, 6 test lines and 1 initialize line. The lines in the multiplexor bus are:

Data Lines	D00:15	(Processor ⇄ Device)	16 lines
Control Lines	SR	(→)	1 line
	DR	(→)	1 line
	CMD	(→)	1 line
	DA	(→)	1 line
	ADRS	(→)	1 line
	ACK0	(→)	1 line
	ACK1	(→)	1 line
	ACK2	(→)	1 line
	ACK3	(→)	1 line
	CL07	(→)	1 line
Test Lines	ATN0	(←)	1 line
	ATN1	(←)	1 line
	ATN2	(←)	1 line
	ATN3	(←)	1 line
	SYN	(←)	1 line
	HW	(←)	1 line
Initialize	SCLR	(→)	1 line

5.2.1 Data Lines

The 16 bidirectional data lines are used to transfer one 8-bit byte, one 10-bit device address, or one 16-bit halfword between the processor and the device. In actuality, 16 bits are always transferred, and the device or the processor accepts as much of the data as is required for the particular operation.

5.2.2 Control Lines

ADRS — Address. The processor presents a 10-bit device address on data lines D06:15. The device controller that recognizes its address becomes the 'on-line' device and responds with a Synchronize (SYN). Once a device has been addressed, it remains so, until a different device is addressed or a system initialize occurs. If the device is halfword oriented, the Halfword test line (HW) is also active, and may be tested by the THWX microinstruction.

DA — Data Available. The processor presents data to be transferred to the addressed device on data lines D00:15. The addressed device controller accepts the low order byte or the entire halfword and responds with a SYN. If SYN does not occur before a fixed time-out, the V flag is set in the microflags and on the CC bus. All other flags are zero in this case.

DR — Data Request. The addressed device controller presents data on data lines D08:15 or D00:15, followed by a SYN. If SYN does not occur before a fixed time-out, the V flag is set in the microflags and on the CC bus. All other flags are zero in this case.

SR — Status Request. The addressed device controller presents status information on data lines D08:15, followed by a SYN. If SYN does not occur before a fixed time-out, the V flag is set in the microflags and on the CC bus. All other flags are zero in this case.

CMD — Output Command. The processor presents a command byte on data lines D08:15 for the addressed device. The addressed device controller accepts the command byte and responds with a SYN. If SYN does not occur before a fixed time-out, the V flag is set in the microflags and on the CC bus. All other flags are zero in this case.

ACK0
ACK1
ACK2
ACK3

Acknowledge. The microprogram generates an Acknowledge signal on the appropriate line in response to an active Attention (ATN) test line. The device controller nearest the processor on the particular acknowledge line that is activating the corresponding ATN line presents its address on data lines D06:15, followed by SYN. That device controller then removes ATN. If SYN does not occur before a fixed time-out, the V flag is set in the microflags and on the CC bus. All other flags are zero in this case.

CL07 — This control line is activated by the initialize key, the PWR STANDBY switch, or when the power fail detector determines that the processor's primary power is failing. The line remains active as long as the PPF interrupt line is active.

5.2.3 Test Lines

ATN0
ATN1
ATN2
ATN3

Attention. When so enabled, any device on one of the attention lines trying to interrupt the processor activates the ATNx line and holds it active until Acknowledge is received from the processor.

HW — Halfword. Any halfword-oriented device activates the halfword test line when it becomes addressed and holds the line active for as long as the device is addressed. The HW line being active suppresses the byte steering done in the I/O module in DA or DR operations.

SYN — Synchronize. This signal is generated by the device controller to inform the processor that it is responding to the active control line.

5.2.4 Initialize line

SCLR — System Clear. This is a metallic contact to ground that occurs during power fail or initialize.

5.3 INPUT/OUTPUT INSTRUCTIONS

Communication over the multiplexor bus is on a request/response basis where each operation started by the processor must receive an SYN response to terminate the operation. If no SYN response is received within approximately 35 microseconds, A False Sync (FSYN) is automatically generated to terminate the operation.

Input/output microinstructions generate one, two, or three multiplexor bus operations. Each operation lasts until SYN is received from the device, meaning that the execution time on I/O instructions is solely device dependent.

NOTE

> All I/O instruction execution times are given using the following assumptions:
>
> 1. Average circuit delays, not maximum or minimum
> 2. SYN delay of 100 nanoseconds
> 3. No bus buffer delay in the system

The instructions described in this section are:

5.3.1	AK	Acknowledge Interrupt
	AKX	Acknowledge Interrupt and Transfer
5.3.2	SSA	Address and Sense Status
	SSAX	Address and Sense Status and Transfer
	SSRA	Address and Sense Status Register
5.3.3	SS	Sense Status
	SSX	Sense Status and Transfer
	SSR	Sense Status Register
5.3.4	OCA	Address and Output Command
	OCAX	Address and Output Command and Transfer
	OCAI	Address and Output Command Immediate
	OCRA	Address and Output Command Register

5.3.5	OC	Output Command
	OCX	Output Command and Transfer
	OCI	Output Command Immediate
	OCR	Output Command Register
5.3.6	RDA	Address and Read Data
	RDAX	Address and Read Data Transfer
	RDRA	Address and Read Data Register
5.3.7	RD	Read Data
	RDX	Read Data and Transfer
	RDR	Read Data Register
5.3.8	WDA	Address and Write Data
	WDAX	Address and Write Data and Transfer
	WDAI	Address and Write Data Immediate
	WDRA	Address and Write Data Register
5.3.9	WD	Write Data
	WDX	Write Data and Transfer
	WDI	Write Data Immediate
	WDR	Write Data Register
5.3.10	RHA	Address and Read Halfword
	RHAX	Address and Read Halfword and Transfer
5.3.11	RH	Read Halfword
	RHX	Read Halfword and Transfer
5.3.12	WHA	Address and Write Halfword
	WHAX	Address and Write Halfword and Transfer
5.3.13	WH	Write Halfword
	WHX	Write Halfword and Transfer
5.3.14	THWX	Test Halfword Line and Transfer

5.3.1 Acknowledge Interrupt

AK S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 1 0	B	0	E	MC

AKX S,E,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 1 1 0	B	C	PAGE ADDRESS

Bits 30 and 31 of the effective second operand select the desired ACK control line. The device number of the interrupting device replaces the contents of the register specified by S. The interrupt condition in the controller is cleared.

AK: S0:21←0

S22:31←DEVICE NUMBER

AKX: S0:21←0

S22:31←DEVICE NUMBER

then RIC10:15←PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out (no device response)

Programming Note

Each ACK control line passes through the interrupt circuits on all of its assigned controllers in a daisy chain fashion. Execution time is increased by 100 nanoseconds for each controller between the processor and the interrupting controller.

Execution Time

AK,AKX: 720 plus SYN response time

5.3.2 Address and Sense Status

SSA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 1 0	B	0	E	MC

SSAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	A	1 0 1 0	B	C	PAGE ADDRESS

SSRA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 1 0	B	1	E	MC

The register specified by A contains the device address. The device is addressed and its 8-bit status byte replaces the contents of the register specified by S. The right-most four bits of the status byte are available on the CC bus and are copied into the microflag register.

SSA,SSRA: CC Bus←DEVICE STATUS (4:7)

S0:15←0

SSAX: Same as SSA
then RLC10:15←PAGE ADDRESS

Resulting Flags

C	V	G	L	
1				Device Busy (BSY)
	1			Examine Status (EX) or Time-out
		1		End of Medium (EOM)
			1	Device Unavailable (DU)

Execution Times

SSA,SSAX,SSRA: 1590 plus SYN response time

5.3.3 Sense Status

SS S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 1 0	B	0	E	MC

SSX S,E,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 0 1 0	B	C	PAGE ADDRESS

SSR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 1 0	B	1	E	MC

The Sense Status instructions are identical to the Address and Sense Status instructions except that the address cycle is avoided. Once addressed, a device controller remains addressed until a different device controller is addressed or a system clear occurs.

Execution Times

SS,SSX,SSR: 720 plus SYN response time

5.3.4 Address and Output Command

OCA S,A,B,I,E,MC [RR CONTROL]

0 - 2	3 - 4	5	6 - 10	11 - 15	16 - 19	20 - 24	25	26	27 - 31
0 1 0	0 1	I	S	A	1 0 1 1	B	0	E	MC

OCAX S,A,B,ADRS,I,C [RR TRANSFER]

0 - 2	3 - 4	5	6 - 10	11 - 15	16 - 19	20 - 24	25	26 - 31
0 1 0	0 0	I	S	A	1 0 1 1	B	C	PAGE ADDRESS

OCAI S,A,DATA,I [RR TRANSFER]

0 - 2	3 - 4	5	6 - 10	11 - 15	16 - 19	20 - 31
0 1 0	1 0	I	S	A	1 0 1 1	DATA

CCRA S,A,B,I,E,MC [RR CONTROL]

0 - 2	3 - 4	5	6 - 10	11 - 15	16 - 19	20 - 24	25	26	27 - 31
0 1 0	0 1	I	S	A	1 0 1 1	B	1	E	MC

The register specified by A contains the device address. The device is addressed and the 8-bit second operand command byte is sent to the device.

OCA,OCAI,OCRA: DEVICE ← B_E(24:31)

OCAX: Same as OCA
then (RLC10:15) ← PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out

Programming Note

The S field is not used and should be null selected.

Execution Times

OCA,OCAX,OCAI,OCRA: 1720 plus SYN response time

5.3.5 Output Command

OC S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 1 1	B	0	E	MC

OCX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 0 1 1	B	C	PAGE ADDRESS

OCI S,B,DATA,I [RR TRANSFER]

0	3	5	6	11	16	20 31
0 1 0	1 0	I	S	1 1 1 1 1	0 0 1 1	DATA

OCR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 1 1	B	1	E	MC

The Output Command instructions are identical to the Address and Output Command instructions except that the address cycle is avoided.

<u>Execution Times</u>

OC,OCX,OCI,OCR: 840 plus SYN response time

5.3.6 Address and Read Data

RDA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 0 0	B	0	E	MC

RDAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	A	1 0 0 0	B	C	PAGE ADDRESS

RDRA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 0 0	B	1	E	MC

The register specified by A contains the device address. The device is addressed and a single 8-bit data byte is transferred from the device to the register specified by S.

RDA,RDRA: S0:15 ← 0

S16:23 ← B_E(16:23)

S24:31 ← DEVICE DATA

RDAX: Same as RDA
then RIC10:15 ← PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out

Execution Times

RDA,RDAX,RDRA: 1590 plus SYN response time

5.3.7 Read Data

RD S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 0 0	B	0	E	MC

RDX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 0 0 0	B	C	PAGE ADDRESS

RDR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 0 0	B	1	E	MC

The Read Data instructions are identical to the Address and Read Data instructions except that the address cycle is avoided.

Execution Times

RD,RDX,RDR: 720 plus SYN response time

5.3.8 Address and Write Data

WDA S,A,B,I,E, MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 0 1	B	0	E	MC

WDAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	A	1 0 0 1	B	C	PAGE ADDRESS

WDAI S,A,DATA,I [RR TRANSFER]

0	3	5	6	11	16	20 31
0 1 0	1 0	I	S	A	1 0 0 1	DATA

WDRA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 0 0 1	B	1	E	MC

The register specified by A contains the device address. The device is addressed and a single 8-bit byte is transferred to the device.

WDA,WDAI,WDRA: DEVICE ← B_E(24:31)

WDAX: Same as WDA
then RIC10:15 ← PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out

Programming Note

The S field is not used and should be null selected.

Execution Times

WDA,WDAX,WDAI,WDRA: 1720 plus SYN response time

5.3.9 Write Data

WD S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 0 1	B	0	E	MC

WDX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 0 0 1	B	C	PAGE ADDRESS

WDI S,B,DATA,I [RR TRANSFER]

0	3	5	6	11	16	20 31
0 1 0	1 0	I	S	1 1 1 1 1	0 0 0 1	DATA

WDR S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 0 0 1	B	1	E	MC

The Write Data instructions are identical to the Address and Write Data instructions except that the address cycle is avoided.

Execution Times

WD,WDX,WDI,WDR: 840 plus SYN response time

5.3.10 Address and Read Halfword

RHA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 1 0 0	B	0	E	MC

RHAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	A	1 1 0 0	B	C	PAGE ADDRESS

The register specified by A contains the device address. The device is addressed and a 16-bit halfword is transferred from the device to the register specified by S. The Read Halfword instructions can be used with both byte and halfword oriented controllers. If the controller is byte oriented, the Halfword test line (HW) is inactive. The I/O module inputs two 8-bit bytes, one after the other. If the controller is halfword oriented, the HW test line is active, and the I/O module inputs one 16-bit halfword in parallel.

RHA: S0:15 ← 0

S16:23 ← First data byte } Byte oriented controller
S24:31 ← Second data byte }

S16:31 ← Halfword of data   Halfword oriented controller

RHAX: Same as RHA
then RLC10:15 ← PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out

Programming Note

The B field is not used and should be null selected.

Execution Times

RHA,RHAX:	2440	plus SYN response time (Byte oriented device)
	1590	plus SYN response time (Halfword oriented device)

5.3.11 Read Halfword

RH S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 0 0	B	0	E	MC

RHX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 1 0 0	B	C	PAGE ADDRESS

The Read Halfword instructions are identical to the Address and Read Halfword instructions except that the address cycle is avoided.

Execution Times

RH,RHX: 1570 plus SYN response time (Byte oriented device)
720 plus SYN response time (Halfword oriented device)

5.3.12 Address and Write Halfword

WHA S,A,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	A	1 1 0 1	B	0	E	MC

WHAX S,A,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	A	1 1 0 1	B	C	PAGE ADDRESS

The register specified by A contains the device address. The device is addressed and a 16-bit halfword is transferred from the processor to the device. The Write Halfword instructions can be used with either byte or halfword oriented controllers. If the controller is byte oriented, the Halfword test line (HW) is inactive. The I/O module outputs two 8-bit bytes, one after the other. If the controller is halfword oriented, the HW test line is active, and the I/O module outputs one 16-bit halfword.

WHA: DEVICE ← B_E(16:23) } Byte oriented controller
DEVICE ← B_E(24:31) }

DEVICE ← B_E(16:31) Halfword oriented controller

WHAX: Same as WHA
then RLC10:15 ← PAGE ADDRESS

Resulting Flags

C	V	G	L	
0	0	0	0	Normal execution
0	1	0	0	Instruction time-out

Programming Note

The S field is not used and should be null selected.

Execution Times

WHA,WHAX: 2520 plus SYN response time (Byte oriented device)
1720 plus SYN response time (Halfword oriented device)

5.3.13 Write Halfword

WH S,B,I,E,MC [RR CONTROL]

0	3	5	6	11	16	20	25	26	27 31
0 1 0	0 1	I	S	1 1 1 1 1	0 1 0 1	B	0	E	MC

WHX S,B,ADRS,I,C [RR TRANSFER]

0	3	5	6	11	16	20	25	26 31
0 1 0	0 0	I	S	1 1 1 1 1	0 1 0 1	B	C	PAGE ADDRESS

The Write Halfword instructions are identical to the Address and Write Halfword instructions except that the address cycle is avoided.

Execution Times

WH,WHX: 1640 plus SYN response time (Byte oriented device)
840 plus SYN response time (Halfword oriented device)

5.3.14 Test Halfword Line and Transfer

THWX S,B,ADRS,I,C [RR TRANSFER]

0–2	3–4	5	6–10	11–15	16–19	20–24	25	26–31
0 1 0	0 0	I	S	1 1 1 1 1	1 1 1 0	B	C	PAGE ADDRESS

This microinstruction is provided to allow the microprogram to test the state of the Halfword test line (HW). The HW test line is active for as long as any halfword oriented controller is addressed.

THWX RLC10:15←—PAGE ADDRESS if C=0 or HW=0

RLC4:15←—(RLC4:15)+1 if C=1 and HW=1

Resulting Flags

C	V	G	L
0	0	0	0

Programming Note

The S and B fields are unused and should be null selected.

Execution Times

THWX (no transfer): 405
(transfer): 260

CHAPTER 6
INTERRUPT SYSTEM

6.1 GENERAL INFORMATION

The hardware priority interrupt structure provides rapid response to internal and external events which require special program attention. When an interrupt occurs, the microprogram is steered to one of nine unique control store addresses, according to the type of interrupt. In order of decreasing priority, these addresses are '208', '207', '206', '205', '204', '203', '202', '201', and '200'.

Certain interrupts can be individually enabled or disabled by bits in the PSW. All interrupts, except those resulting from illegal instructions or from a memory read or write operation, can be collectively armed by the BALA or EXLA microinstructions, or collectively disarmed by the BALD or EXLD microinstructions. Illegal instruction and memory read/write interrupts cannot be disarmed by the microprogram. All interrupts are collectively armed at the completion of a microinstruction which specifies the decode option (IRD or D), so that interrupt service can occur before starting the next user instruction. Interrupts are then disarmed as the emulation of the decoded user instruction proceeds, until specifically armed by the microprogram.

When an interrupt occurs, the microinstruction at the corresponding interrupt trap location is executed. The RLC is not changed so that the microprogram could return to the interrupted program sequence if desired. The standard emulator uses this capability for faults occurring while in the console service routine.

The various possible interrupts, with pertinent enabling PSW bits and trap locations, are shown in Table 1-2. The following descriptions are oriented towards the emulator.

6.2 INTERNAL INTERRUPTS

Although the hardware provides only four unique internal interrupt trap locations, at the time of a hardware interrupt, sufficient information is provided that action appropriate to the emulated machine may be taken by the microprogram. Other internal interrupts are created by the emulator, and do not have dedicated trap locations.

6.2.1 Illegal Instruction Interrupt (208)

An illegal instruction interrupt is generated by the hardware in the following instances:

1. when an instruction not in the user-level instruction repertoire is attempted
2. when execution of a privileged instruction is attempted and PSW bit 23 is set
3. when execution of a floating-point instruction is attempted and PSW bit 13 is set

As a result of an instruction read, the main memory gates its read-out into the User's Instruction Register (UIR). When the decode (IRD or D) option is also specified, at the conclusion of the present microinstruction, the processor waits until the next user instruction is available in the UIR, at which time the privileged/illegal ROM is interrogated.

The privileged/illegal ROM is addressed by the operation code field of UIR (UIR bits 0:7). There is a 4-bit data entry in the privileged/illegal ROM for each of the 256 possible user op-codes.

If the user instruction is not a legal instruction for the current user, the hardware causes an illegal instruction interrupt to occur and the microinstruction at control store location '208' is executed. The user-level illegal instruction PSW swap is then emulated.

In some instances, additional tests are performed on the user instruction by the emulator. If the instruction proves to be illegal, the user-level illegal instruction PSW swap is executed, as though the user instruction were found illegal by the hardware.

6.2.2 Access/Data/Boundary/Floating-Point Interrupt (207)

When using the optional Floating-Point Processor (FPP), if a floating-point arithmetic fault condition occurs, a floating-point interrupt is queued. This interrupt remains queued until an FPP operation is performed which does not result in a fault condition being indicated. If the interrupt is queued when a microinstruction directed to the FPP is executed, and if that microinstruction specifies the IRD and E functions, the interrupt is taken and the microinstruction at location '207' is executed.

Note that if the fault condition is indicated by the RCC instruction, a subsequent

```
RCC NULL,NULL,IRD,E
```

results in an interrupt to location '207' each time it is performed, until the FPP flags are changed by some nonfaulting FPP operation. If the Memory Address Translator (MAT) is enabled by bit 21 of the PSW, violation of any of the relocation and protection conditions in the MAT controller causes the microinstruction at control store location '207' to be executed.

When PSW bit 18 is set, if fetching data from memory results in a noncorrectable data error, the microinstruction at control store location '207' is executed.

This is also the case if a memory access is attempted for a nonconfigured memory location (STM, controller absent on a read, or controller absent on a write). If the Shared Memory Controller (optional) is in a power-down or off-line state, the interrupt occurs for any access to that Shared Memory Bank.

If a fullword memory read or write operation is directed to a location not aligned to a fullword boundary, or if a halfword memory read or write operation is directed to a location not aligned to a halfword boundary, the microinstruction at control store location '207' is executed. This is also the case if a DR2IB, DR4IB, IR, or IRD operation is attempted, and CLOC contains an address not aligned to a halfword boundary. This interrupt cannot be inhibited.

If any of the above faults occur while fetching any halfword of an instruction, the interrupt is deferred until an attempt is made by the microprogram to decode the offending instruction; otherwise, the interrupt occurs immediately, and the memory operation is aborted. For an instruction fetch, if a subsequent buffer refill or cache and buffer refill occurs, and no error occurs as a result of refetching the instruction, the interrupt condition is reset. Up to 2 microinstructions may be fetched, although not executed, before the interrupt is reported.

A unique code identifying the type of fault, and the program address in MAR at the time the fault occurred are available to the microprogram by unloading RMDR following the interrupt. (Refer to Figure 6-1.)

0–7	8–31
FAULT CODE	PROGRAM ADDRESS

Figure 6-1 Contents of RMDR Following a Fault

The fault codes and their meanings are shown in Table 6-1. The microinstruction at the trap location must disarm interrupts. Once the fault information has been retrieved from RMDR (if it is to be known), the fault condition may be reset only by the RFAULT MC field option.

Following the fault interrupt, fault information is available in RMDR until a microinstruction specifying an MC function is executed. Once the fault information has been unloaded from RMDR, the fault condition must be reset by specifying the RFAULT MC function, on the same or a subsequent microinstruction. Attempts to access the instruction buffer or memory are unsuccessful until the fault is reset. Once RFAULT has been issued, the contents of RMDR are undefined until data is explicitly fetched from memory or the instruction buffer.

If bit 0 of the fault information in RMDR is set, the interrupt is due to a floating-point fault. In this case, a floating-point fault was queued when a microinstruction which specified both IRD and E terminated. ILOC contains the address of the user instruction following the faulted one. The data in CLOC may be considered as undefined. RMDR bits 01:02 contain a binary number equal to the length of the faulted instruction in halfwords. For RR, RX1, RX2, and RX3 floating-point user instructions, this number may be extracted from RMDR and doubled. The result, 2, 4, or 6, may be subtracted from ILOC to yield the address of the first halfword of the faulted user instruction.

If the program address returned in RMDR is equal to (CLOC-2), the emulator assumes the fault occurred during the fetch of a user instruction, unless a floating-point fault interrupt is being serviced.

TABLE 6-1 RMDR FAULT CODES

CODE	MEANING	EMULATED INTERRUPT
8X	Floating-Point Fault	ARITH
00	No faults	-
10	Not used	MAT
11	Execute protect violation	MAT
12	Write protect violation	MAT
13	Read protect violation	MAT
14	Access level violation	MAT
15	Segment limit violation	MAT
16	Nonpresent segment	MAT
17	Shared seg table size exceeded	MAT
18	Private seg table size exceeded	MAT
19	Noncorrectable memory data error	MMF
1A	Nonconfigured memory	MMF
1B	Not used	MMF
1C	Not used	MMF
1D	Not used	MMF
1E	Fullword alignment fault	ALIGN
1F	Halfword alignment fault	ALIGN

6.2.3 Primary Power Fail Interrupt (206)

A Primary Power Fail (PPF) Interrupt is generated when the power supply reports a loss of primary power. The microinstruction at control store location '206' is executed. This interrupt cannot be inhibited. The emulator fetches the PSW save pointer in main memory location X'84', forces the two least-significant bits to zero, and proceeds to save the contents of the current PSW, ILOC, all of the user's general registers, the scratchpad registers, and the FPP floating-point registers (if equipped), at sequential main memory locations starting at the indicated address. The system clear relay is released by the hardware one millisecond after the PPF interrupt, holding the system in an initialized state until the relay is reenergized by the power supply.

The microinstruction at trap location '206' must disarm interrupts.

When power is resto ed, if MCR bit 6 is set, the INIT switch on the consolette is depressed, and the emulator enters the console service routine.

6.2.4 Machine Malfunction Interrupt (205)

The machine malfunction interrupt, enabled by PSW bit 18, occurs whenever Early Power Fail (EPF) is detected, when voltage at the memories goes out of regulation, when the optional module timeout detect (MCR bit 11) recognizes a module start time failure, or if a shared memory bank (optional) goes into a power-down or off line state. Any of these conditions causes the microinstruction at control store location '205' to be executed. The microinstruction at the trap location must disarm interrupts. The reason for the interrupt is determined by using the SMCR instruction; resulting flags are shown in Table 6-2. the appropriate MCR bit is then reset for subsequent detection of similar interrupts.

TABLE 6-2 FLAGS RETURNEL BY SMCR AFTER MACHINE MALFUNCTION

FLAG				INDICATION
C	V	G	L	
1		X		Module time-out/Shared Memory Power failure
	1	X		Memory vcltage failure (NVM)
		X	1	Early power failure

6.2.4.1 Early Power Fail (EPF)

The EPF bit sets if the EPF detector shows that the primary line voltage is low, when the initialize key is depressed, or when the key-operated power switch or chassis-mounted circuit breaker is set to the STANDBY or OFF position. When any of the above events occurs, a one millisecond timer is started and the EPF bit in MCR is set (MCR bit 15). The user program may perform any necessary system shutdown procedures during this one millisecond interval. PSW bit 18 may again be set to look for memory voltage failure or module time-cut, or to prepare for the interrupt on power up. The EPF interrupt does nct reoccur. At the end of the EPF one millisecond timeout, the EPF interrupt is generated. (Refer to Section 6.2.3.)

6.2.4.2 Memory Voltage Failure

If voltage at any memory chassis goes out of regulation, that chassis asserts the NVMO signal, setting MCR bit 13. This bit cannot be reset while NVMC remains active; however, an interrupt is generated only as MCR bit 13 changes state from zero to one. This interrupt causes an EPF interrupt to occur within one millisecond. Because this delay can have a minimum value of zero milliseconds, the emulator ignores the Memory Voltage Failure (NVM) interrupt by loading CLOC from ILOC, and branching to routine TWAIT. Refer to Chapter 8.

6.2.4.3 Module Start Time Failure

When the SMCR following a machine malfunction interrupt indicates module time cut (C flag set), the MCR date must be ANDed with X'40' to determine whether a power failure has been detected for a shared memory bank (optional equipment). If the result is not zerc, then shared memory power fail has occurred. Refer to Section 6.2.4.4.

The cptional module timeout detect circuit is present if MCR bit 11 is set. Each microinstruction specifies a CPU module and issues a STARTC signal; it then waits for acknowledgement before proceeding to the next micrcinstruction. When MCR bit 11 is set, the module timeout detect circuit ensures that each STARTO is acknowledged within a certain time period. If this period elapses and no acknowledgement is returned, MCR bit 12 is set, causing a machine malfunction interrupt. This interrupt prevents the processor from waiting indefinitely for response from a module which may be damaged or has been removed from the system.

6.2.4.4 Shared Memory Power Fail

When the Shared Memory option is equipped, an interrupt is generated if the Early Power Failure detector in the Shared Memory Power Supply detects a low voltage. This interrupt is also generated when the Shared Memory Bank is placed in a power-down or off-line mode. Each processor attached to the Shared Memory System is interrupted.

If the C flag is set by an SMCR instruction following a machine malfunction interrupt, the MCR data must be ANDed with X'40' to determine whether a power failure has been detected for a Shared Memory Bank. If the result is zero, a Module Start Time Failure has occurred (refer to Section 6.2.4.3); otherwise, a Shared Memory Power Failure has occurred.

Following Shared Memory Power Fail Detect, the MOS Shared Memory System is available for a period of one millisecond. Before that the memory system enters a power-down or off-line mode. Once this mode has been entered, any attempt to access the Shared Memory Bank results in a noncorrectable memory data error, or a nonconfigured memory address fault.

There is no mechanism to indicate to the processor that Shared Memory Power has been restored. This can only be determined by software means.

6.3 EXTERNAL INTERRUPTS

Five unique external interrupt trap locations are provided in the hardware. One of these locations is dedicated to each of the four I/O attention lines; the fifth is dedicated to the console attention line.

6.3.1 Console Attention Interrupts (204)

The console attention interrupt is queued whenever the momentary EXE/HLT switch on the consolette is depressed, or when an instruction read/decode cycle is completed in the microcode and the single-step switch on the consolette is in the SNGL position.

The console attention interrupt is tested only during the decode phase of a microinstruction. The implication is that a console interrupt can be serviced only at the end of a user's instruction. When the console interrupt is taken, the microinstruction at control store location '204' is executed. The microinstruction at the trap location must disarm interrupts.

The Branch and Disable Console interrupt microinstruction momentarily disables the console attention signal so that lower priority I/O interrupts can be examined. The interrupt is disabled for this one microinstruction only.

The console interrupt must be cleared by resetting MCR bit 10 before exiting the console service routine.

6.3.2 I/O Interrupts (203, 202, 201, 200)

If individually enabled by the user, a peripheral device controller may request processor service when the device itself is ready to transfer data. The processor has four priority interrupt lines for handling device interrupt requests. Whenever an external interrupt occurs, it remains pending until the processor recognizes and services the interrupt, or until the interrupt is programmed reset at the device interface.

The four I/O attention lines are processed in the priority shown below:

Priority	Attention Line	Trap Location
First	0	203
Second	1	202
Third	2	201
Fourth	3	200

PSW bits 17, 20, 25, 26, and 27 affect the enable status of the four I/O attention lines as shown in Table 1-3. The emulator handles I/O interrupts in one of two ways, depending upon data in main memory.

When the interrupt is serviced by the emulator, the address of the interrupting device is doubled and used as an index into the Interrupt Service Pointer Table at absolute address X'D0'. If the halfword entry at the resulting address has a zero as its least significant bit, an immediate interrupt is emulated. If the entry has a one as its least significant bit, the auto driver channel is activated.

CHAPTER 7
INSTRUCTION EXECUTION

7.1 INTRODUCTION

User instructions are maintained in the main memory. The user instruction to be executed next is at the main memory address specified by the Current Location Counter (CLOC). The microprogram begins to emulate that user instruction by doing an instruction read. On the same microinstruction or on a subsequent microinstruction, the decode option is specified. Because the microprogram need not specify instruction read and decode in the same microinstruction, the instruction fetch is discussed in two phases.

7.2 INSTRUCTION READ

In response to an instruction read, the halfword whose address is in CLOC is fetched and placed in the User's Instruction Register (UIR). Simultaneously, CLOC is copied to the Instruction Location Counter (ILOC), which always points to the first halfword of the user instruction.

At the same time that the user's operation code is loaded into UIR, a decision is made whether or not additional halfwords must be fetched from memory to make up the complete instruction word. As soon as the first halfword of UIR is filled, the format ROM is interrogated to determine the instruction format. The format ROM is a separate Read-Only-Memory containing 256 4-bit words, one word for each possible user-level operation code. The nature of the data in the format ROM is shown below.

1	0	0	0	RX format
0	0	0	1	RI1 format
0	1	0	0	RI2 format
0	0	1	0	RR or Short format
0	0	0	0	RXRX format

The hardware automatically fetches the appropriate number of halfwords so that after the instruction read is performed, the UIR contains the most significant 16 bits of the instruction and the Memory Data Register (RMDR) contains the information shown in Table 7-1. For each halfword fetched, CLOC is incremented by two, so that when the entire instruction has been fetched, CLOC contains the address of the next sequential user instruction.

TABLE 7-1 STATE OF RMDR AFTER INSTRUCTION READ

INSTRUCTION FORMAT	CONTENTS OF RMDR
RR or SF	Bits 0–31: UNDEFINED
RI1	Bits 0–15: I2 FIELD OF INSTRUCTION; bits 16–31: I2 FIELD OF INSTRUCTION
RI2	Bits 0–31: I2 FIELD OF INSTRUCTION
RX1	Bit 0: 0; bit 1: 0; bits 2–15: D2 FIELD OF INSTRUCTION; bit 16: 0; bit 17: 0; bits 18–31: D2 FIELD OF INSTRUCTION
RX2	Bit 0: 1; bits 1–15: D2 FIELD OF INSTRUCTION; bit 16: 1; bits 17–31: D2 FIELD OF INSTRUCTION
RX3	Bits 0–3: 0100; bits 4–7: SX2; bits 8–31: A2 FIELD OF INSTRUCTION

The processor knows, from the output of the format ROM and from bits 16 and 17 of the second RX halfword, if a third halfword for RX3 and RI2 formats is required.

Loading the UIR has no immediate effect on the YDI and YSI registers. These registers are not modified until decode time so that the microprogram can continue using YD and YS for selecting the user's registers. (See Figure 1-1.) However, when the microprogram attempts to unload RMDR to the B bus, the data shown in Table 7-2 is received instead of the actual RMDR data.

TABLE 7-2 B BUS GATING AFTER INSTRUCTION READ

INSTRUCTION FORMAT	STATE OF B BUS WHEN UNLOADING RMDR
RR or SF	Bits 0–31: UNDEFINED
RI1	Bits 0–15: EQUALS BIT 16; Bits 16–31: I2 FIELD OF INSTRUCTION
RI2	Bits 0–31: I2 FIELD OF INSTRUCTION
RX1	Bits 0–16: ZERO; Bit 17: 0; Bits 18–31: D2 FIELD OF INSTRUCTION
RX2	Bits 0–16: EQUALS BIT 17; Bits 17–31: D2 FIELD OF INSTRUCTION
RX3	Bits 0–31: CONTENTS OF REGISTER SELECTED BY SX2

For the RI1 format, bits 0:15 of RMDR are set equal to the sign bit of the halfword in bits 16:31. For the RX1 format, bits 0:16 of RMDR are zero. For the RX2 format, bits 0:16 of RMDR are set equal to bit 17. For the RX3 format, until a microinstruction is performed that loads the Memory Address Register (MAR), any reference to RMDR as a source causes the contents of the general register whose address is in the SX2 field of the instruction to appear on the B bus instead of RMDR.

The RXRX instruction resembles a pair of adjacent RX format instructions with the op-code in the OP field of the first member of the pair. The XOP field of the second member of the RXRX instruction has no hardware significance. For the RXRX format, B bus gating reflects the RX1, RX2, or RX3 format of the first member of the RXRX instruction. The second member must be fetched from the Instruction Buffer (IB) by the microprogram. The DR2IB and DR4IB MC options are available to allow fetching this data. CLOC is incremented by 2 for every halfword fetched from the instruction buffer. No particular fullword alignment is required for DR4IB.

REGISTER TO REGISTER (RR)

0 7	8 11	12 15
OP	R1	R2

SHORT FORMAT (SF)

0 7	8 11	12 15
OP	R1	N

REGISTER AND IMMEDIATE STORAGE 1 (RI1)

0 7	8 11	12 15	16 31
OP	R1	X2	I2

REGISTER AND IMMEDIATE STORAGE 2 (RI2)

0 7	8 11	12 15	16 47
OP	R1	X2	I2

REGISTER AND INDEXED STORAGE 1 (RX1)

0 7	8 11	12 15	16	17	18 31
OP	R1	X2	0	0	D2

REGISTER AND INDEXED STORAGE 2 (RX2)

0 7	8 11	12 15	16	17 31
OP	R1	X2	1	D2

REGISTER AND INDEXED STORAGE 3 (RX3)

0 7	8 11	12 15	16	17	18	19	20 23	24 47
OP	R1	FX2	0	1	0	0	SX2	A2

RXRX

0 7	8 11	12 31/47
OP	R1	FIRST RX1,RX2 or RX3 INDEX & ADDRESS

32/48	39/55 44/60	63/79/95
XOP	XR1	SECOND RX1,RX2, or RX3 INDEX & ADDRESS

7.3 INSTRUCTION DECODE

When decode is specified, the processor first tests for any pending interrupts. If an interrupt is pending, the instruction decode is aborted and the interrupt is serviced. If no interrupt is pending, the YDI and YSI registers are updated. Twice the user's operation code is presented to the ROM address gate as the starting address of the appropriate emulation sequence and the privileged/illegal ROM is interrogated. This is a separate read-only-memory containing 256 4-bit words, one for each possible user level operation code. The privileged/illegal ROM has four outputs that are decoded from the user op-code. They are defined as per the following.

1. PRIV1 - masked with PSW23 to decode all privileged instructions

2. ILEGA - defines floating point instructions

3. ILEGB and ILEGC - defines WCS and communications assist instructions and all illegal instructions.

If the output of the privileged/illegal ROM indicates that the operation code presently in UIR is illegal, or if the specified optional unit is not present, the instruction fetch is aborted and the illegal instruction interrupt is taken. If the output of the privileged/illegal ROM indicates that the operation code presently in UIR is that of a privileged instruction and PSW bit 23 is set, the illegal instruction interrupt is taken. If the output of the privileged/illegal ROM indicates that the floating-point unit is required and PSW bit 13 is set, the illegal instruction interrupt is taken.

If no interrupt occurs, the ROM Location Counter (RLC) is set equal to twice the user's operation code and the emulation sequence begins.

7.4 OPERAND FETCH

Following instruction read, if the user instruction is register to register or short format, the second operand is available in a general register or in the instruction word itself.

If the user-level instruction is one of the register and immediate storage formats, the immediate operand is available in RMDR. All that remains to be done is to add in the contents of the specified index register. The first microinstruction of a register and immediate storage format instruction could be:

```
A    WMDR,YX,RMDR
```

After the instruction, the WMDR used for writing to memory contains the sum of the I2 field of the instruction and the contents of the indexing general register specified by the X2 field of the instruction. Note that the RMDR used for reading from memory is not modified.

If the user-level instruction format is one of the register and indexed storage types, a memory read or write operation may be performed after calculating the effective second operand address. For example, the emulation sequence could begin as follows:

```
A MAR,YX,RMDR,DR2      Calculate address and read halfword
```

or

```
A MAR,YX,RMDR,DW4      Calculate address

L WMDR,YD,DW4          Copy general register to WMDR
```

depending upon whether a memory read or write is to be performed, and whether the second operand is to be a halfword (2 bytes), or a fullword (4 bytes).

If the instruction format is RX1, the sum of RMDR and the contents of of the indexing general register specified by the X2 field of the instruction replaces the contents of MAR. Referring to Figure 1-1, the output of MAR is passed, unaltered, through the 24-bit adder to the Memory Address Translator (MAT) controller. The MAT presents this address, or a translated address, to the memory bus and the memory read is started. As soon as the data becomes available in RMDR, the instruction fetch is over.

If the instruction format is RX2, the sum of RMDR and the contents of the indexing general register specified by the X2 field of the instruction replaces the contents of MAR. The output of MAR is added to the contents of CLOC (equal to ICOC+4). This sum is presented via the MAT to the memory bus and the memory read is started.

If the instruction format is RX3, the sum of the contents of the indexing general register specified by the SX2 field of the instruction and the contents of the indexing general register specified by the FX2 field of the instruction replaces the contents of MAR. If the format is RX3, until MAR is loaded, any reference to RMDR as a source causes the second level index register (SX2) to be accessed instead. The output of MAR is added to the contents of RMDR and this sum is presented via the MAT to the memory bus. The memory read is then begun.

Only when the Add and Transfer microinstruction is the first of an RX emulation sequence, the condition for transfer is whether or not the user instruction format is RX2 rather than the state of the ALU carry. The microprogram can know the RX format of a user instruction if the first microinstruction of the emulation sequence is a conditional RR transfer instruction.

CHAPTER 8
EMULATOR

8.1 INTRODUCTION

The following sections describe major aspects of the processor emulator microprogram. The microprogram listing, provided in the Model 3250 Processor Maintenance Manual, Publication Number 47-029, is well annotated and recommended as a self-explanatory reference for details of the simpler microcode sequences.

8.2 SYSTEM INITIALIZATION

8.2.1 General Information

On power-up or following initialization, microcode execution begins at control store address '001'. The FAULT lamp on the consolette is lit. A basic check of the machine's major internal buses and registers is performed; any detected failure causes the microcode to loop in the failing mode as long as the failure is demonstrated.

The contents of the Machine Control Register (MCR) are then tested. The Non-Valid Memory (NVM) bit in MCR is set if memory voltage was not maintained within limits since the last time the bit was programmed to zero. If the NVM bit is set, the contents of memory are assumed to have been lost, and a cold start sequence is performed. Otherwise, a warm start sequence is performed. The FAULT lamp is turned off at the successful termination of either sequence.

Before testing for an enabled LSU, the INIT bit of the MCR is tested. If this bit is set, the initialize switch on the consolette is depressed, and routine CONSER is entered.

8.2.2 Cold Start

If the NVM bit is set in MCR following system initialization, the first 256 kbytes of memory are written, with each fullword containing its address. This causes the ECC syndrome bits to agree with the data for these fullwords, and prevents spurious ECC failure indications from occurring within this area of memory due to a prior power failure.

The single-precision and double-precision floating-point registers are loaded with zero, the scratchpad registers are loaded with the address of the illegal instruction interrupt emulation routine ILEGAL, and each user general register is loaded with its set number and register address.

Next, the first 256 kbytes of memory are tested to see if they can retain data. A nondestructive test is used; original data is restored on successful completion of the test. Any detected failure causes the microcode to loop in the failing mode as long as the failure is demonstrated. If the system is initialized during this test, the contents of a fullword of memory under test may be lost.

When the memory test has been successfully completed, the Loader Storage Unit (LSU, device '05') is addressed. If false SYNC occurs, the LSU is not present or not enabled. In this case, PSW is loaded with '008000' (wait bit only), CLOC is loaded with 'FFFFFE' , and the console service routine CONSER is entered. The NVM bit is reset in routine CONSER. (See Section 8.2.5.) Otherwise, if the LSU is enabled, routine BOOT is entered. (See Section 8.2.4.) Software must reset the NVM bit following successful load from the LSU, with the RMVF instruction.

8.2.3 Warm Start

If the NVM bit is not set in MCR following system initialization, it is assumed that memory data was not lost as a result of power failure.

The first 256 kbytes of memory are tested to see if they can retain data. A nondestructive test is used; original data is restored on successful completion of the test. Any detected failure causes the microcode to loop in the failing mode as long as the failure is demonstrated. If the system is initialized during this test, the contents of the fullword of memory under test may be lost.

The Loader Storage Unit (LSU, device '05') is then addressed. If no false SYNC occurs, the LSU is present and enabled, and routine BOOT is entered. (See Section 8.2.4.)

If the LSU is not present or not enabled, the fullword pointer contained in memory at physical address '84' is fetched and aligned to fullword boundary. The PSW, LOC, eight sets of user's general registers, the module 7 scratchpad registers, and the single-precision and double-precision floating-point registers (if the machine is so equipped) are loaded from contiguous fullwords in memory beginning at the address indicated by the pointer. The console status fullword at memory location '28' is then fetched. If bit 0 of the status is zero, a power-up machine malfunction interrupt is emulated by routine TMMF, according to the state of PSW bit 18. Otherwise, the console service routine CONSER is entered. (See Section 8.2.5.)

8.2.4 Loader Storage Unit

If MCR bit 6 is set, routine CONSER is entered. MCR bit 6 is set while the INIT button is depressed on the consolette. If MCR bit 6 is not set, then, if the LSU is present and enabled, Routine BOOT proceeds to read the first eight bytes of data from the LSU. The nature of this data is as follows:

First two bytes	=	Least significant 16 bits of a new PSW
Second two bytes	=	Least significant 16 bits of a new LOC
Third two bytes	=	Least significant 16 bits of an absolute start address
Fourth two bytes	=	Least significant 16 bits of an absolute end address

The most significant 16 bits of PSW and LOC are set to zero. As a consequence, the location count value can only address a location within the first 64 kb of main memory. The start and end addresses identify an area in the first 64kb of main memory to be loaded with the ninth and successive bytes of data from the LSU.

If the start address is initially greater than the end address, routine CONSER is entered; otherwise, data bytes are read from the LSU and stored at successive byte locations in main memory. The start address is incremented by one for each byte read. Reading continues until the start address becomes greater than the end address, at which time routine TMMF is entered. If bit 18 of the PSW is set, a power-up machine malfunction interrupt is emulated. Otherwise, the state of the wait bit is tested.

8.2.5 Console Service Routine

The system console terminal is a full duplex asynchronous device. The microprogram, on entry to the routine CONSER, programs this device for no echoplex, maximum baud rate with seven data bits and two stop bits per character, and even parity. Local connection is assumed; modem connection is not presently supported.

Entry to CONSER causes the NVM bit of MCR to be reset, and the current PSW and LOC to be displayed on the terminal screen, followed by an operator prompt. CONSER allows the user to examine and modify PSW, LOC, general and floating-point registers, and memory. Program execution may begin from the console, breakpoint instructions may be inserted, and instructions may be executed in single-step mode if the SNGL/RUN switch on the system control panel is in the SNGL position.

8.3 INTERRUPT SUPPORT

8.3.1 Routine FAULT

This routine, detailed in Figure 8-1, is entered whenever an MAIO abort interrupt or machine malfunction interrupt occurs. In the case of an MAIO abort interrupt (vector through '207'), MR1 is loaded with the fault code and program address contained in RMDR following the fault, and the fault is reset. If RMDR bit 0 is set, a floating-point fault interrupt occurred, and routine FPPFAUL is entered to service for fault. For further details of the floating-point fault interrupt, refer to Section 6.2.2, otherwise, the steps described in the following paragraphs are performed.

The MCR bits are tested, and RFAULT is issued. If the EPF bit is set, an early power fail machine malfunction interrupt is emulated. If the STF bit is set and MCR bit 9 is zero, a start time failure machine malfunction interrupt is emulated. If the STF bit is set, and MCR bit 9 is set, a Shared Memory Power Fail machine malfunction interrupt is emulated. If the NVM bit is set, a nonvalid memory machine malfunction interrupt is ignored, as it causes a subsequent EPF interrupt. All bits in the MCR are forced reset except NVM.

If none of the above mentioned MCR bits are set, MRO was loaded with an address indicating where, in the emulation sequence, the fault occurred. If the fault occurred in the CONSER routine, it is ignored, and CONSER is reentered. If the fault occurred as a result of a machine malfunction interrupt PSW swap, the machine is stopped (Hard Stop) by loading both CLOC and ILOC with X'040', and entering routine CONSER; double faults are not recoverable without manual intervention.

If the address returned with the fault code is equal to (CLOC-2), it is assumed that the fault occurred during the fetch of a user instruction. Otherwise, it is assumed that the fault occurred while reading data from or writing data to memory. A special case exists if the fault occurred while emulating an auto driver channel operation.

If the fault code returned is in the range from '00' to '17', routine MATINT is entered, and a MAT interrupt is emulated. If the fault code is in the range from '1E' to '1F', a data format fault interrupt is emulated by routine FORFAUL6. Otherwise, a machine malfunction interrupt is emulated as follows. PSW and ILOC are stored in the doubleword at memory location '20'. If the fault occurred as a result of emulating a Load Multiple instruction, the calculated second operand address is stored in the fullword at memory location '2C'. The machine malfunction status word (refer to Figure 8-2) at location '40' is adjusted according to the particular type of malfunction to be emulated; location '44' receives the program address unloaded from RMDR at the time of the fault. PSW bit 18 is forced set, and the new PSW and LOC are fetched from the doubleword at memory location '38'. Routine IWAIT is then entered.

When a machine malfunction interrupt occurs due to a memory access, RLC may continue to advance one or two microinstructions before the fault is reported. These instructions are not executed; however, when the BALR FAULT (MRO) instruction is executed at the trap location, MRO may be loaded with a value one or two greater than the expected value.

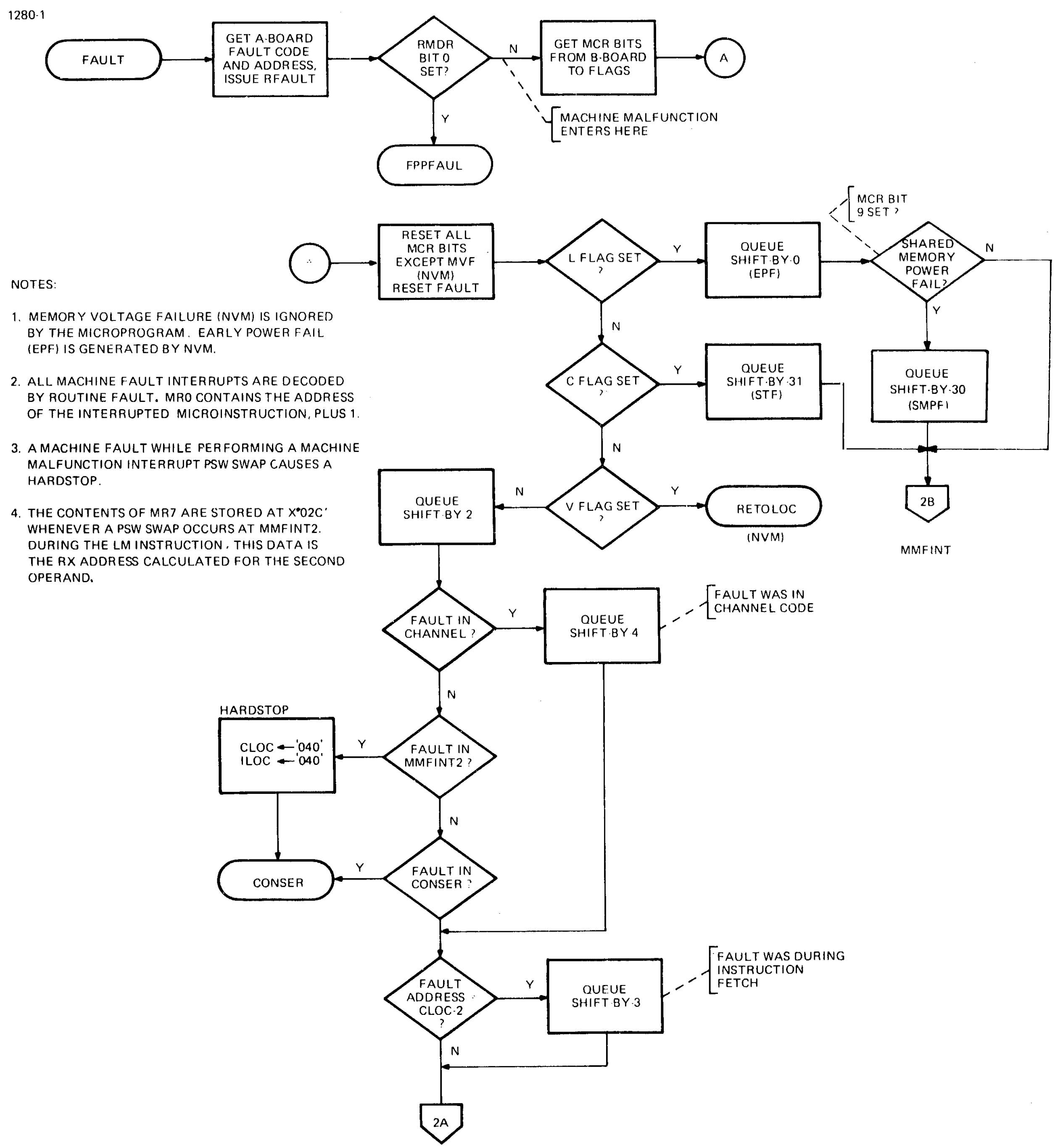


Figure 8-1 FAULT Routine

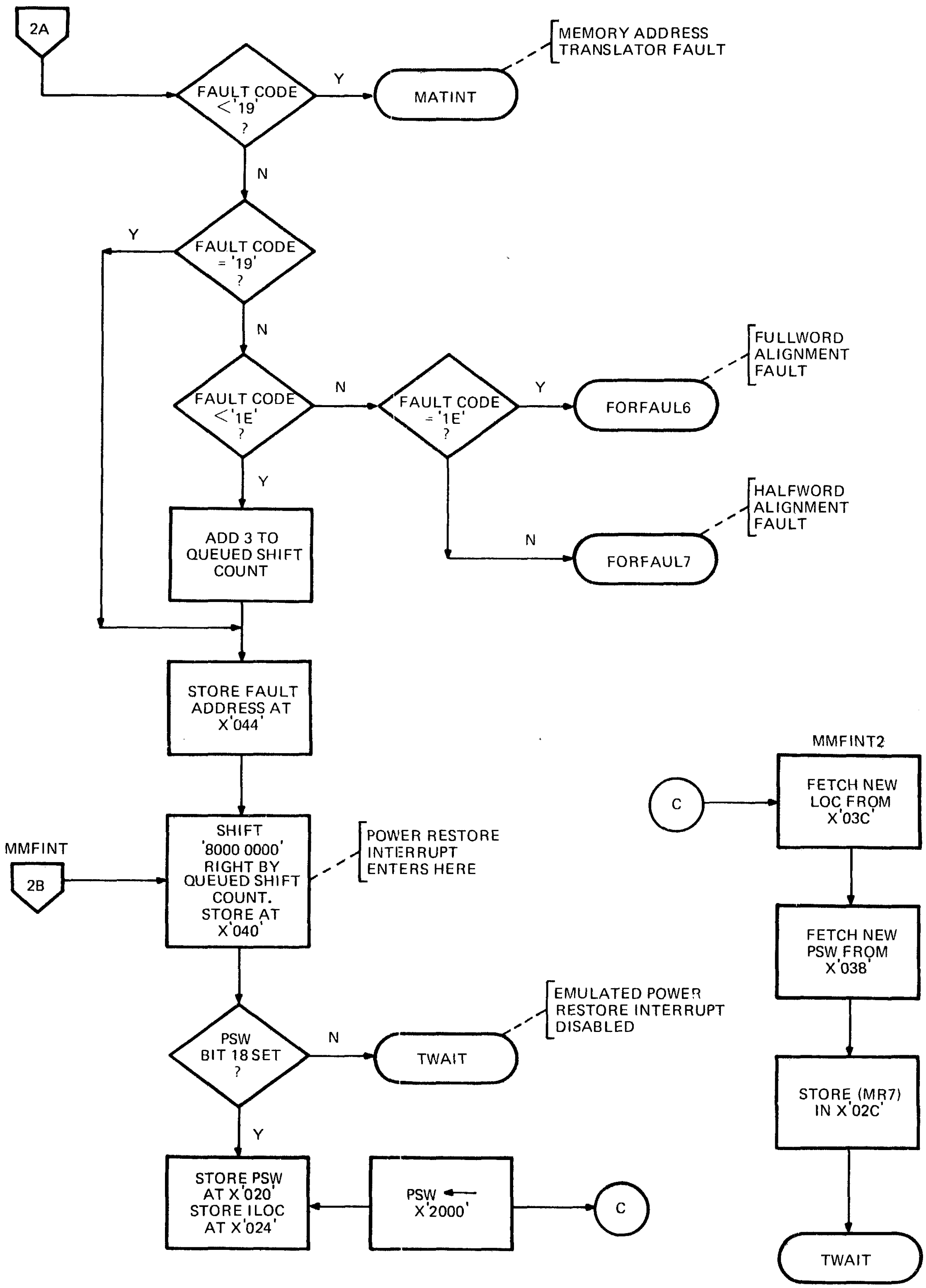


Figure 8-1 FAULT Routine (Continued)

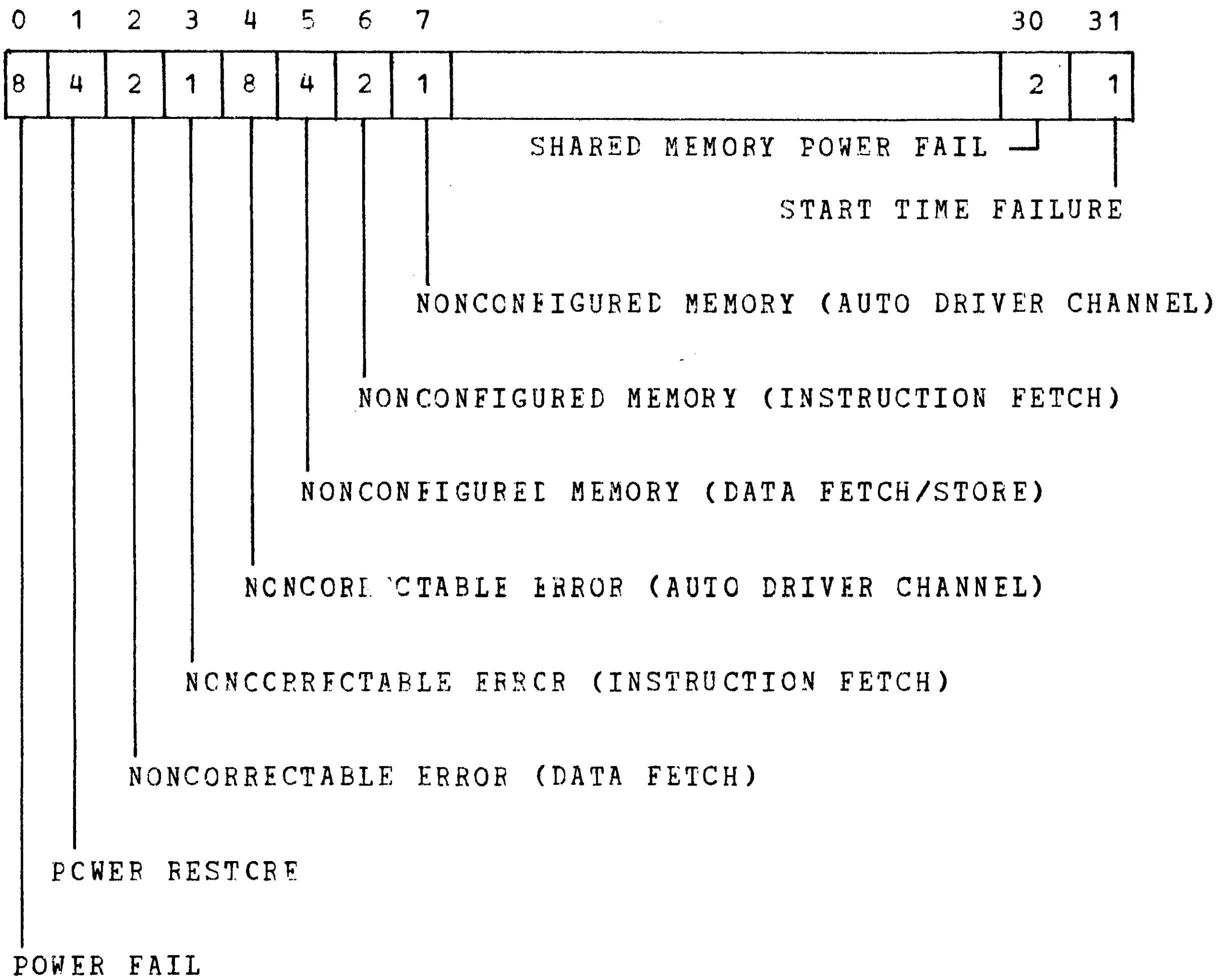


Figure 8-2 Machine Malfunction Status Word (MMSW)

8.3.2 Routine TWAIT

This routine tests the WAIT bit of the current PSW (PSW bit 16). If the bit is set, routine WAIT is entered. Otherwise, the wait lamp is turned off, and the user instruction indicated by CLOC is fetched and executed.

8.3.3 Routine WAIT

The WAIT lamp is lit by the first instruction of routine WAIT. The remainder of this routine consists of a single instruction which branches to itself, with all enabled interrupts armed. Any interrupt causes the microinstruction at the interrupt trap address to be executed.

8.3.4 Routine MATINT

This routine is entered from routine FAULT, as a result of an MAIC abort interrupt caused by the MAT controller. Routine COMSWAP fetches the MAT interrupt new PSW at memory location '90', and saves the 64-bit PSW at the time of the fault in registers 14 and 15 of the new register set. MATINT then places the code returned at the time of the fault in register 13 of the new set, and the returned program address in register 12. If the fault occurred while emulating the Load Multiple instruction, the calculated second operand address is placed in register 11.

8.3.5 Routine FORFAUL6

This routine is entered from routine FAULT, as a result of an MAIC abort sequence caused by an alignment error. The proper fault code is generated in the least significant four bits of MR0 as routine FORFAULT is called.

8.3.6 Routine FORFAULT

This routine is entered from routines FORFAUL0 through FORFAUL7, whenever a data format fault interrupt occurs or is forced by the emulator. Routine COMSWAP fetches the data format fault interrupt new PSW from memory location 'C8', and saves the 64-bit PSW at the time of the fault in registers 14 and 15 of the new register set. FORFAULT then places the code indicating the type of fault in register 13. If a halfword or fullword alignment fault occurred, the program address causing the fault is placed in register 12.

8.4 I/O INTERRUPTS

The occurrence of one of the four I/O interrupts causes the microinstruction at the respective trap location to be executed. Register 'LEVEL' is set equal to the number of the interrupt line and the interrupt is acknowledged. The returned device number is placed in register 'DEV' and routine IOINTX is entered.

The current PSW is set aside in register 'TEMP'. The halfword service pointer table entry is fetched from the memory location whose address is 'D0' plus twice the interrupting device number. A new PSW is loaded which has only bits 18 and 20 set, and selects the register set corresponding to the number of the interrupt line.

General register 0 of the newly selected set is set equal to the old PSW; general register 1 is set equal to ILOC, and general register 2 is set equal to the device number. The device is addressed and a sense status is performed. The device status byte is copied to general register 3 and the condition code. The service pointer table entry in RMDR is tested. If the least significant bit is zero, an immediate interrupt is performed; the wait lamp is turned off, and the user instruction whose address is in RMDR bits 16:31 is fetched and executed. Otherwise, if the least significant bit of the service pointer table entry is set, RMDR contains the address of a Channel Command Block (CCB) resident within the first 64 kb, and routine CHANEL is entered.

8.5 AUTO DRIVER CHANNEL

Routine CHANEL can perform a variety of functions, depending upon bits in the Channel Command Word (CCW) which is the first halfword in the CCB. (See Figure 8-3.)

	0 … 7	8	9	10	11	12	13	14	15	
0	STATUS MASK	E		S	C	B	R/W	T	F	CHANNEL COMMAND WORD
2	BUFFER 0 BYTE COUNT									
4	BUFFER 0 END ADDRESS									
8	CHECK WORD									
10	BUFFER 1 BYTE COUNT									
12	BUFFER 1 END ADDRESS									
16	TRANSLATION TABLE ADDRESS									
20	SUBROUTINE ADDRESS									

Figure 8-3 Channel Command Block

The Channel Command Word is fetched and placed in a register labeled CCW. The EXECUTE bit of CCW is tested. If the bit is zero, routine EXSUB0 is entered. If the EXECUTE bit is set, the status mask is ANDed with the actual device status in register 3. If the result is nonzero, the status check fails, and routine EXSUB1 is entered.

If the status check does not fail, the FAST bit in CCW is tested. If the bit is zero, routine NFAST is entered for normal mode CCB activities. If the bit is set, routine FASTMODE is entered.

8.5.1 Routine FASTMODE

The buffer 0 byte count is fetched into register 'COUNT'. If the count is greater than zero, it is assumed that software has not yet set it up, and routine EXAUTO is entered. If the count is not greater than zero, the buffer 0 end address is fetched in RMDR. The halfword test line is then examined. If inactive, routine BYTEIO is entered; otherwise, a halfword device controller is currently addressed, and routine HWIO is entered.

8.5.1.1 Routine BYTEIO

The buffer 0 end address is added to the contents of register 'COUNT' in MAR, and the Read/Write bit of CCW is tested. If the bit is set, a data byte is fetched from memory and output to the addressed device; if the bit is zero, a data byte is input from the addressed device and written to memory. Routine COMMON then increments the contents or register 'COUNT' by one, and updates the buffer 0 byte count in the CCB. If the new count is not greater than zero, routine EXAUTO is entered; otherwise routine EXSUB2 is entered.

8.5.1.2 Routine HWIO

The buffer 0 end address is added to the contents of register 'COUNT' in MAR and the Read/Write bit of the CCW is tested. If the bit is set, a data halfword is fetched from memory and output to the addressed device; if the bit is zero, a data halfword is input from the addressed device and written to memory. The contents of register 'COUNT' are incremented by two and the buffer 0 byte count in the CCB is updated by routine COMMON. If the new count is not greater than zero, routine EXAUTO is entered; otherwise, routine EXSUB2 is entered.

8.5.2 Routine NFAST

The buffer switch bit of the CCW is captured and ORed with binary '0010'. The result is the byte offset from the address contained in register 4 of the desired buffer byte count field in the CCW. The count is fetched. Two is added to the byte count address contained in register 'TEMP', giving the memory address of the corresponding buffer end address field of the CCW. The buffer byte count is loaded into register 'COUNT'. If the count is greater than zero, it is assumed that software has not yet set it up, and routine EXAUTO is entered. If the count is not greater than zero, the buffer end address and the byte count are added. The result, placed in both MAR and MR1, is the memory address of the data byte to participate in I/O operations. The Read/Write bit of the CCW is then tested. If the bit is set, the data byte is fetched from memory, and routine NFWRIT is entered. If the bit is zero, routine NFREAD is entered.

8.5.2.1 Routine NFWRIT

The data byte from memory is copied into register 3, and the translation bit of the CCW is tested. If the bit is set, routine TRANSL is called to translate the data byte. If the bit is zero, or on return from routine TRANSL, the data byte in RMDR is output to the addressed device. Routine REDCHK is then called to update the checkword in the CCB using the translated (I/O) byte. Upon return from REDCHK, MAR is loaded with the address of the buffer byte count, and routine COMMON3 is entered.

8.5.2.2 Routine NFREAD

A data byte is input from the addressed device and copied to WMDR. The translation bit of the CCW is tested. If the bit is set, routine TRANSL is called to translate the data byte. If the bit is zero, or on return from routine TRANSL, the byte address in MR1 is copied to MAR, and the data byte is stored in memory. Routine REDCHK is then called to update the check word in the CCB using the untranslated (I/O) byte. Upon return from REDCHK, MAR is loaded with the address of the buffer byte count, and routine COMMON3 is entered.

8.5.2.3 Routine TRANSL

The data byte in register 3 is doubled to form an index, and is added to the translation table address defined in the CCB. The corresponding halfword table entry is fetched. If the halfword is negative, the corresponding translated byte is in RMDR 24:31, and TRANSL returns to the caller. If the halfword is not negative, the contents of RMDR are doubled and copied to CLOC; the WAIT indicator is reset, and the user instruction indicated by CLOC is executed. Data available to the user's translation routine is shown in Table 8-1.

8.5.2 Routine NFAST

The buffer switch bit of the CCW is captured and ORed with binary '0010'. The result is the byte offset from the address contained in register 4 of the desired buffer byte count field in the CCW. The count is fetched. Two is added to the byte count address contained in register 'TEMP', giving the memory address of the corresponding buffer end address field of the CCW. The buffer byte count is loaded into register 'COUNT'. If the count is greater than zero, it is assumed that software has not yet set it up, and routine EXAUTO is entered. If the count is not greater than zero, the buffer end address and the byte count are added. The result, placed in both MAR and MR1, is the memory address of the data byte to participate in I/O operations. The Read/Write bit of the CCW is then tested. If the bit is set, the data byte is fetched from memory, and routine NFWRIT is entered. If the bit is zero, routine NFREAD is entered.

8.5.2.1 Routine NFWRIT

The data byte from memory is copied into register 3, and the translation bit of the CCW is tested. If the bit is set, routine TRANSL is called to translate the data byte. If the bit is zero, or on return from routine TRANSL, the data byte in RMDR is output to the addressed device. Routine REDCHK is then called to update the checkword in the CCB using the translated (I/O) byte. Upon return from REDCHK, MAR is loaded with the address of the buffer byte count, and routine COMMON3 is entered.

8.5.2.2 Routine NFREAD

A data byte is input from the addressed device and copied to WMDR. The translation bit of the CCW is tested. If the bit is set, routine TRANSL is called to translate the data byte. If the bit is zero, or on return from routine TRANSL, the byte address in MR1 is copied to MAR, and the data byte is stored in memory. Routine REDCHK is then called to update the check word in the CCB using the untranslated (I/O) byte. Upon return from REDCHK, MAR is loaded with the address of the buffer byte count, and routine COMMON3 is entered.

8.5.2.3 Routine TRANSL

The data byte in register 3 is doubled to form an index, and is added to the translation table address defined in the CCB. The corresponding halfword table entry is fetched. If the halfword is negative, the corresponding translated byte is in RMDR 24:31, and TRANSL returns to the caller. If the halfword is not negative, the contents of RMDR are doubled and copied to CLOC; the WAIT indicator is reset, and the user instruction indicated by CLOC is executed. Data available to the user's translation routine is shown in Table 8-1.

8.5.3 Exit Routines Used by FASTMODE and NFAST

Several short routines are used as common exits for routine FASTMODE and for routine NFAST. These exit routines are described in the following paragraphs.

8.5.3.1 Routine EXAUTO

This routine is entered when an auto driver channel operation has been completed, and an interrupt at the user level is not desired. The entry PSW and LOC are restored from registers 0 and 1. After PSW is restored, interrupts are collectively armed, and routine TWAIT is entered. (See Section 8.3.2.)

8.5.3.2 Routine EXSUBO

This routine is entered when the Execute bit in the CCW is seen to be zero at entry to routine CHANEL. MRO contains zero when routine EXSUB is entered. (See Section 8.5.3.5.)

8.5.3.3 Routine EXSUB1

This routine is entered when the result of ANDing the status of the interrupting device with the status mask in the CCW is not zero. MRO contains a small negative value as routine EXSUB is entered. (See Section 8.5.3.5.)

8.5.3.4 Routine EXSUB2

This routine is entered when a result greater than zero is yielded by incrementing a buffer byte count. MRO contains a small positive value as routine EXSUB is entered.

8.5.3.5 Routine EXSUB

This routine is entered from routine EXSUBO, EXSUB1, or EXSUB2. The subroutine address is fetched from the CCB. This halfword is forced even and copied to CLOC. The contents of MRO are loaded to NULL, resulting in a condition code setting of 0, 1, or 2 (no flags, L flag, or G flag). The wait lamp is turned off, and the user instruction indicated by CLOC is fetched and executed.

INDEX

S

T

U V

W X

Y Z

PUBLICATION COMMENT FORM

Please use this postage-paid form to make any comments, suggestions, criticisms, etc. concerning this publication.

From ______________________ Date ______________________

Title ______________________ Publication Title ______________________

Company ______________________ Publication Number ______________________

Address ______________________

____

____

FOLD FOLD

Check the appropriate item.

☐ Error Page No. ________ Drawing No. ______________

☐ Addition Page No. ________ Drawing No. ______________

☐ Other Page No. ________ Drawing No. ______________

Explanation:

FOLD FOLD

CUT ALONG LINE

Fold and Staple
No postage necessary if mailed in U.S.A.

STAPLE

STAPLE

FOLD

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 22 OCEANPORT, N.J.

POSTAGE WILL BE PAID BY ADDRESSEE

PERKIN-ELMER
Computer Systems Division
2 Crescent Place
Oceanport, NJ 07757

TECH PUBLICATIONS DEPT. MS 322A

FOLD

FOLD

STAPLE

STAPLE

3250

MICROCODE

PERKIN-ELMER

PROG= M3250B   ASSEMBLED BY MICROCAL II (32-BIT)

```
                          SCRAT                                                            32500000
                          CROSS                                                            32500020
                          TARGT 3240                                                       32500030
                          SQCHK                                                            32500040

               * MODEL 3250 PROCESSOR EMULATOR 05-086R03                                   32500050
               * MICROPROGRAM WRITTEN BY KARL STEES KLEIN  DECEMBER, 1978                  32500060
               * COPYRIGHT PERKIN-ELMER CORP., 1982                                        32500070

               PAGE0.1  PARTS 19-195R03F47,19-195R03F48,19-195R03F49,19-195R03F50 R03      32500090
                        PARTS 19-195R03F51,19-195R03F52,19-195R03F53,19-195R03F54 R03      32500100
               PAGE2.3  PARTS 19-195R02F55,19-195R02F56,19-195R02F57,19-195R02F58 R02      32500110
                        PARTS 19-195R03F59,19-195R03F60,19-195R03F61,19-195R02F62 R03      32500120
               PAGE4.5  PARTS 19-195R03F63,19-195R03F64,19-195R03F65,19-195R03F66 R03      32500130
                        PARTS 19-195R03F67,19-195R03F68,19-195R03F69,19-195R03F70 R03      32500140
               PAGE6.7  PARTS 19-195R01F71,19-195R01F72,19-195R01F73,19-195R01F74 R01      32500150
                        PARTS 19-195R01F75,19-195R01F76,19-195R01F77,19-195R01F78 R01      32500160
               **       NOTE  PRIV/ILEG ROM IS 19-195R00F79.                               32500170

               * IN ALL CASES WHERE A BRANCH OR TRANSFER COULD OCCUR TO A                  32500190
               * LISTING PAGE OTHER THAN THE CURRENT ONE, THE DESTINATION                  32500200
               * PAGE NUMBER IS SHOWN IN PARENTHESES IN THE COMMENT FIELD.                 32500210

               * USER  LEVEL  INSTRUCTION  EMULATION  ENTRY-POINTS                         32500230
               * FOLLOW.  IN RESPONSE TO AN INSTRUCTION READ COMMAND                       32500240
               * THE HARDWARE READS THE NEXT USER INSTRUCTION FROM                         32500250
               * THE MAIN MEMORY LOCATION SPECIFIED BY (ILOC). TWO,                        32500260
               * FOUR, SIX, OR MORE BYTES ARE READ, DEPENDING UPON                         32500270
               * INSTRUCTION TYPE.  TWICE THE USER'S OPERATION CODE                        32500280
               * IS THE STARTING ADDRESS IN ROM OF THE APPROPRIATE                         32500290
               * EMULATION SEQUENCE.  THE OP-CODE IS  SHOWN IN THE                         32500300
               * COMMENT FIELD AND THE USER'S MNEMONIC IS THE LABEL.                        32500310

0000 0000      R0       EQU   0                                                            32500330
0000 0001      R1       EQU   1                                                            32500340
0000 0002      R2       EQU   2                                                            32500350
0000 0003      R3       EQU   3                                                            32500360
0000 0004      R4       EQU   4                                                            32500370
0000 0005      R5       EQU   5                                                            32500380
0000 0006      R6       EQU   6                                                            32500390
0000 0007      R7       EQU   7                                                            32500400
0000 0008      R8       EQU   8                                                            32500410
0000 0009      R9       EQU   9                                                            32500420
0000 000A      R10      EQU   10                                                           32500430
0000 000B      R11      EQU   11                                                           32500440
0000 000C      R12      EQU   12                                                           32500450
0000 000D      R13      EQU   13                                                           32500460
0000 000E      R14      EQU   14                                                           32500470
0000 000F      R15      EQU   15                                                           32500480
0000 0000      FREEWORD EQU   0                 TRACER                                     32500490
```

ROM SEGMENT 0 - OPCODES 00 TO 1F

```
0000   17FC 8240     TRAP00    BALD  ILEGAL(NULL)        HARDWARE TRAP FOR BAD DECODE (P.18)  32500510
                     *                                   ON RELEASE OF SCLR0, EXECUTION STARTS32500520
0001   17FD 5D40     START     BALD  SELFTEST(NULL)      AT LOCATION '001'. (P.49).           32500530
                     *                                                   * 01 *               32500540
0002   2A1C 1F80     BALR      A     MR0,CLOC,NULL       INCREMENTED LOC TO MR0               32500550
0003   235F 1C05               LX    CLOC,YS,BALR1       NEW LOC FROM YS                      32500560
                     *                                                   * 02 *               32500570
0004   17EC 01D2     BTCR      BALT  BRR(NULL),IRD       BRANCH IF MASK TRUE                  32500580
0005   2B3F 1812     BALR1     L     YD,MR0,IRD          YD GETS OLD INCREMENTED LOC.         32500590
                     *                                                   * 03 *               32500600
0006   13EC 01D2     BFCR      BALF  BRR(NULL),IRD       BRANCH IF MASK FALSE                 32500610
0007   235F 1C37     BRR       LX    CLOC,YS,EXIT3       LOAD LOC (P.3)                       32500620
                     *                                                   * 04 *               32500630
0008   2B39 5C32     NR        N     YD,YD,YS,IRD,E                                           32500640
0009   0004 0000     BIT13     DC    '00040000'          CONSTANT                             32500650
                     *                                                   * 05 *               32500660
000A   2BF9 0C32     CLR       S     NULL,YD,YS,IRD,E                                         32500670
000B   E3FF FFFF     BI03.050  DC    'E3FFFFFF'          CONSTANT                     R02     32500680
                     *                                                   * 06 *               32500690
000C   2B39 7C32     OR        O     YD,YD,YS,IRD,E                                           32500700
000D   0000 4000     BIT17     DC    '00004000'          CONSTANT                             32500710
                     *                                                   * 07 *               32500720
000E   2B39 6C32     XR        X     YD,YD,YS,IRD,E                                           32500730
000F   0001 0000     BIT15     DC    '00010000'          CONSTANT                             32500740
                     *                                                   * 08 *               32500750
0010   2B3F 1C32     LR        L     YD,YS,IRD,E                                              32500760
0011   4E00 0000     CONST4E   DC    '4E000000'          CONSTANT                             32500770
                     *                                                   * 09 *               32500780
0012   221F 1C31     CR        LX    MR0,YS,C2           GET SECOND OPERAND (P.3)             32500790
0013   003E 0000     BI10.14   DC    '003E0000'          CONSTANT                             32500800
                     *                                                   * 0A *               32500810
0014   2B39 1C32     AR        A     YD,YD,YS,IRD,E                                           32500820
0015   FFFF 0000     BI00.15   DC    'FFFF0000'          CONSTANT                             32500830
                     *                                                   * 0B *               32500840
0016   2B39 0C32     SR        S     YD,YD,YS,IRD,E                                           32500850
0017   0000 FFFF     BI16.31   DC    '0000FFFF'          CONSTANT                             32500860
                     *                                                   * 0C *               32500870
0018   13F9 3900     MHR       BAL   MHR1(NULL)          (P.44)                               32500880
0019   0000 8000     BIT16     DC    '00008000'          CONSTANT                             32500890
                     *                                                   * 0D *               32500900
001A   13F9 3B40     DHR       BAL   DHR1(NULL)          (P.44)                               32500910
                     *                                   .                            R02     32500920
001B   3673 605B     LCER1     XI    MR3,MR3,BIT00,I     REVERSE SIGN BIT             R02     32500930
001C   CBF9 29B2               LE    YD,MR3,IRD,E        LOAD COMPLEMENT.             R02     32500940

001D   2B5F 1C80     LPSWR1    L     CLOC,YD             NEW LOC FROM R2+1            R02     32500960
001E   2BBF 1C11               L     PSW,YS,@LOC         NEW PSW FROM R2              R02     32500970
001F   13F8 9200               BAL   QTEST(NULL)         CHECK SYSTEM QUEUE SERVICE (P.22)    32500980
```

ROM SEGMENT 0 - OPCODES 00 TO 1F

```
                          101 *                                                       * 10 *      32501000
0020   2B39 8EB2          102 SRLS     SRL   YD,YD,YSI,IRD,E                                      32501010
0021   CE00 0000          103 CONSTCE  DC    'CE000000'           CONSTANT                        32501020
                          104 *                                                       * 11 *      32501030
0022   2B39 9EB2          105 SLLS     SLL   YD,YD,YSI,IRD,E                                      32501040
0023   FFFF 7FFF          106 BIT160   DC    'FFFF7FFF'           CONSTANT                        32501050
                          107 *                                                       * 12 *      32501060
0024   321D 5008          108 CHVR     NI    MR0,PSW,8            SAVE PREVIOUS CARRY             32501070
0025   13F9 2E00          109                BAL   CHVR1(NULL)          (P.43)                          32501080
                          110 *                                                       * 13 *      32501090
0026   CA7F 1C00          111 LPER     RRE   MR3,YS               GET SPFP DATA               R02 32501100
0027   3673 5073          112          NI    MR3,MR3,BI01.31,I    FORCE POSITIVE              R02 32501110
0028   CBF9 29B2          113          LE    YD,MR3,IRD,E         LOAD SPFP, SET CC, EXIT.    R02 32501120
                          114 *                                   .                           R02 32501130
0029   221F 1DB1          115 C1RX     LX    MR0,RMDR,C2          MEMORY COMPARAND; GO COMPARE    32501140
                          116 *                                                       * 15 *      32501150
002A   CA9F 1C02          117 LGER     RRE   MR4,YS,IR            GET SPFP REGISTER               32501160
002B   2B3F 1A30          118          L     YD,MR4,D,E           COPY TO GENERAL REG, EXIT, CC SET 32501170
                          119 *                                                       * 16 *      32501180
002C   2A1F 1F00          120 LGDR     L     MR0,YDI              SAVE R1 SPEC                    32501190
002D   13F9 93C0          121          BAL   LGDR1(NULL)          (P.56)                          32501200
                          122 *                                                       * 17 *      32501210
002E   C27F 1C1B          123 LCER     RREX  MR3,YS,LCER1         READ SPFP DATA (P.2)        R02 32501220
002F   2BF9 0830          124 C3       S     NULL,YD,MR0,D,E      COMPARE, SET CC, EXIT.      R02 32501230
                          125 *                                                       * 18 *      32501240
0030   23DF 3E9D          126 LPSWR    AINCX YDI,NULL,YSI,LPSWR1 (P.2)                    R02 32501250
                          127 *                                                                   32501260
0000 0031                 128 C1RI     EQU   *                                                    32501270
0031   2BF9 6802          129 C2       X     NULL,YD,MR0,IR       COMPARE SIGNS:                  32501280
0032   17E4 0BC0          130          BALNL C3(NULL)             BRANCH: SIGNS ALIKE.            32501290
0033   3219 C001          131          SRAI  MR0,YD,1             PROPAGATE 1ST OP SIGN           32501300
0034   2BF0 3830          132          AINC  NULL,MR0,MR0,D,E     SET CONDITION CODE              32501310
                          133 *                                                                   32501320
0035   2B9A 1D80          134 B        A     MAR,YX,RMDR          CALCULATE ADDRESS               32501330
0036   2B5F 1E00          135          L     CLOC,MAR             LOAD NEW LOC                    32501340
0037   2BFF 1F92          136 EXIT3    L     NULL,NULL,IRD        EXIT.                           32501350
                          137 *                                                       * 1C *      32501360
0038   2B3B EC12          138 MR       M     YD,YDP1,YS,IRD       MULTIPLY, EXIT.                 32501370
0039   0000 FFFE          139 BI16.30  DC    '0000FFFE'           CONSTANT                        32501380
                          140 *                                                       * 1D *      32501390
003A   2A5F 1C80          141 DR       L     MR2,YD               SAVE DIVIDEND                   32501400
003B   2A7B 1F80          142          A     MR3,YDP1,NULL        .                               32501410
003C   2A9F 1C00          143          L     MR4,YS               REMEMBER DIVISOR                32501420
003D   2B3B FC00          144          D     YD,YDP1,YS           DIVIDE                          32501430
003E   13F4 96D2          145          BALV  DFAULT(NULL),IRD     ERROR IF V FLAG. (P.23)         32501440

003F   0000 0000          147          DC    FREEWORD             .                           R02 32501460
```

ROM SEGMENT 1 - OPCODES 20:3F

```
0040                    ORG   '040'              .                          R02  32501480
                  *                                                 * 20 *       32501490
0040  17EC 1052   BTBS     BALT  BBS(NULL),IRD      BRANCH IF MASK TRUE               32501500
0041  235E 0E83   BBS      SX    CLOC,ILOC,YSI,BBS1 DECREMENT BY TWICE YSI            32501510
                  *                                                 * 21 *       32501520
0042  17EC 1152   BTFS     BALT  BFS(NULL),IRD      BRANCH IF MASK TRUE               32501530
0043  235C 0EA9   BBS1     SX    CLOC,CLOC,YSI,EXIT5 GO EXIT WITH NEW LOC.            32501540
                  *                                                 * 22 *       32501550
0044  13EC 1052   BFBS     BALF  BBS(NULL),IRD      BRANCH IF MASK FALSE              32501560
0045  235E 1E87   BFS      AX    CLOC,ILOC,YSI,BFS1 INCREMENT BY TWICE YSI            32501570
                  *                                                 * 23 *       32501580
0046  13EC 1152   BFFS     BALF  BFS(NULL),IRD      BRANCH IF MASK FALSE              32501590
0047  235C 1EA9   BFS1     AX    CLOC,CLOC,YSI,EXIT5 GO EXIT WITH NEW LOC.            32501600
                  *                                                 * 24 *       32501610
0048  2B3F 1EB2   LIS      L     YD,YSI,IRD,E                                         32501620
0049  FF00 0000   BI00.07  DC    'FF000000'         CONSTANT                    R02  32501630
                  *                                                 * 25 *       32501640
004A  2B3F 0E82   LCS      S     YD,NULL,YSI,IR     SUBTRACT TO TWO'S COMP            32501650
004B  2BFF 1CB0            L     NULL,YD,D,E        SET G, L                          32501660
                  *                                                 * 26 *       32501670
004C  2B39 1EB2   AIS      A     YD,YD,YSI,IRD,E                                      32501680
004D  0000 2800   BI1820   DC    '00002800'         CONSTANT                          32501690
                  *                                                 * 27 *       32501700
004E  2B39 0EB2   SIS      S     YD,YD,YSI,IRD,E                                      32501710
004F  4000 0000   BIT01    DC    '40000000'         CONSTANT                          32501720
                  *                                                 * 28 *       32501730
0050  CBF9 2C32   LER      LE    YD,YS,IRD,E        LOAD SPFP REGISTER, SET CC  R02  32501740
0051  0000 2000   BIT18    DC    '00002000'         CONSTANT                    R02  32501750
                  *                                                 * 29 *       32501760
0052  CBF9 3C32   CER      CER   YD,YS,IRD,E        COMPARE, SET CC, EXIT.            32501770
0053  FFFF 8000   BI00.16  DC    'FFFF8000'         CONSTANT                          32501780
                  *                                                 * 2A *       32501790
0054  CBF9 4C32   AER      AER   YD,YS,IRD,E        ADD, SET FLAGS              R02  32501800
0055  00FF 0000   BI08.15  DC    '00FF0000'         CONSTANT                    R02  32501810
                  *                                                 * 2B *       32501820
0056  CBF9 5C32   SER      SER   YD,YS,IRD,E        SUBTRACT, SET FLAGS         R02  32501830
0057  0001 FFFF   BI15.31  DC    '0001FFFF'         CONSTANT                    R02  32501840
                  *                                                 * 2C *       32501850
0058  CBF9 6C32   MER      MER   YD,YS,IRD,E        MULTIPLY, SET FLAGS         R02  32501860
0059  0080 0000   BIT08    DC    '00800000'         CONSTANT                    R02  32501870
                  *                                                 * 2D *       32501880
005A  CBF9 7C32   DER      DER   YD,YS,IRD,E        DIVIDE, SET FLAGS           R02  32501890
005B  8000 0000   BIT00    DC    '80000000'         CONSTANT                    R02  32501900
                  *                                                 * 2E *       32501910
005C  CA1F 1C00   FXR      RRE   MR0,YS             ARGUMENT TO MR0                   32501920
005D  13F9 8E40            BAL   FXR1(NULL)         (P.55)                            32501930
                  *                                                 * 2F *       32501940
005E  12D8 4440   FLR      BAL   FLR1(MR6)          (P.10)                            32501950
005F  CBF9 28B0            LE    YD,MR1,D,E         EXECUTED INSTRUCTION              32501960
```

ROM SEGMENT 1 - OPCODES 20:3F

```
0060   33F5 5040    STFAIL2  NI    NULL,MR5,'040'       IS FAULT STFAIL OR SMPF ?        R03  32501980
0061   13E0 8340             BALZ  STFAIL(NULL)         BRANCH: STFAIL. CODE 00000001 (P.19) 32501990
0062   327F 101E             LI    MR3,30               SET CODE 00000002,               R03  32502000
0063   13F8 8C00             BAL   MMFINT(NULL)         SERVICE SMPF (P.20)              R03  32502010
                    *                                                            * 32 *       32502020
0064   13F9 80C0    PBR      BAL   PBR1(NULL)           (P.53)                                32502030
0065   FF7F FFFF    BIT080   DC    'FF7FFFFF'           CONSTANT                              32502040
                    *                                                            * 33 *       32502050
0066   12D9 9580    LPDR     BAL   LPDR1(MR6)           LOAD POSITIVE DOUBLE (P.56)           32502060
0067   3694 5073             NI    MR4,MR4,BI01.31,I    EXECUTED INSTRUCTION                  32502070
                    *                                                            * 34 *       32502080
0068   3338 B010    EXHR     RLLI  YD,YS,16             EXCHANGE HALFWORDS                    32502090
0069   2BFF 1F92    EXIT5    L     NULL,NULL,IRD        EXIT.                                 32502100
                    *                                                                         32502110
006A   0000 0000             DC    FREEWORD             .                                R02  32502120
006B   0000 0000             DC    FREEWORD             .                                R02  32502130
006C   0000 0000             DC    FREEWORD             .                                R02  32502140
006D   0000 0000             DC    FREEWORD             .                                R02  32502150
                    *                                                            * 37 *       32502160
006E   12D9 9580    LCDR     BAL   LCDR1(MR6)           LOAD COMPLEMENT DOUBLE (P.56)         32502170
006F   3694 605B             XI    MR4,MR4,BIT00,I      EXECUTED INSTRUCTION                  32502180
                    *                                                            * 38 *       32502190
0070   CBF9 AC32    LDR      LD    YD,YS,IRD,E          LOAD, SET FLAGS                  R02  32502200
0071   0000 F800    BI16.20  DC    '0000F800'           CONSTANT                         R02  32502210
                    *                                                            * 39 *       32502220
0072   CBF9 BC32    CDR      CDR   YD,YS,IRD,E          COMPARE, SET CC, EXIT.                32502230
0073   7FFF FFFF    BI01.31  DC    '7FFFFFFF'           CONSTANT                              32502240
                    *                                                            * 3A *       32502250
0074   CBF9 CC32    ADR      ADR   YD,YS,IRD,E          ADD, SET FLAGS                   R02  32502260
0075   0000 7FFF    BI17.31  DC    '00007FFF'           CONSTANT                         R02  32502270
                    *                                                            * 3B *       32502280
0076   CBF9 DC32    SDR      SDR   YD,YS,IRD,E          SUBTRACT, SET FLAGS              R02  32502290
0077   0000 0000             DC    FREEWORD             .                                R02  32502300
                    *                                                            * 3C *       32502310
0078   CBF9 EC32    MDR      MDR   YD,YS,IRD,E          MULTIPLY, SET FLAGS              R02  32502320
0079   AAAA AAAA    TENS     DC    'AAAAAAAA'           CONSTANT                         R02  32502330
                    *                                                            * 3D *       32502340
007A   CBF9 FC32    DDR      DDR   YD,YS,IRD,E          DIVIDE, SET FLAGS                R02  32502350
007B   5555 5555    FIVES    DC    '55555555'           CONSTANT                         R02  32502360
                    *                                                            * 3E *       32502370
007C   CA1F 9C00    FXDR     RRD   MR0,YS               ARGUMENT TO MR0                       32502380
007D   13F9 8BC0             BAL   FXDR1(NULL)          (P.55)                                32502390
                    *                                                            * 3F *       32502400
007E   12D8 4440    FLDR     BAL   FLDR1(MR6)           (P.10)                                32502410
007F   CBF9 A8B0             LD    YD,MR1,D,E           EXECUTED INSTRUCTION                  32502420
```

ROM SEGMENT 2 - OPCODES 40:5F

```
0080                              ORG   '080'             .                            R02  32502440
                         *                                                   * 40 *          32502450
0080   239A 1D9C         STH      AX    MAR,YX,RMDR,STH1  CALCULATE ADDRESS                  32502460
0081   8000 0001         BI0031   DC    '80000001'        CONSTANT                           32502470
                         *                                                   * 41 *          32502480
0082   2B9A 1D80         BAL      A     MAR,YX,RMDR       CALCULATE EFFECTIVE ADDRESS        32502490
0083   233C 1F85                  AX    YD,CLOC,NULL,BAL2 INCREMENTED LOC TO YD              32502500
                         *                                                   * 42 *          32502510
0084   17EC 0D52         BTC      BALT  B(NULL),IRD       BRANCH IF MASK TRUE (P.3)          32502520
0085   235F 1E1D         BAL2     LX    CLOC,MAR,EXIT6    LOAD NEW LOC                       32502530
                         *                                                   * 43 *          32502540
0086   13EC 0D52         BFC      BALF  B(NULL),IRD       BRANCH IF MASK FALSE (P.3)         32502550
0087   227B 1FA5         D1       AX    MR3,YDP1,NULL,D2  LS HALF, DIVIDEND (P.7)     R02    32502560
                         *                                                   * 44 *          32502570
0088   2B9A 1D87         NH       A     MAR,YX,RMDR,DR2                                      32502580
0089   2B39 5DB2                  N     YD,YD,RMDR,IRD,E                                     32502590
                         *                                                   * 45 *          32502600
008A   2B9A 1D87         CLH      A     MAR,YX,RMDR,DR2                                      32502610
008B   2BF9 0DB2                  S     NULL,YD,RMDR,IRD,E                                   32502620
                         *                                                   * 46 *          32502630
008C   2B9A 1D87         OH       A     MAR,YX,RMDR,DR2                                      32502640
008D   2B39 7DB2                  O     YD,YD,RMDR,IRD,E                                     32502650
                         *                                                   * 47 *          32502660
008E   2B9A 1D87         XH       A     MAR,YX,RMDR,DR2                                      32502670
008F   2B39 6DB2                  X     YD,YD,RMDR,IRD,E                                     32502680
                         *                                                   * 48 *          32502690
0090   2B9A 1D87         LH       A     MAR,YX,RMDR,DR2                                      32502700
0091   2B3F 1DB2                  L     YD,RMDR,IRD,E                                        32502710
                         *                                                   * 49 *          32502720
0092   2B9A 1D87         CH       A     MAR,YX,RMDR,DR2                                      32502730
0093   13F8 0A40                  BAL   C1RX(NULL)        (P.3)                              32502740
                         *                                                   * 4A *          32502750
0094   2B9A 1D87         AH       A     MAR,YX,RMDR,DR2                                      32502760
0095   2B39 1DB2                  A     YD,YD,RMDR,IRD,E                                     32502770
                         *                                                   * 4B *          32502780
0096   2B9A 1D87         SH       A     MAR,YX,RMDR,DR2                                      32502790
0097   2B39 0DB2                  S     YD,YD,RMDR,IRD,E                                     32502800
                         *                                                   * 4C *          32502810
0098   2B9A 1D87         MH       A     MAR,YX,RMDR,DR2   FETCH MULTIPLIER                   32502820
0099   13F9 38C0                  BAL   MH1(NULL)         (P.44)                             32502830
                         *                                                   * 4D *          32502840
009A   2B9A 1D87         DH       A     MAR,YX,RMDR,DR2   FETCH DIVISOR                      32502850
009B   13F9 3B00                  BAL   DH1(NULL)         (P.44)                             32502860
                         *                                                                   32502870
009C   2B7F 1C97         STH1     L     WMDR,YD,DW2       STORE                              32502880
009D   2BFF 1F92         EXIT6    L     NULL,NULL,IRD     EXIT.                              32502890
                         *                                                                   32502900
009E   2B7F 1C9F         ST1      L     WMDR,YD,DW4       STORE FULLWORD                     32502910
009F   2BFF 1F92                  L     NULL,NULL,IRD     EXIT.                              32502920
```

ROM SEGMENT 2 - OPCODES 40:5F

```
                            295 *                                                  * 50 *      32502940
00A0   239A 1D9E            296 ST       AX   MAR,YX,RMDR,ST1     CALCULATE ADDRESS (P.6)              32502950
00A1   13F8 96C0            297 DFAULTY  BAL  DFAULT(NULL)        (P.23)                               32502960
                            298 *                                                  * 51 *      32502970
00A2   2B9A 1D8F            299 AM       A    MAR,YX,RMDR,DR4     FETCH DATA                    R02   32502980
00A3   2B79 1DBF            300          A    WMDR,YD,RMDR,DW4,E  ADD, SET CC, STORE BACK       R02   32502990
00A4   2BFF 1F92            301 EXIT7    L    NULL,NULL,IRD       EXIT.                         R02   32503000
                            302 *                                 .                             R02   32503010
00A5   2B3B FD80            303 D2       D    YD,YDP1,RMDR        DIVIDE                               32503020
00A6   13F4 29D2            304          BALV DFAULTX(NULL),IRD   EXIT UNLESS FAULT                    32503030
00A7   229F 1DA1            305 DFAULTX  LX   MR4,RMDR,DFAULTY    DIVISOR TO TEST IN FAULT ROUTINE     32503040
                            306 *                                                  * 54 *      32503050
00A8   2B9A 1D8F            307 N        A    MAR,YX,RMDR,DR4                                          32503060
00A9   2B39 5DB2            308          N    YD,YD,RMDR,IRD,E                                         32503070
                            309 *                                                  * 55 *      32503080
00AA   2B9A 1D8F            310 CL       A    MAR,YX,RMDR,DR4                                          32503090
00AB   2BF9 0DB2            311          S    NULL,YD,RMDR,IRD,E                                       32503100
                            312 *                                                  * 56 *      32503110
00AC   2B9A 1D8F            313 O        A    MAR,YX,RMDR,DR4                                          32503120
00AD   2B39 7DB2            314          O    YD,YD,RMDR,IRD,E                                         32503130
                            315 *                                                  * 57 *      32503140
00AE   2B9A 1D8F            316 X        A    MAR,YX,RMDR,DR4                                          32503150
00AF   2B39 6DB2            317          X    YD,YD,RMDR,IRD,E                                         32503160
                            318 *                                                  * 58 *      32503170
00B0   2B9A 1D8F            319 L        A    MAR,YX,RMDR,DR4                                          32503180
00B1   2B3F 1DB2            320          L    YD,RMDR,IRD,E                                            32503190
                            321 *                                                  * 59 *      32503200
00B2   2B9A 1D8F            322 C        A    MAR,YX,RMDR,DR4                                          32503210
00B3   13F8 0A40            323          BAL  C1RX(NULL)          (P.3)                                32503220
                            324 *                                                  * 5A *      32503230
00B4   2B9A 1D8F            325 A        A    MAR,YX,RMDR,DR4                                          32503240
00B5   2B39 1DB2            326          A    YD,YD,RMDR,IRD,E                                         32503250
                            327 *                                                  * 5B *      32503260
00B6   2B9A 1D8F            328 S        A    MAR,YX,RMDR,DR4                                          32503270
00B7   2B39 0DB2            329          S    YD,YD,RMDR,IRD,E                                         32503280
                            330 *                                                  * 5C *      32503290
00B8   2B9A 1D8F            331 M        A    MAR,YX,RMDR,DR4                                          32503300
00B9   2B3B ED92            332          M    YD,YDP1,RMDR,IRD                                         32503310
                            333 *                                                  * 5D *      32503320
00BA   2B9A 1D8F            334 D        A    MAR,YX,RMDR,DR4     FETCH DIVISOR                        32503330
00BB   225F 1C87            335          LX   MR2,YD,D1           MS HALF DIVIDEND (P.6)               32503340
                            336 *                                                  * 5E *      32503350
00BC   2B9A 1D87            337 CRC12    A    MAR,YX,RMDR,DR2     CALCULATE ADDRESS           R02     32503360
00BD   13F9 5000            338          BAL  CRC121(NULL)        (P.47)                               32503370
                            339 *                                                  * 5F *      32503380
00BE   2B9A 1D87            340 CRC16    A    MAR,YX,RMDR,DR2     CALCULATE ADDRESS           R02     32503390
00BF   13F9 51C0            341          BAL  CRC161(NULL)        (P.47)                               32503400
```

ROM SEGMENT 3 - OPCODES 60:7F

```
00C0                         ORG   '0C0'                                                   32503420
                    *                                                      * 60 *          32503430
00C0   239A 1DBF    STE      AX    MAR,YX,RMDR,STE1    CALCULATE ADDRESS (P.9)        R02  32503440
00C1   0000 1000    BIT19    DC    '00001000'          CONSTANT                       R02  32503450
                    *                                                      * 61 *          32503460
00C2   2B9A 1D87    AHM      A     MAR,YX,RMDR,DR2     FETCH MEMORY DATA              R02  32503470
00C3   13F9 3500             BAL   AHM1(NULL)          (P.43)                         R02  32503480
                    *                                                      * 62 *          32503490
00C4   2B9A 1D87    PB       A     MAR,YX,RMDR,DR2     FETCH RESIDUAL                 R02  32503500
00C5   13F9 7F00             BAL   PB1(NULL)           (P.53)                              32503510
                    *                                                      * 63 *          32503520
00C6   2B9A 1D8F    LRA      A     MAR,YX,RMDR,DR4     FETCH PSTD                     R02  32503530
00C7   13F9 3F80             BAL   LRA1(NULL)          (P.45)                              32503540
                    *                                                      * 64 *          32503550
00C8   12D9 1300    ATL      BAL   LIST(MR6)           COMMON OVERHEAD (P.40)              32503560
00C9   13F9 1440             BAL   ATL1(NULL)          (P.40)                              32503570
                    *                                                      * 65 *          32503580
00CA   12D9 1300    ABL      BAL   LIST(MR6)           COMMON OVERHEAD (P.40)              32503590
00CB   13F9 1840             BAL   ABL1(NULL)          (P.41)                              32503600
                    *                                                      * 66 *          32503610
00CC   12D9 1300    RTL      BAL   LIST(MR6)           COMMON OVERHEAD (P.40)              32503620
00CD   13F9 1C00             BAL   RTL1(NULL)          (P.41)                              32503630
                    *                                                      * 67 *          32503640
00CE   12D9 1300    RBL      BAL   LIST(MR6)           COMMON OVERHEAD (P.40)              32503650
00CF   13F9 1FC0             BAL   RBL1(NULL)          (P.41)                              32503660
                    *                                                      * 68 *          32503670
00D0   2B9A 1D8F    LE       A     MAR,YX,RMDR,DR4                                         32503680
00D1   CBF9 2DB2             LE    YD,RMDR,IRD,E       .                              R02  32503690
                    *                                                      * 69 *          32503700
00D2   2B9A 1D8F    CE       A     MAR,YX,RMDR,DR4     FETCH COMPARAND                     32503710
00D3   CBF9 3DB2             CER   YD,RMDR,IRD,E       COMPARE, EXIT WITH CC SET.          32503720
                    *                                                      * 6A *          32503730
00D4   2B9A 1D8F    AE       A     MAR,YX,RMDR,DR4     FETCH ADDEND                        32503740
00D5   CBF9 4DB2             AER   YD,RMDR,IRD,E       ADD, SET FLAGS                 R02  32503750
                    *                                                      * 6B *          32503760
00D6   2B9A 1D8F    SE       A     MAR,YX,RMDR,DR4     FETCH SUBTRAHEND                    32503770
00D7   CBF9 5DB2             SER   YD,RMDR,IRD,E       SUBTRACT, SET FLAGS            R02  32503780
                    *                                                      * 6C *          32503790
00D8   2B9A 1D8F    ME       A     MAR,YX,RMDR,DR4     FETCH MULTIPLIER                    32503800
00D9   CBF9 6DB2             MER   YD,RMDR,IRD,E       MULTIPLY, SET FLAGS            R02  32503810
                    *                                                      * 6D *          32503820
00DA   2B9A 1D8F    DE       A     MAR,YX,RMDR,DR4     FETCH DIVISOR                       32503830
00DB   CBF9 7DB2             DER   YD,RMDR,IRD,E       DIVIDE, SET FLAGS              R02  32503840
                    *                                                      * 6E *          32503850
00DC   12D8 8E00    STBP     BAL   IIPCHECK(MR6)       CHECK IF INTERRUPT RESUME (P.21)R02 32503860
00DD   13F9 EEC0             BAL   STBP1(NULL)         (P.71)                              32503870
                    *                                                      * 6F *          32503880
00DE   12D8 8E00    LPB      BAL   IIPCHECK(MR6)       CHECK IF INTERRUPT RESUME (P.21)R02 32503890
00DF   13F9 DD80             BAL   LPB1(NULL)          (P.68)                              32503900
```

ROM SEGMENT 3 - OPCODES 60:7F

```
                                   393 *                                                               * 70 *       32503920
00E0    13F9 8500                  394 STD      BAL   STD1(NULL)           (P.54)                             R02   32503930
00E1    0000 0000                  395          DC    FREEWORD             .                                  R02   32503940
                                   396 *                                                               * 71 *       32503950
00E2    2B9A 1D80                  397 STME     A     MAR,YX,RMDR          CALCULATE ADDRESS                        32503960
00E3    13F8 7800                  398          BAL   STME1(NULL)          GO TO ROUTINE (P.17)                     32503970
                                   399 *                                                               * 72 *       32503980
00E4    2B9A 1D8F                  400 LME      A     MAR,YX,RMDR,DR4      FETCH FIRST DATA                   R02   32503990
00E5    13F8 57C0                  401          BAL   LME1(NULL)           GO TO ROUTINE (P.12)                     32504000
                                   402 *                                                               * 73 *       32504010
00E6    2B9A 1D87                  403 LHL      A     MAR,YX,RMDR,DR2                                               32504020
00E7    13F8 5740                  404          BAL   LHL1(NULL)           (P.12)                                   32504030
                                   405 *                                                               * 74 *       32504040
00E8    12D9 0D80                  406 TBT      BAL   COMBIT(MR6)          (P.39)                                   32504050
00E9    2BFF 1F92                  407 EXIT9    L     NULL,NULL,IRD        EXIT.                                    32504060
                                   408 *                                                               * 75 *       32504070
00EA    12D9 0D80                  409 SBT      BAL   COMBIT(MR6)          (P.39)                                   32504080
00EB    2B72 7D9B                  410          O     WMDR,MR2,RMDR,DW1    EXECUTED INSTRUCTION                     32504090
                                   411 *                                                               * 76 *       32504100
00EC    12D9 0D80                  412 RBT      BAL   COMBIT(MR6)          (P.39)                                   32504110
00ED    2B73 6D9B                  413          X     WMDR,MR3,RMDR,DW1    EXECUTED INSTRUCTION                     32504120
                                   414 *                                                               * 77 *       32504130
00EE    12D9 0D80                  415 CBT      BAL   COMBIT(MR6)          (P.39)                                   32504140
00EF    2B72 6D9B                  416          X     WMDR,MR2,RMDR,DW1    EXECUTED INSTRUCTION                     32504150
                                   417 *                                                               * 78 *       32504160
00F0    2B9A 1D8F                  418 LD       A     MAR,YX,RMDR,DR4      FETCH FIRST DATA                   R02   32504170
00F1    13F9 8480                  419          BAL   LD1(NULL)            (P.54)                                   32504180
                                   420 *                                                               * 79 *       32504190
00F2    12D8 42C0                  421 CD       BAL   CDADSDMD(MR6)        FETCH COMPARAND (P.10)                   32504200
00F3    CBF9 BDB2                  422          CDR   YD,RMDR,IRD,E        EXECUTED INSTRUCTION                     32504210
                                   423 *                                                               * 7A *       32504220
00F4    12D8 42C0                  424 AD       BAL   CDADSDMD(MR6)        FETCH ADDEND (P.10)                      32504230
00F5    CBF9 CDB2                  425          ADR   YD,RMDR,IRD,E        EXECUTED INSTRUCTION               R02   32504240
                                   426 *                                                               * 7B *       32504250
00F6    12D8 42C0                  427 SD       BAL   CDADSDMD(MR6)        FETCH SUBTRAHEND (P.10)                  32504260
00F7    CBF9 DDB2                  428          SDR   YD,RMDR,IRD,E        EXECUTED INSTRUCTION               R02   32504270
                                   429 *                                                               * 7C *       32504280
00F8    12D8 42C0                  430 MD       BAL   CDADSDMD(MR6)        FETCH MULTIPLIER (P.10)                  32504290
00F9    CBF9 EDB2                  431          MDR   YD,RMDR,IRD,E        EXECUTED INSTRUCTION               R02   32504300
                                   432 *                                                               * 7D *       32504310
00FA    12D8 42C0                  433 DD       BAL   CDADSDMD(MR6)        FETCH DIVISOR (P.10)                     32504320
00FB    CBF9 FDB2                  434          DDR   YD,RMDR,IRD,E        EXECUTED INSTRUCTION               R02   32504330
                                   435 *                                                               * 7E *       32504340
00FC    13F9 8640                  436 STMD     BAL   STMD1(NULL)          (P.54)                                   32504350
                                   437 *                                                                            32504360
00FD    0000 0000                  438          DC    FREEWORD                                                      32504370
                                   439 *                                                               * 7F *       32504380
00FE    13F9 8900                  440 LMD      BAL   LMD1(NULL)           (P.54)                                   32504390
00FF    CB7F 1C9F                  441 STE1     RRE   WMDR,YD,DW4          STORE SPFP DATA                    R02   32504400
```

ROM SEGMENT 4 - OPCODES 80:9F

```
0100                      ORG   '100'                                             32504420
0100   2BFF 1F92   EXIT10   L     NULL,NULL,IRD        EXIT.                     R02  32504430
0101   FFFD FFFF   BIT140   DC    'FFFDFFFF'           CONSTANT                       32504440
                   *                                                              32504450
0102   4B3F EC42   EXBR1    EXB   YD,YS,IR             SWAP LOW BYTES                 32504460
0103   2B39 7810            O     YD,YD,MR0,D          RESTORE R1 B00:15, EXIT.       32504470
                   *                                                  * 82 *      32504480
0104   2B9A 1D80   STDE     A     MAR,YX,RMDR          CALCULATE ADDRESS         R02  32504490
0105   13F9 9780            BAL   STDE1(NULL)          (P.56)                    R02  32504500

0106   0000 0000            DC    FREEWORD             .                         R02  32504520
0107   0000 0000            DC    FREEWORD             .                         R02  32504530
                   *                                   .                         R02  32504540
                   *                                                  * 84 *      32504550
0108   2B9A 1D8F   LED      A     MAR,YX,RMDR,DR4      CALCULATE ADDRESS         R02  32504560
0109   CBF9 8D8E            LW    YD,RMDR,I4DR4        LOAD HIGH HALF            R02  32504570
010A   CBF9 2DB2            LE    YD,RMDR,IRD,E        LOW HALF, ROUNDED.        R02  32504580
                   *                                                              32504590
010B   2B9A 1D8F   CDADSDMD A     MAR,YX,RMDR,DR4      FETCH HIGH HALF           R02  32504600
010C   CBFF 8D8E            LW    NULL,RMDR,I4DR4      LOAD MS 32 BITS, FETCH LS 32   32504610
010D   0BF8 0B00            EXL   (MR6)(NULL)          PERFORM FUNCTION, GET FLAGS    32504620
                   *                                                  * 87 *      32504630
010E   2B9A 1D8F   LDE      A     MAR,YX,RMDR,DR4      GET FLOATING DATA              32504640
010F   C3F9 8D97            LWX   YD,RMDR,LDE1         LOAD LOW HALF             R02  32504650
                   *                                                  * 88 *      32504660
0110   235F 1D1A   BRK      LX    CLOC,ILOC,BRK1       'BACK UP' LOC                  32504670

0000 0111          FLR1     EQU   *                                                   32504690
0111   2A3F 1C02   FLDR1    L     MR1,YS,IR            GET DATA TO FLOAT              32504700
0112   17E4 4540            BALNL FLR2(NULL)           BRANCH: POSITIVE               32504710
0113   D7FF 8021            LWI   NULL,CONSTCE,I       LOAD 'CE000000'                32504720
0114   223F 0896            SX    MR1,NULL,MR1,FLR3    COMPLEMENT DATA                32504730
0115   D7FF 8011   FLR2     LWI   NULL,CONST4E,I       LOAD '4E000000'                32504740
0116   0BF8 0B00   FLR3     EXL   (MR6)(NULL)          LOAD VALUE, EXIT.              32504750

0117   CBF9 AFB2   LDE1     LD    YD,NULL,IRD,E        FOLLOWED BY TRAILING ZEROS; EXIT.  32504770
                   *                                                  * 8C *      32504780
0118   12D8 8E00   RXRX     BAL   IIPCHECK(MR6)        CHECK IF IN PROGRESS (P.21)  R02  32504790
0119   13F9 A180            BAL   RXRX1(NULL)          (P.58)                    R02  32504800

011A   2B9F 1D0B   BRK1     L     MAR,ILOC,DR1         FETCH OPCODE AS DATA           32504820
011B   321F 1088            LI    MR0,'88'                                            32504830
011C   2BF0 6D80            X     NULL,MR0,RMDR        JUST A GLITCH ?                32504840
011D   17E0 8240            BALNZ ILEGAL(NULL)         BRANCH: YES (UNLIKELY)(P.18)   32504850
011E   17FC C000            BALD  CONSER(NULL)         BREAKPOINT (P.30)              32504860

011F   0000 0000            DC    FREEWORD             .                         R02  32504880
```

ROM SEGMENT 4 - OPCODES 80:9F

```
                   *                                                    * 90 *
0120   2B39 8EF2   SRHLS     SRHL  YD,YD,YSI,IRD,E
0121   0000 0F01   C0F01     DC    '00000F01'          CONSTANT
                   *                                                    * 91 *
0122   2B39 9EF2   SLHLS     SLHL  YD,YD,YSI,IRD,E
0123   0000 A001   CA001     DC    '0000A001'          CONSTANT
                   *                                                    * 92 *
0124   2219 6C3E   STBR      XX    MR0,YD,YS,STBR1     GET LOGICAL DIFFERENCE
0125   0000 0000             DC    FREEWORD
                   *                                                    * 93 *
0126   4B3F 5C52   LBR       LBR   YD,YS,IRD

0127   13F8 9200   EPSR1     BAL   QTEST(NULL)         TEST QUEUE SERVICE BIT (P.22)
                   *                                                    * 94 *
0128   3619 5015   EXBR      NI    MR0,YD,BI00.15,I    SAVE MS 16 BITS
0129   13F8 4080             BAL   EXBR1(NULL)         (P.10)
                   *                                                    * 95 *
012A   2B3D 1F91   EPSR      A     YD,PSW,NULL,@LOC    PSW TO R1                     R02
012B   23BF 1C27             LX    PSW,YS,EPSR1        LOAD NEW PSW FROM YS

                   * FOLLOWING CODE USED BY ROUTINE 'STBP1'
0000 012C          STBP.ZIP EQU    *                   'FAST EXIT' FOR ZERO
012C   F17F 112D             LI    M7R11,STBP.Z1       INTERRUPT RETURN
012D   16DD FC80   STBP.Z1   BALD  STBPSTOR(MR6)       STORE DATA BYTE (P.73)
012E   12DC 4B50             BALA  STBP.Z1(MR6),D      LOOP; ALLOW INTERRUPT.
012F   0000 0000             DC    FREEWORD
                   *                                                    * 98 *
0130   4BF9 DC32   WHR       WHA   NULL,YD,YS,IRD,E
0131   0000 0000             DC    FREEWORD
                   *                                                    * 99 *
0132   4B19 CFB2   RHR       RHA   YS,YD,NULL,IRD,E
0133   0000 0000             DC    FREEWORD
                   *                                                    * 9A *
0134   4BF9 9C72   WDR       WDRA  NULL,YD,YS,IRD,E
0135   221F 1DB7   OC1       LX    MR0,RMDR,OCR1       MR0 = COMMAND BYTE
                   *                                                    * 9B *
0136   4B19 8FF2   RDR       RDRA  YS,YD,NULL,IRD,E
                   *
0137   323F 100A   OCR1      LI    MR1,10              SHIFT COUNT FOR DELAY
0138   4BF9 B860             OCRA  NULL,YD,MR0,E       SEND OUTPUT COMMAND
0139   2BFF 8892             SRL   NULL,NULL,MR1,IRD   DELAY ABOUT 1 USEC
                   *                                                    * 9D *
013A   4B19 AFF2   SSR       SSRA  YS,YD,NULL,IRD,E
013B   3FFE 0000   BI02.14   DC    '3FFE0000'          CONSTANT
                   *                                                    * 9E *
013C   221F 1C37   OCR       LX    MR0,YS,OCR1         MR0 HAS COMMAND BYTE
013D   0002 0000   BIT14     DC    '00020000'          CONSTANT
                   *
013E   3210 5F00   STBR1     NI    MR0,MR0,'F00'       DROP LS BYTE FROM DIFFERENCE
013F   2B10 6C92             X     YS,MR0,YD,IRD       STORE YD B24:31 IN YS 24:31, EXIT.
```

ROM SEGMENT 5 - OPCODES A0:BF

```
0140                            ORG    '140'                                            32505410
                           * COMMON R14/R15 INTERRUPT PSW FETCH/SWAP ROUTINE                  32505420
                           *                                                            32505430
0000 0140                  COMSWAP  EQU    *                     COMMON PSW SWAP ROUTINE          32505440
0140   2A9D 1F8C                    A      MR4,PSW,NULL,PR4      MR4 = OLD PSW; FETCH NEW.        32505450
0141   2A7F 1D15                    L      MR3,ILOC,I4           MR3 = A(FAULTED INSTRUCTION)     32505460
0142   2ABF 1D8C                    L      MR5,RMDR,PR4          FETCH NEW LOC                    32505470
0143   2B5F 1D80                    L      CLOC,RMDR             NEW LOC -                        32505480
0144   2BBF 1A91                    L      PSW,MR5,@LOC          SELECT NEW PSW, UPDATE ILOC.     32505490
0145   29DF 1A00                    L      R14,MR4               OLD PSW TO R14                   32505500
0146   29FF 1980                    L      R15,MR3               OLD LOC TO R15                   32505510
0147   03F8 0B00                    BAL    (MR6)(NULL)           RETURN TO CALLER                 32505520
                           *                                                  * A4 *     32505530
0148   2A1F 1F00           LEDR     L      MR0,YDI               REMEMBER R1 SELECT               32505540
0149   C29F 9C11                    RRDX   MR4,YS,LEDR1          FETCH HIGH HALF DPFP DATA    R02 32505550
                           *                                                  * A5 *     32505560
014A   2A7F 1C00           LEGR     L      MR3,YS                GET GENERAL REGISTER         R02 32505570
014B   CBF9 29B2                    LE     YD,MR3,IRD,E          COPY TO SPFP.                R02 32505580
                           *                                                  * A6 *     32505590
014C   2A1F 1F00           LDGR     L      MR0,YDI               REMEMBER R1 SELECT               32505600
014D   227F 1C16                    LX     MR3,YS,LDGR1          LOAD HIGH HALF DPFP DATA     R02 32505610
                           *                                                  * A7 *     32505620
014E   CA7F 1C00           LDER     RRE    MR3,YS                GET SPFP VALUE               R02 32505630
014F   CBF9 8980                    LW     YD,MR3                LOAD                         R02 32505640
0150   CBF9 AFB2                    LD     YD,NULL,IRD,E         FOLLOWED BY TRAILING ZEROS.  R02 32505650

0151   2BDF 3E80           LEDR1    AINC   YDI,NULL,YSI          SELECT R2+1                  R02 32505670
0152   CA7F 9C80                    RRD    MR3,YD                READ LS HALF, DPFP DATA      R02 32505680
0153   2BDF 1800                    L      YDI,MR0               RESELECT R1                  R02 32505690
0154   CBF9 8A00                    LW     YD,MR4                HIGH HALF                    R02 32505700
0155   CBF9 29B2                    LE     YD,MR3,IRD,E          FOLLOWED BY LOW HALF, ROUNDED.  R02 32505710

0156   CBF9 8980           LDGR1    LW     YD,MR3                MS HALF DPFP DATA            R02 32505730
0157   2BDF 3E80                    AINC   YDI,NULL,YSI          DELECT R2+1                  R02 32505740
0158   2A7F 1C80                    L      MR3,YD                .                            R02 32505750
0159   2BDF 1800                    L      YDI,MR0               RESELECT R1                  R02 32505760
015A   CBF9 A9B2                    LD     YD,MR3,IRD,E          LS HALF DPFP DATA. EXIT.     R02 32505770

015B   3239 50FF           CLB1     NI     MR1,YD,'0FF'          ISOLATE FIRST OPERAND BYTE       32505790
015C   2BF1 0DB2                    S      NULL,MR1,RMDR,IRD,E SUBTRACT TO COMPARE            32505800

015D   361F 1017           LHL1     LI     MR0,BI16.31,I         MR0 = '0000FFFF'             R02 32505820
015E   2B30 5DB2                    N      YD,MR0,RMDR,IRD,E     LOAD LOW HALF, SET CC, EXIT. R02 32505830

015F   323F 1FF2           LME1     LI     MR1,'FF2'             MR1 = 'FFFFFFF2'                 32505850
0160   23FF 1FA2                    LX     NULL,NULL,LME3        FIRST FETCH ONGOING NOW -    R02 32505860
0161   2BFF 1F8F           LME2     L      NULL,NULL,DR4         FETCH FIRST WORD             R02 32505870
0162   CBF9 2DD5           LME3     LE     YD,RMDR,K,I4          LOAD REGISTER, INCREMENT MAR.    32505880
0163   23D1 1F61                    AX     YDI,MR1,YDI,LME2,C    LOOP THROUGH R15, THEN           32505890
0164   2BFF 1F92           EXIT12   L      NULL,NULL,IRD         EXIT.                            32505900
```

ROM SEGMENT 5 - OPCODES A0:BF

```
0165   323F 1FF2            593  LME@     LI    MR1,'FF2'          MR1 = 'FFFFFFF2'                       32505920
0166   03F0 0B0F            594  LME@1    BALC  (MR6)(NULL),DR4    RETURN IF DONE                         32505930
0167   CBF9 2DD5            595           LE    YD,RMDR,K,I4       LOAD REGISTER                          32505940
0168   23D1 1F26            596           AX    YDI,MR1,YDI,LME@1  LOOP THROUGH R15                       32505950

                            598  * ****************************************************                   32505970
                            599  *                                                                    32505980
                            600  *        SUPERVISOR CALL (SVC) INTERRUPT                             32505990
                            601  *                                                                    32506000
                            602  * ****************************************************                   32506010

0169   2AFF 1E00            604  SVC1     L     MR7,MAR            SAVE ADDRESS IN MR7                    32506030
016A   2A3F 1F05            605           L     MR1,YDI,RFAULT     COLLECT R1 FIELD, RESET RX FLOPS       32506040
016B   2A31 1880            606           A     MR1,MR1,MR1        2X R1 FIELD PLUS X'9C'                 32506050
016C   3391 109C            607           AI    MAR,MR1,'9C'       IS HALFWORD ADRS                       32506060
016D   2A3C 1F86            608           A     MR1,CLOC,NULL,PR2  OF SVC NEW LOC; SAVE OLOC.             32506070
016E   367F 1017            609           LI    MR3,BI16.31,I      MR3 = '0000FFFE'                       32506080
016F   339F 1098            610           LI    MAR,'98'           MAR = A(SVC NEW PSW)                   32506090
0170   2A73 5D8C            611           N     MR3,MR3,RMDR,PR4   MR3 = NEW LOC                      R02 32506100
0171   13F8 9500            612           BAL   COMSWAP2(NULL)     TAKE PSW SWAP, EXIT. (P.22)            32506110

                            614  * ************************************************************           32506130
                            615  *                                                                    32506140
                            616  *        MEMORY ADDRESS TRANSLATOR INTERRUPT                         32506150
                            617  *                                                                    32506160
                            618  * ************************************************************           32506170

0000 0172                   620  MATINT   EQU   *                  (RX FLOPS RESET IN 'FAULT' (P.19))     32506190
0172   339F 1090            621           LI    MAR,'90'           A(MAT INTPT NEW PSW)                   32506200
0173   12D8 5000            622           BAL   COMSWAP(MR6)       GO SWAP PSW'S (P.12)                   32506210
0174   31B2 500F            623           NI    R13,MR2,'0F'       R13 = FAULT CODE, FORM 000000XX        32506220
0175   299F 1891            624           L     R12,MR1,@LOC       R12 = PGM ADRS GENERATING FAULT        32506230
0176   33F0 048B            625           SI    NULL,MR0,LM1       DID IT HAPPEN ON 'LM' INSTRUCTION      32506240
0177   13F0 9000            626           BALC  TWAIT(NULL)        BRANCH: NO (P.22)                      32506250
0178   33F0 04A1            627           SI    NULL,MR0,LMTABE    FINAL CHECK -                          32506260
0179   17F0 9000            628           BALNC TWAIT(NULL)        BRANCH: NOT LM FAULT (P.22)            32506270
017A   297F 1B80            629           L     R11,MR7            R11 = A(LM BLOCK START)                32506280
017B   13FC 9010            630           BALA  TWAIT(NULL),D      GO TEST WAIT BIT. (P.22)               32506290

017C                        632           IFP   '180'-*            .                                  R02 32506310
017C                        633           DO    '180'-*            .                                  R02 32506320
017C   0000 0000            634           DC    FREEWORD           .                                  R02 32506330
017D   0000 0000            634           DC    FREEWORD           .                                  R02 32506330
017E   0000 0000            634           DC    FREEWORD           .                                  R02 32506330
017F   0000 0000            634           DC    FREEWORD           .                                  R02 32506330
                            635           ENDC                     .                                  R02 32506340
```

ROM SEGMENT 6 - OPCODES C0:DF

```
0180                    637          ORG   '180'                                                 32506360

                        639  *                                                      * C0 *      32506380
0180   13F9 32C0        640  BXH     BAL   BXH1(NULL)           (P.43)                          32506390
0181   0000 0000        641          DC    FREEWORD                                             32506400
                        642  *                                                      * C1 *      32506410
0182   13F9 3040        643  BXLE    BAL   BXLE1(NULL)          (P.43)                          32506420
0183   0000 0800        644  BIT20   DC    '00000800'           CONSTANT                        32506430
                        645  *                                                      * C2 *      32506440
0184   2B9A 1D8F        646  LPSW    A     MAR,YX,RMDR,DR4      CALCULATE ADDRESS          R02  32506450
0185   13F8 9140        647          BAL   LPSW1(NULL)          (P.22)                          32506460
                        648  *                                                      * C3 *      32506470
0186   2A1A 1D82        649  THI     A     MR0,YX,RMDR,IR                                       32506480
0187   2BF9 5830        650          N     NULL,YD,MR0,D,E                                      32506490
                        651  *                                                      * C4 *      32506500
0188   2A1A 1D82        652  NHI     A     MR0,YX,RMDR,IR                                       32506510
0189   2B39 5830        653          N     YD,YD,MR0,D,E                                        32506520
                        654  *                                                      * C5 *      32506530
018A   2A1A 1D82        655  CLHI    A     MR0,YX,RMDR,IR                                       32506540
018B   2BF9 0830        656          S     NULL,YD,MR0,D,E                                      32506550
                        657  *                                                      * C6 *      32506560
018C   2A1A 1D82        658  OHI     A     MR0,YX,RMDR,IR                                       32506570
018D   2B39 7830        659          O     YD,YD,MR0,D,E                                        32506580
                        660  *                                                      * C7 *      32506590
018E   2A1A 1D82        661  XHI     A     MR0,YX,RMDR,IR                                       32506600
018F   2B39 6830        662          X     YD,YD,MR0,D,E                                        32506610
                        663  *                                                      * C8 *      32506620
0190   2B3A 1DB2        664  LHI     A     YD,YX,RMDR,IRD,E     YD HAS RESULT. EXIT, CC SET.    32506630
0191   0000 0000        665          DC    FREEWORD                                             32506640
                        666  *                                                      * C9 *      32506650
0192   2A1A 1D80        667  CHI     A     MR0,YX,RMDR                                          32506660
0193   13F8 0C40        668          BAL   C1RI(NULL)           (P.3)                           32506670
                        669  *                                                      * CA *      32506680
0194   2A1A 1D82        670  AHI     A     MR0,YX,RMDR,IR                                       32506690
0195   2B39 1830        671          A     YD,YD,MR0,D,E                                        32506700
                        672  *                                                      * CB *      32506710
0196   2A1A 1D82        673  SHI     A     MR0,YX,RMDR,IR                                       32506720
0197   2B39 0830        674          S     YD,YD,MR0,D,E                                        32506730
                        675  *                                                      * CC *      32506740
0198   2A1A 1D82        676  SRHL    A     MR0,YX,RMDR,IR                                       32506750
0199   2B39 8870        677          SRHL  YD,YD,MR0,D,E                                        32506760
                        678  *                                                      * CD *      32506770
019A   2A1A 1D82        679  SLHL    A     MR0,YX,RMDR,IR                                       32506780
019B   2B39 9870        680          SLHL  YD,YD,MR0,D,E                                        32506790
                        681  *                                                      * CE *      32506800
019C   2A1A 1D82        682  SRHA    A     MR0,YX,RMDR,IR                                       32506810
019D   2B39 C870        683          SRHA  YD,YD,MR0,D,E                                        32506820
                        684  *                                                      * CF *      32506830
019E   2A1A 1D82        685  SLHA    A     MR0,YX,RMDR,IR                                       32506840
019F   2B39 D870        686          SLHA  YD,YD,MR0,D,E                                        32506850
```

ROM SEGMENT 6 - OPCODES C0:DF

```
                    *                                                     * D0 *       32506870
01A0   13F9 2840    STM      BAL    STM1(NULL)           STORE (P.42)          R02    32506880
01A1   0000 0000             DC     FREEWORD             .                     R02    32506890
                    *                                                     * D1 *       32506900
01A2   13F9 22C0    LM       BAL    LM1(NULL)            LOAD (P.42)           R02    32506910
01A3   0000 0000             DC     FREEWORD             .                     R02    32506920
                    *                                                     * D2 *       32506930
01A4   237F 1CB7    STB      LX     WMDR,YD,STB1         DATA TO WMDR                 32506940
01A5   0000 0000             DC     FREEWORD                                          32506950
                    *                                                     * D3 *       32506960
01A6   2B9A 1D8B    LB       A      MAR,YX,RMDR,DR1      FETCH BYTE                   32506970
01A7   2B3F 1D92             L      YD,RMDR,IRD          LOAD DATA TO YD, EXIT.       32506980
                    *                                                     * D4 *       32506990
01A8   2B9A 1D8B    CLB      A      MAR,YX,RMDR,DR1      FETCH BYTE                   32507000
01A9   13F8 56C0             BAL    CLB1(NULL)           (P.12)                       32507010
                    *                                                     * D5 *       32507020
01AA   2B9A 1D80    AL       A      MAR,YX,RMDR          CALCULATE ADDRESS            32507030
01AB   13F9 0840             BAL    AL1(NULL)            (P.38)                       32507040

01AC   0000 0000             DC     FREEWORD             .                     R02    32507060
01AD   0000 0000             DC     FREEWORD             .                     R02    32507070

01AE   4B79 CFB7    RH1      RHA    WMDR,YD,NULL,DW2,E   READ HALFWORD, STORE  R02    32507090
01AF   2BFF 1F92             L      NULL,NULL,IRD        EXIT.                 R02    32507100
                    *                                                     * D8 *       32507110
01B0   2B9A 1D87    WH       A      MAR,YX,RMDR,DR2      FETCH HALFWORD               32507120
01B1   4BF9 DDB2             WHA    NULL,YD,RMDR,IRD,E                                32507130
                    *                                                     * D9 *       32507140
01B2   239A 1DAE    RH       AX     MAR,YX,RMDR,RH1      CALCULATE ADDRESS     R02    32507150
01B3   0000 0000             DC     FREEWORD             .                     R02    32507160
                    *                                                     * DA *       32507170
01B4   2B9A 1D8B    WD       A      MAR,YX,RMDR,DR1      FETCH BYTE                   32507180
01B5   4BF9 9DF2             WDRA   NULL,YD,RMDR,IRD,E   ADDRESS, WRITE DATA, EXIT, NEW CC.  32507190
                    *                                                     * DB *       32507200
01B6   4B79 8FE0    RD       RDRA   WMDR,YD,NULL,E       READ DATA, SET CC            32507210
01B7   2B9A 1D9B    STB1     A      MAR,YX,RMDR,DW1      STORE BYTE                   32507220
01B8   2BFF 1F92    EXIT15   L      NULL,NULL,IRD        EXIT.                        32507230

01B9   0000 0000             DC     FREEWORD             .                     R02    32507250
                    *                                                     * DD *       32507260
01BA   4B79 AFE0    SS       SSRA   WMDR,YD,NULL,E       SENSE STATUS                 32507270
01BB   13F8 6DC0             BAL    STB1(NULL)           GO STORE STATUS BYTE         32507280
                    *                                                     * DE *       32507290
01BC   2B9A 1D8B    OC       A      MAR,YX,RMDR,DR1      FETCH COMMAND BYTE           32507300
01BD   13F8 4D40             BAL    OC1(NULL)            GO SEND IT. (P.11)           32507310
                    *                                                     * DF *       32507320
01BE   2AFF 1F00    PSF      L      MR7,YDI              SAVE FUNCTION SPEC           32507330
01BF   13F8 F680             BAL    PSF1(NULL)           AND GO DECODE. (P.36)        32507340
```

ROM SEGMENT 7 - OPCODES E0:FF

```
01C0                       ORG   '1C0'            .                          R02  32507360
                *                                                  * E0 *        32507370
01C0  2B9A 1D84      TS    A     MAR,YX,RMDR,RAS  CALCULATE ADRS, READ-AND-SET                32507380
01C1  2BFF 1DB2            L     NULL,RMDR,IRD,E  NEGATIVE IF ALREADY SET.                    32507390
                *                                                  * E1 *        32507400
01C2  2B9A 1D80      SVC   A     MAR,YX,RMDR      CALCULATE EFFECTIVE ADDRESS                 32507410
01C3  13F8 5A40            BAL   SVC1(NULL)       (P.13)                                      32507420
                *                                                  * E2 *        32507430
01C4  2A3A 1D91      SINT  A     DEV,YX,RMDR,@LOC DEV = I2+(X2); ILOC = CLOC.  R02  32507440
01C5  13F8 A1C0            BAL   SINT1(NULL)      (P.25)                       R02  32507450
                *                                                  * E3 *        32507460
01C6  2B9A 1D87      SCP   A     MAR,YX,RMDR,DR2  CALCULATE ADDRESS OF CCW   R02    32507470
01C7  13F9 4900            BAL   SCP1(NULL)       (P.46)                              32507480

01C8  0000 0000            DC    FREEWORD         .                            R02  32507500
01C9  0000 0000            DC    FREEWORD         .                            R02  32507510
                *                                                  * E5 *        32507520
01CA  2B9A 1D80      BDCS  A     MAR,YX,RMDR      CALCULATE ADDRESS                           32507530
01CB  22DF 1E11            AX    MR6,NULL,MAR,GO.BY.6 .                                       32507540
                *                                                  * E6 *        32507550
01CC  2B9A 1D80      LA    A     MAR,YX,RMDR      CALCULATE ADDRESS                           32507560
01CD  233F 1E25            LX    YD,MAR,EXIT17    PUT IN YD. CANNOT DO IR HERE (P.17)         32507570
                *                                                  * E7 *        32507580
01CE  2B9A 1D8F      TLATE A     MAR,YX,RMDR,DR4  FETCH TABLE ADDRESS                         32507590
01CF  13F9 36C0            BAL   TLATE1(NULL)     (P.44)                                      32507600
                *                                                  * E8 *        32507610
01D0  13FD 7990      CCS   BALA  CCS1(NULL),D     (P.52)                                      32507620
01D1  07FC 0B00      GO.BY.6 BALD (MR6)(NULL)     BRANCH, DISARM INTERRUPTS.                  32507630
                *                                                  * E9 *        32507640
01D2  32DF 1800      ECS   LI    MR6,'800'        SELECT FIRST WCS MODULE @ '800'             32507650
01D3  22D6 7F11            OX    MR6,MR6,YDI,GO.BY.6 COMPUTE ECS VECTOR ADDRESS               32507660
                *                                                  * EA *        32507670
01D4  2A1A 1D82      RRL   A     MR0,YX,RMDR,IR                                               32507680
01D5  2B39 A830            RRL   YD,YD,MR0,D,E                                                32507690
                *                                                  * EB *        32507700
01D6  2A1A 1D82      RLL   A     MR0,YX,RMDR,IR                                               32507710
01D7  2B39 B830            RLL   YD,YD,MR0,D,E                                                32507720
                *                                                  * EC *        32507730
01D8  2A1A 1D82      SRL   A     MR0,YX,RMDR,IR                                               32507740
01D9  2B39 8830            SRL   YD,YD,MR0,D,E                                                32507750
                *                                                  * ED *        32507760
01DA  2A1A 1D82      SLL   A     MR0,YX,RMDR,IR                                               32507770
01DB  2B39 9830            SLL   YD,YD,MR0,D,E                                                32507780
                *                                                  * EE *        32507790
01DC  2A1A 1D82      SRA   A     MR0,YX,RMDR,IR                                               32507800
01DD  2B39 C830            SRA   YD,YD,MR0,D,E                                                32507810
                *                                                  * EF *        32507820
01DE  2A1A 1D82      SLA   A     MR0,YX,RMDR,IR                                               32507830
01DF  2B39 D830            SLA   YD,YD,MR0,D,E                                                32507840
```

ROM SEGMENT 7 - OPCODES E0:FF

```
01E0    2A1F 1E05                787  STME1     L     MR0,MAR,RFAULT      GET ADDRESS, RESET RX FLOPS          32507860
01E1    3390 0004                788            SI    MAR,MR0,4           PRE-DECREMENT MAR                    32507870
01E2    323F 000E                789            SI    MR1,NULL,14         MR1 = 'FFFFFFF2'                     32507880
01E3    CB7F 1C9D                790  STME2     RRE   WMDR,YD,I4DW4       FETCH, STORE                         32507890
01E4    23D1 1F63                791            AX    YDI,MR1,YDI,STME2,C LOOP THROUGH REGISTER E              32507900
01E5    2BFF 1F92                792  EXIT17    L     NULL,NULL,IRD       EXIT.                                32507910
                                 793  *                                                        * F3 *      32507920
01E6    2A1A 1D82                794  TI        A     MR0,YX,RMDR,IR                                           32507930
01E7    2BF9 5830                795            N     NULL,YD,MR0,D,E                                          32507940
                                 796  *                                                        * F4 *      32507950
01E8    2A1A 1D82                797  NI        A     MR0,YX,RMDR,IR                                           32507960
01E9    2B39 5830                798            N     YD,YD,MR0,D,E                                            32507970
                                 799  *                                                        * F5 *      32507980
01EA    2A1A 1D82                800  CLI       A     MR0,YX,RMDR,IR                                           32507990
01EB    2BF9 0830                801            S     NULL,YD,MR0,D,E                                          32508000
                                 802  *                                                        * F6 *      32508010
01EC    2A1A 1D82                803  OI        A     MR0,YX,RMDR,IR                                           32508020
01ED    2B39 7830                804            O     YD,YD,MR0,D,E                                            32508030
                                 805  *                                                        * F7 *      32508040
01EE    2A1A 1D82                806  XI        A     MR0,YX,RMDR,IR                                           32508050
01EF    2B39 6830                807            X     YD,YD,MR0,D,E                                            32508060
                                 808  *                                                        * F8 *      32508070
01F0    2B3A 1DB2                809  LI        A     YD,YX,RMDR,IRD,E    YD HAS RESULT; EXIT, CC SET.         32508080
01F1    0000 0000                810            DC    FREEWORD                                                 32508090
                                 811  *                                                        * F9 *      32508100
01F2    2A1A 1D80                812  CI        A     MR0,YX,RMDR                                              32508110
01F3    13F8 0C40                813            BAL   C1RI(NULL)          (P.3)                                32508120
                                 814  *                                                        * FA *      32508130
01F4    2A1A 1D82                815  AI        A     MR0,YX,RMDR,IR                                           32508140
01F5    2B39 1830                816            A     YD,YD,MR0,D,E                                            32508150
                                 817  *                                                        * FB *      32508160
01F6    2A1A 1D82                818  SI        A     MR0,YX,RMDR,IR                                           32508170
01F7    2B39 0830                819            S     YD,YD,MR0,D,E                                            32508180

01F8    323F 000E                821  STME@     SI    MR1,NULL,14         MR1='FFFFFFF2'                       32508200
01F9    CB7F 1C9F                822  STME@1    RRE   WMDR,YD,DW4         STORE REGISTER                       32508210
01FA    2BFF 1F95                823            L     NULL,NULL,I4        INCREMENT MAR                        32508220
01FB    23D1 1F79                824            AX    YDI,MR1,YDI,STME@1,C LOOP THROUGH REGISTER E.            32508230
01FC    03F8 0B00                825            BAL   (MR6)(NULL)         RETURN TO CALLER                     32508240

01FD    235F 1D3F                827  RETOLOC   LX    CLOC,ILOC,RETOLOC1  'BACK UP' LOC                        32508260
01FE    17FC 8240                828  TRAPFF    BALD  ILEGAL(NULL)        HARDWARE TRAP FOR BAD DECODE (P.18)  32508270
01FF    13FC 9010                829  RETOLOC1  BALA  TWAIT(NULL),D       GO TEST WAIT BIT (P.22)              32508280
```

ROM SEGMENT 8 - INTERRUPT HANDLERS

```
0200                          ORG   '200'                                              32508300

                     * FOR THE INTERRUPT VECTORS BELOW, RAG IS LOADED WITH THE         32508320
                     * VECTOR ADDRESS. HOWEVER, RLC POINTS TO THE MICRO-INSTRUCTION    32508330
                     * BEING EXECUTED AT THE TIME OF THE INTERRUPT. THEREFORE, MRO     32508340
                     * IS LOADED WITH THE ADDRESS OF THE INSTRUCTION FOLLOWING THE     32508350
                     * INTERRUPTED ONE, IN CONTROL STORE, WHEN THE BAL(XXX)(MRO)      32508360
                     * IS EXECUTED.                                                     32508370
                     * ILOC IS ASSUMED TO BE THE VALID LOC FOR NEXT IRD.               32508380

0200  161C A140               BALD  IOINT3(MRO)       I/O INTERRUPT LEVEL 3 (P.25)         32508400
0201  161C A0C0               BALD  IOINT2(MRO)       I/O INTERRUPT LEVEL 2 (P.25)         32508410
0202  161C A040               BALD  IOINT1(MRO)       I/O INTERRUPT LEVEL 1 (P.25)         32508420
0203  161C A000               BALD  IOINT0(MRO)       I/O INTERRUPT LEVEL 0 (P.25)         32508430
0204  161C C000               BALD  CONSER(MRO)       CONSOLE ATTENTION     (P.30)         32508440
0205  161C 8440               BALD  FAULT.0(MRO)      MACHINE MALFUNCTION   (P.19)    R02  32508450
0206  161D 5800               BALD  PPFINT(MRO)       PRIMARY POWER FAIL    (P.48)         32508460
0207  161C 83C0               BALD  FAULT(MRO)        ACCESS/DATA/BOUND/FPP (P.19)    R02  32508470
0208  161C 8240               BALD  ILEGAL(MRO)       ILLEGAL INSTRUCTION                  32508480


                     * ****************************************************            32508500
                     *                                                                  32508510
                     *          ILLEGAL INSTRUCTION INTERRUPT                           32508520
                     *                                                                  32508530
                     * ****************************************************            32508540

0209  2BFF 1F85      ILEGAL   L     NULL,NULL,RFAULT  RESET RX FLOPS                       32508560
020A  339F 1030               LI    MAR,'30'          ADRS OF ILLEGAL INSTR. NEW PSW       32508570
020B  12D8 5000               BAL   COMSWAP(MR6)      EXCHANGE PSW'S (P.12)                32508580
020C  13FC 9050               BALA  TWAIT1(NULL),D    GO TEST WAIT BIT. (P.22)             32508590
```

ROM SEGMENT 8 - INTERRUPT HANDLERS

```
                    * *******************************************************                      32508610
                    *                                                                              32508620
                    *        MACHINE MALFUNCTION INTERRUPT                                         32508630
                    *                                                                              32508640
                    * *******************************************************                      32508650

                    *    '80000000'     POWER FAILURE                                              32508670
                    *    '40000000'     POWER RESTORE                                              32508680
                    *    '20000000'     MEMORY DATA ERROR, DATA FETCH                              32508690
                    *    '10000000'     MEMORY DATA ERROR, INSTRUCTION FETCH                       32508700
                    *    '08000000'     MEMORY DATA ERROR, AUTO-DRIVER CHANNEL                     32508710
                    *    '04000000'     NON-CONFIG'D MEMORY,  DATA FETCH                           32508720
                    *    '02000000'     NON-CONFIG'D MEMORY,  INSTRUCTION FETCH                    32508730
                    *    '01000000'     NON-CONFIG'D MEMORY,  AUTO-DRIVER CHANNEL                  32508740
                    *    '00000002'     SHARED MEMORY POWER FAIL                              R03  32508750
                    *    '00000001'     MODULE START TIME FAILURE                                  32508760

                    * MODULE START TIME FAILURE INTERRUPT                                           32508780
020D   227F 2FB0    STFAIL   SDECX MR3,NULL,NULL,MMFINT CODE = '00000001' (P.20)                   32508790

                    * EARLY POWER FAILURE INTERRUPT                                                32508810
020E   227F 1FB0    EPFINT   LX    MR3,NULL,MMFINT     CODE = '80000000' (P.20)                    32508820

                    * HERE ON ANY DATA/INSTRUCTION FETCH/DECODE/STORE ERROR.                       32508840
                    * OR MACHINE MALFUNCTION INTERRUPT                                             32508850

0000 020F           FAULT    EQU   *                   SORTS OUT INTERRUPTS                        32508870
020F   2A3F 1D85             L     MR1,RMDR,RFAULT     GET FAULT CODE AND ADDRESS                  32508880
0210   13E4 9AC0             BALL  FPPFAUL(NULL)       BRANCH: FPP INTERRUPT (P.23)           R02  32508890
0211   325F 1FFB    FAULT.0  LI    MR2,'FFB'           WILL NOT RESET MVF                     R02  32508900
0212   4ABF 7F85             SMCR  MR5,NULL,RFAULT     TEST MCR FAULTS                             32508910
0213   4BFF 7940             CMCR  NULL,MR2            RESET ALL BUT MVF (SMCR FLAGS)         R02  32508920
0214   13E4 8380             BALL  EPFINT(NULL)        BRANCH: EPF BIT SET                         32508930
0215   13F0 1800             BALC  STFAIL2(NULL)       BRANCH: STFAIL OR SMPF (P.5)           R03  32508940
0216   13F4 7F40             BALV  RETOLOC(NULL)       BRANCH: IGNORE NVMO INTERRUPT (P.17)        32508950
0217   3651 5049             NI    MR2,MR1,BI00.07,I   MR2 = FAULT CODE RETURNED                   32508960
0218   2A32 5880             X     MR1,MR2,MR1         MR1 = PROGRAM ADDRESS AT FAULT              32508970
0219   3252 B008             RLLI  MR2,MR2,8           FAULT CODE, FORM 000000XX                   32508980
021A   327F 1002             LI    MR3,2               BASE COUNT FOR SHIFTS                       32508990
021B   33F0 0298             SI    NULL,MR0,CHANEL     FAULT IN CHANEL CODE ?                      32509000
021C   13F0 8900             BALC  FAULT.2(NULL)       BRANCH: NOT INTERESTING.                    32509010
021D   33F0 02F6             SI    NULL,MR0,CHANEND    REALLY CHANEL ?                             32509020
021E   17F0 8800             BALNC FAULT.1(NULL)       BRANCH: NO.                                 32509030
021F   2273 19A4             AX    MR3,MR3,MR3,FAULT.2 CHANEL - SET SHIFTS = 4                     32509040
0220   33F0 02FE    FAULT.1  SI    NULL,MR0,MMFEND     FAULTED FAULT SWAP ?                        32509050
0221   13F0 BF80             BALC  HARDSTOP(NULL)      BRANCH: YES. STOP MACHINE (LOC='40')        32509060
                    *                                  DOUBLE FAULT IS NOT TOLERATED.(P.29)        32509070
0222   33F0 03D0             SI    NULL,MR0,CONSEND    FAULT IN CONSER CODE ?                      32509080
0223   13F0 C000             BALC  CONSER(NULL)        RETURN IF YES. (P.30)                       32509090
0000 0224           FAULT.2  EQU   *                                                               32509100
0224   329C 0002             SI    MR4,CLOC,2          CLOC ADVANCES ON FAULT BY 2            R02  32509110
0225   2BF1 6A00             X     NULL,MR1,MR4        FAULT ON INSTRUCTION FETCH ?           R02  32509120
0226   17E0 8A00             BALNZ FAULT.3(NULL)       BRANCH: NO. (P.20)                     R02  32509130
```

ROM SEGMENT 8 - INTERRUPT HANDLERS

```
0227     327F 1003                  915            LI     MR3,3                 ASSUME FAULT ON INSTRUCTION FETCH.   32509140
0228     33F2 0019                  916  FAULT.3   SI     NULL,MR2,'19'         WAS IT A MEMORY-ACCESS FAULT ?       32509150
0229     13F0 5C80                  917            BALC   MATINT(NULL)          BRANCH: YES, PURE & SIMPLE. (P.13)   32509160
022A     13E0 8B80                  918            BALZ   FAULT.4(NULL)         BRANCH: PARITY/ECC; SHIFT 2,3, OR 4  32509170
022B     33F2 001E                  919            SI     NULL,MR2,'1E'         ALIGNMENT FAULT ?                    32509180
022C     17F0 9D40                  920            BALNC  FORFAUL6(NULL)        BRANCH: ALIGNMENT FAULT (P.24)       32509190
022D     3273 1003                  921            AI     MR3,MR3,3             NON-CONFIG'D MEMORY. SHIFT 5,6, OR 7 32509200
0000 022E                           922  FAULT.4   EQU    *                                                          32509210
022E     339F 1044                  923            LI     MAR,'44'              ADDRESS DEDICATED LOCATION           32509220
022F     2B7F 189C                  924            L      WMDR,MR1,PW4          STORE (MAR) AT TIME OF FAULT         32509230

0000 0230                           926  MMFINT    EQU    *                     MALFUNCTION INTERRUPT SWAP           32509250
0230     339F 1040                  927            LI     MAR,'40'              A(MACHINE MALFUNCTION STATUS WORD)   32509260
0231     365F 105B                  928            LI     MR2,BIT00,I           MR2 = '80000000'                     32509270
0232     2B72 899C                  929            SRL    WMDR,MR2,MR3,PW4      '40-43' = MALFUNCTION STATUS WORD    32509280
0233     339F 1020                  930            LI     MAR,'20'              A(MMF OLD PSW SAVE AREA)             32509290
0234     2B7D 1F80                  931            A      WMDR,PSW,NULL         OLD PSW TO GO AT '20-23'             32509300
0235     37ED 5051                  932            NI     NULL,PSW,BIT18,I      MALFUNCTION SWAP ENABLED ?           32509310
0236     13E0 901C                  933            BALZ   TWAIT(NULL),PW4       BRANCH: NO (P.22)                    32509320
0237     13F8 BD40                  934            BAL    MMFINT2(NULL)         GO DO SWAP (P.29)                    32509330

                                    936  *         C P A   F A U L T   C O D E   S U M M A R Y                       32509350

                                    938  *          8X - FLOATING POINT FAULT             (ARITH)            R02  32509370
                                    939  *          00 - NO FAULTS                                                32509380
                                    940  *          10 - (NOT USED)                      (MAT)                    32509390
                                    941  *          11 - EXECUTE PROTECT VIOLATION       (MAT)                    32509400
                                    942  *          12 - WRITE   PROTECT VIOLATION       (MAT)                    32509410
                                    943  *          13 - READ    PROTECT VIOLATION       (MAT)                    32509420
                                    944  *          14 - ACCESS LEVEL    VIOLATION       (MAT)                    32509430
                                    945  *          15 - SEGMENT LIMIT   VIOLATION       (MAT)                    32509440
                                    946  *          16 - SEGMENT NOT PRESENT             (MAT)                    32509450
                                    947  *          17 - SHARED SEG TAB SIZE EXCEEDED    (MAT)                    32509460
                                    948  *          18 - PRIVATE SEG TAB SIZE EXCEEDED   (MAT)                    32509470
                                    949  *          19 - ECC/PARITY ERROR                (MMF)                    32509480
                                    950  *          1A - NON-CONFIGURED MEMORY           (MMF)                    32509490
                                    951  *          1B - (NOT USED)                      (MMF)                    32509500
                                    952  *          1C - (NOT USED)                      (MMF)                    32509510
                                    953  *          1D - (NOT USED)                      (MMF)                    32509520
                                    954  *          1E - FULLWORD ALIGNMENT FAULT        (ALIGN)                  32509530
                                    955  *          1F - HALFWORD ALIGNMENT FAULT        (ALIGN)                  32509540
```

ROM SEGMENT 8 - INTERRUPT HANDLERS

```
                 * THIS ROUTINE DETERMINES WHETHER TO RESUME AN INTERRUPTED
                 * STRING INSTRUCTION, OR TO BEGIN A NEW ONE. FOR THIS
                 * PROCESSOR, PSW BIT 14 IS SET DURING THE TIME AN INTERRUPTIBLE
                 * INSTRUCTION IS EXECUTING.

0238   37FD 513D   IIPCHECK NI    NULL,PSW,BIT14,I    INSTRUCTION IN PROGRESS ?
0239   03E0 0B00            BALZ  (MR6)(NULL)         BRANCH: NO. START IT.

0000 023A          IIPRESUM EQU   *                   RESUME INTERRUPTED INSTRUCTION
023A   EADF 1585            L     MR6,M7R11,RFAULT    GET RETURN ADDRESS FROM R11,
                 *                                    RESET RX FLOPS.
023B   E35F 153F            LX    CLOC,M7R10,WINDOW   RESTORE INCREMENTED LOC

                 * THIS ROUTINE ESTABLISHES THE CONTROL STORE ADDRESS AT WHICH
                 * AN INTERRUPTED STRING INSTRUCTION IS TO BE RESUMED. FOR THIS
                 * PROCESSOR, NO ADVANCE WARNING IS GIVEN OF A PENDING INTERRUPT.

0000 023C          SET.RTN  EQU   *                   SET INTERRUPT RETURN ADDRESS
023C   E97F 1B00            L     M7R11,MR6           LINK ADDRESS BECOMES RETURN ADDRESS
023D   37BD 713D            OI    PSW,PSW,BIT14,I     SET IIP BIT
023E   13FC 8FD0            BALA  WINDOW(NULL),D      SERVICE ANY INTERRUPT
023F   07FC 0B00   WINDOW   BALD  (MR6)(NULL)         RETURN TO CALLER.

                            ENDC
```

ROM SEGMENT 9 - INTERRUPT HANDLERS

```
0240                      ORG   '240'                                                       32509840

                  * ********************************************************               32509860
                  *                                                                         32509870
                  *         INTERRUPTIBLE WAIT LOOP                                         32509880
                  *                                                                         32509890
                  * ********************************************************               32509900

0240  4BFF 6FD1   TWAIT    LWFF  NULL,NULL,@LOC       RESET WAIT, UPDATE ILOC               32509920
0241  361D 5019   TWAIT1   NI    MR0,PSW,BIT16,I      TEST WAIT BIT                         32509930
0242  17E0 90D2            BALNZ WAIT(NULL),IRD       BRANCH IF SET, ELSE EXIT.             32509940
0243  4BFF 6840   WAIT     LWFF  NULL,MR0             SET WAIT INDICATOR                    32509950
0244  13FC 9110            BALA  *(NULL),D            WAIT FOR INTERRUPT.                   32509960

0245  2A7F 1D8E   LPSW1    L     MR3,RMDR,I4DR4       MR3 = NEW PSW                     R02 32509980
0246  2B5F 1D85            L     CLOC,RMDR,RFAULT     LOAD NEW PSW, RESET RX FLOPS      R02 32509990
0247  2BBF 1991   LPSW2    L     PSW,MR3,@LOC         LOAD NEW PSW, UPDATE ILOC         R03 32510000

                  * ********************************************************               32510020
                  *                                                                         32510030
                  *         TEST SYSTEM QUEUE FOR SERVICE INTERRUPT                         32510040
                  *                                                                         32510050
                  * ********************************************************               32510060

0248  339F 1080   QTEST    LI    MAR,'80'             MAR = A(SYSTEM QUEUE POINTER)     R02 32510080
0249  33FD 5200            NI    NULL,PSW,'200'       QUEUE SERVICE ENABLED ?               32510090
024A  13E0 904C            BALZ  TWAIT1(NULL),PR4     BRANCH: NO.                           32510100
024B  36DF 1017            LI    MR6,BI16.31,I        MR6 = '0000FFFF'                  R02 32510110
024C  2B9F 1D80            L     MAR,RMDR             MAR = A(QUEUE)                    R02 32510120
024D  2AFF 1E0C            L     MR7,MAR,PR4          SAVE IN MR7                       R02 32510130
024E  339F 108C            LI    MAR,'8C'             A(QUEUE SERVICE NEW PSW LOC)      R02 32510140
024F  2BF6 5D80            N     NULL,MR6,RMDR        LOOK AT 'NUMBER USED'             R02 32510150
0250  13E0 904C            BALZ  TWAIT1(NULL),PR4     BRANCH: QUEUE EMPTY.              R02 32510160

0251  2A3C 1F80   SYSQINT  A     MR1,CLOC,NULL        MR1 = CURRENT LOC                 R02 32510180
0252  339F 1088            LI    MAR,'88'             MAR = A(QUEUE SERVICE NEW PSW)        32510190
0253  2A7F 1D8C            L     MR3,RMDR,PR4         MR3 = NEW LOC                         32510200

                  * THIS CODE SHARED BY SVC, EPSR, LPSW, LPSWR, LDPS                        32510220
0000 0254         COMSWAP2 EQU   *                                                          32510230
0254  2A1D 1F80            A     MR0,PSW,NULL         MR0 = OLD PSW                     R02 32510240
0255  2BBF 1D80            L     PSW,RMDR             LOAD NEW PSW                          32510250
0256  2B5F 1980            L     CLOC,MR3             LOAD NEW LOC                          32510260
0257  29BF 1B80            L     R13,MR7              R13 = A(SYSTEM QUEUE), OR             32510270
                  *                                   SVC PARAM BLK ADDRESS                 32510280
0258  29DF 1811            L     R14,MR0,@LOC         R14 = OLD PSW                         32510290
0259  21FF 1881            LX    R15,MR1,TWAIT1       R15 = OLD LOC                         32510300

025A  0000 0000            DC    FREEWORD             .                                 R02 32510320
```

ROM SEGMENT 9 - INTERRUPT HANDLERS

```
                             1035  * ***********************************************************          32510340
                             1036  *                                                                      32510350
                             1037  *          ARITHMETIC FAULT INTERRUPT                                  32510360
                             1038  *                                                                      32510370
                             1039  * ***********************************************************          32510380

025B   2B3F 1900             1041  DFAULT    L     YD,MR2             RESTORE 64-BIT DIVIDEND            32510400
025C   2BDF 3F00             1042            AINC  YDI,NULL,YDI                                          32510410
025D   2B3F 1980             1043            L     YD,MR3                                                32510420
025E   23F4 2FE0             1044            SDECX NULL,MR4,NULL,AFAUL1,C BRANCH: QUOTIENT OVERFLOW      32510430


025F                         1046            ORG   '25F'              ALIGNS LINKS                       32510450

025F   1218 9980             1048  AFAUL0    BAL   AFAULT(MR0)        FIX POINT DIV-BY-0.                32510470
0260   1218 9980             1049  AFAUL1    BAL   AFAULT(MR0)        FIX POINT QUOTIENT O'FLOW          32510480
0261   1210 9980             1050  AFAUL2    BALC  AFAULT(MR0)        BRANCH: FLOAT POINT DIV-BY-0       32510490
0262   1200 9900             1051  AFAUL3    BALZ  UFAULT(MR0)        BRANCH: FLOAT POINT EXPONENT U'FLOW  32510500
0263   1218 9980             1052  AFAUL4    BAL   AFAULT(MR0)        BRANCH: FLOAT POINT EXPONENT O'FLOW  32510510

0264   37FD 50C1             1054  UFAULT    NI    NULL,PSW,BIT19,I   AFAULT ENABLED ?                   32510530
0265   17E0 9992             1055            BALNZ AFAULT(NULL),IRD   BRANCH: YES. ELSE, EXIT.           32510540

0266   2AFC 1F85             1057  AFAULT    A     MR7,CLOC,NULL,RFAULT MR7 = INCR'D LOC, RESET RX FLOPS  32510560
0267   339F 1048             1058            LI    MAR,'48'           MAR = A(ARITH FAULT NEW PSW)       32510570
0268   12D8 5000             1059            BAL   COMSWAP(MR6)       DO PSW SWAP (P.12)                 32510580
0269   31B0 5007             1060            NI    R13,MR0,'07'       R13 = FAULT CODE                   32510590
026A   219F 1B81             1061            LX    R12,MR7,TWAIT1     R12 = NEXT LOC (P.22)              32510600


026B   3251 B004             1063  FPPFAUL   RLLI  MR2,MR1,4          POSITION FORMAT INFORMATION    P02 32510620
026C   3252 5006             1064            NI    MR2,MR2,6          LENGTH 2, 4, OR 6 BYTES        P02 32510630
026D   2B5E 0900             1065            S     CLOC,ILOC,MR2      POINT TO START OF FPP INSTRUCTION  32510640
026E   2A5E 1F80             1066            A     MR2,ILOC,NULL      SAVE ILOC; ILOC GETS CLOC      P02 32510650
026F   2B5F 1911             1067            L     CLOC,MR2,@LOC      CLOC GETS 'NEXT LOC'           P02 32510660
                             1068  *                                  ILOC HAS FAULT LOC             P02 32510670
                             1069  *                                  NOTE - @LOC HAPPENS EARLY HERE. P02 32510680
0270   C3FF 0FA1             1070            RCCX  NULL,NULL,AFAUL2   COLLECT FLAGS, GO SORT FAULT.  P02 32510690
```

```
                          1072  * ***********************************************************             32510710
                          1073  *                                                                          32510720
                          1074  *         DATA FORMAT FAULT INTERRUPT                                      32510730
                          1075  *                                                                          32510740
                          1076  * ***********************************************************             32510750

0271                      1078           ORG   '271'                                                       32510770

0271  1218 9CC0           1080  FORFAUL2 BAL   IIPFAUL(MR0)       INV SIGN DIGIT, PACKED DATA      R01  32510790
0272  1218 9CC0           1081  FORFAUL3 BAL   IIPFAUL(MR0)       INV SIGN DIGIT, PACKED DATA      R01  32510800
0273  373D 5101           1082  IIPFAUL  NI    PSW,PSW,BIT140,I   ZERO IIP BIT                     R01  32510810
0274  13F8 9DC0           1083           BAL   FORFAULT(NULL)     .                                R01  32510820
0275  1200 9E00           1084  FORFAUL6 BALZ  ALGFAULT(MR0)      BRANCH: FULLWORD ALIGNMENT FAULT      32510830
0276  1218 9E00           1085  FORFAUL7 BAL   ALGFAULT(MR0)      HALFWORD ALIGNMENT FAULT.             32510840

0277  2A3F 1E05           1087  FORFAULT L     MR1,MAR,RFAULT     PROGRAM ADDRESS IN MAR AT FAULT       32510860
                          1088  *                                 RESET RX FLOPS.                       32510870
0278  339F 10C8           1089  ALGFAULT LI    MAR,'C8'           A(FORMAT FAULT NEW PSW)               32510880
0279  12D8 5000           1090           BAL   COMSWAP(MR6)       SWAP PSW'S (P.12)                     32510890
027A  3130 5007           1091           NI    R13,MR0,'007'      R13 = FAULT CODE                      32510900
027B  33FD 0006           1092           SI    NULL,R13,6         HARDWARE FAULT ?                      32510910
027C  13F0 9040           1093           BALC  TWAIT1(NULL)       BRANCH: NO (P.22) ELSE,               32510920
027D  219F 1881           1094           LX    R12,MR1,TWAIT1     R12 = ADDRESS CAUSING FAULT (P.22)    32510930

027E  0000 0000           1096           DC    FREEWORD           .                                R02  32510950
027F  0000 0000           1097           DC    FREEWORD           .                                R02  32510960
```

ROM SEGMENTS A, B - I/O INTERRUPT PROCESSOR

```
0280                          1099          ORG   '280'                                                   32510980

                              1101  * REGISTER ASSIGNMENTS FOR CHANNEL I/O                                32511000
                              1102  *                                                                     32511010
0000 0010                     1103  TEMP     EQU   '10'                                                   32511020
0000 0011                     1104  DEV      EQU   '11'                                                   32511030
0000 0012                     1105  LEVEL    EQU   '12'                                                   32511040
0000 0012                     1106  CCW      EQU   '12'                                                   32511050
0000 0013                     1107  DAT      EQU   '13'                                                   32511060
0000 0014                     1108  COUNT    EQU   '14'                                                   32511070
0000 0015                     1109  RETURN   EQU   '15'                                                   32511080


                              1111  * ***********************************************************         32511100
                              1112  *                                                                     32511110
                              1113  *        I/O INTERRUPT                                                32511120
                              1114  *                                                                     32511130
                              1115  * ***********************************************************         32511140

0280   225F 1F86              1117  IOINTO   LX    LEVEL,NULL,AUTOIO   SELECT REGISTER SET 0              32511160

0281   325F 1011              1119  IOINT1   LI    LEVEL,'11'          SELECT REGISTER SET 1              32511180
0282   423F 690B              1120           AKX   DEV,LEVEL,IOINTX    ACKNOWLEDGE INTERRUPT              32511190

0283   325F 1022              1122  IOINT2   LI    LEVEL,'22'          SELECT REGISTER SET 2              32511210
0284   423F 690B              1123           AKX   DEV,LEVEL,IOINTX    ACKNOWLEDGE INTERRUPT              32511220

0285   325F 1033              1125  IOINT3   LI    LEVEL,'33'          SELECT REGISTER SET 3              32511240
0286   423F 690B              1126  AUTOIO   AKX   DEV,LEVEL,IOINTX    ACKNOWLEDGE INTERRUPT              32511250


                              1128  * **************************************************************      32511270
                              1129  *                                                                     32511280
                              1130  *        SIMULATED (I/O) INTERRUPT                                    32511290
                              1131  *                                                                     32511300
                              1132  * **************************************************************      32511310

0287   3231 53FF              1134  SINT1    NI    DEV,DEV,'3FF'       FORCE DEVICE VALID                 32511330
0288   225F 0F4B              1135           SX    LEVEL,NULL,YDI,IOINTX,C BRANCH: LEVEL 0 REQUESTED.     32511340
0289   3259 9004              1136           SLLI  LEVEL,YD,4          SELECT SPECIFIED LEVEL             32511350
028A   3252 5030              1137           NI    LEVEL,LEVEL,'030'   FORCE VALID LEVEL                  32511360

028B   3271 10D0              1139  IOINTX   AI    DAT,DEV,'D0'        2X DEVICE NUMBER + 'D0'            32511380
028C   2A1D 1F85              1140           A     TEMP,PSW,NULL,RFAULT SAVE ENTRY PSW,                   32511390
                              1141  *                                  RESET RX FLOPS.                    32511400
028D   2B93 1886              1142           A     MAR,DAT,DEV,PR2     INDEXES SERVICE POINTER TABLE      32511410
                              1143  *                                  FETCH APPROPRIATE ENTRY            32511420
028E   37B2 704D              1144           OI    PSW,LEVEL,BI1820,I  SET PSW BITS 18 & 20 AND           32511430
                              1145  *                                  SELECT REGISTER SET                32511440
028F   281F 1800              1146           L     R0,TEMP             REG 0 = PSW                        32511450
0290   283E 1F80              1147           A     R1,ILOC,NULL        REG 1 = ADJUSTED LOC               32511460
                              1148  *                                  ASSUMES ILOC VALID NEXT INSTRUCTION 32511470
```

ROM SEGMENTS A, B - I/O INTERRUPT PROCESSOR

```
0291   285F 1880                  L     R2,DEV             REG 2 = DEVICE NUMBER              32511480
0292   4871 AFE0                  SSRA  R3,DEV,NULL,E      REG 3 = DEVICE STATUS, SET CC      32511490
0293   2A7F 1D80                  L     DAT,RMDR           TABLE ENTRY TO DAT                 32511500
0294   33F3 5001                  NI    NULL,DAT,1         TEST LSB OF SERVICE POINTER        32511510
0295   17E0 A600                  BALNZ CHANEL(NULL)                                          32511520
                       *                                   IF SET, SERVICE POINTER IS         32511530
                       *                                   ADDRESS PLUS ONE OF CCB            32511540
                       *                                   IF RESET, SERVICE POINTER IS       32511550
                       *                                   ADDRESS IN FIRST 64K OF A          32511560
                       *                                   USER'S SERVICE SUBROUTINE          32511570
0296   3753 5039                  NI    CLOC,DAT,BI16.30,I ENTRY IS LOCATION COUNT:           32511580
0297   4BFF 6FD2       EXIT26     LWFF  NULL,NULL,IRD      RESET WAIT INDICATOR, EXECUTE.     32511590


                       * ********************************************************            32511610
                       *                                                                      32511620
                       *          AUTO-DRIVER CHANNEL                                         32511630
                       *                                                                      32511640
                       * ********************************************************            32511650

                       * CCW BIT DESIGNATIONS                                                 32511670

0000 0080              EBIT       EQU   '80'               EXECUTE        TEMP   = MR0        32511690
0000 0020              SBIT       EQU   '20'               SDLC CHECKTYPE                     32511700
0000 0010              CBIT       EQU   '10'               CHECK TYPE     DEV    = MR1        32511710
0000 0008              BBIT       EQU   '08'               BUFFER SWITCH  CCW    = MR2        32511720
0000 0004              RWBIT      EQU   '04'               READ/WRITE     DAT    = MR3        32511730
0000 0002              TBIT       EQU   '02'               TRANSLATE      COUNT  = MR4        32511740
0000 0001              FBIT       EQU   '01'               FAST MODE      RETURN = MR5        32511750

0298   3793 5039       CHANEL     NI    MAR,DAT,BI16.30,I  MAR = EVEN ADRS(CCW)               32511770
0299   289F 1E07                  L     R4,MAR,DR2         COPY TO R4, FETCH CCW              32511780
029A   2A5F 1D80                  L     CCW,RMDR                                              32511790
029B   3212 5080                  NI    MR0,CCW,EBIT       TEST THE EXECUTE BIT               32511800
029C   13E0 ADC0                  BALZ  EXSUB0(NULL)       NO EXECUTE, CC=0 (P.27)            32511810
                       *                                                                      32511820
029D   4A7F E940                  EXB   DAT,CCW            ISOLATE STATUS MASK                32511830
029E   2BE3 5980                  N     NULL,R3,DAT        TEST DEVICE STATUS AGAINST MASK    32511840
029F   17E0 AD80                  BALNZ EXSUB1(NULL)       BAD STATUS (P.27)                  32511850
02A0   3384 1002                  AI    MAR,R4,2           ADDRESS BUF0 BYTE COUNT            32511860
02A1   33F2 5001                  NI    NULL,CCW,FBIT      TEST IF FAST MODE                  32511870
02A2   13E0 AFC7                  BALZ  NFAST(NULL),DR2    NOT FAST MODE (P.28)               32511880
                       *                                   ELSE, FETCH BUF0 BYTE COUNT.       32511890

                       *                F A S T   M O D E            *                        32511910

02A3   3384 1004       FASTMODE   AI    MAR,R4,4           POINT TO BUF0 END ADRS             32511930
02A4   2A9F 1D80                  L     COUNT,RMDR         TEST BYTE COUNT:                   32511940
02A5   13E8 AF0F                  BALG  EXAUTO(NULL),DR4   EXIT, COUNT POSITIVE (P.27)        32511950
                       *                                                                      32511960
02A6   43FF EFED                  THWX  NULL,NULL,BYTEIO,C TEST HW LINE (P.27)                32511970
                       *                                                                      32511980
                       * FALL THROUGH IF LINE IS ACTIVE                                       32511990
```

ROM SEGMENTS A, B - I/O INTERRUPT PROCESSOR

```
                                 1201  *                                                          32512000
02A7   2B94 1D80                 1202  HWIO      A     MAR,COUNT,RMDR       BUFFER END ADRS + COUNT    32512010
02A8   33F2 5004                 1203            NI    NULL,CCW,RWBIT       TEST R/W BIT               32512020
02A9   13E0 AAC7                 1204            BALZ  HWRD(NULL),DR2       BRANCH: R/W = 0 = READ     32512030
02AA   43FF 5DAC                 1205  HWRT      WHX   NULL,RMDR,HWRT1      WRITE HALFWORD             32512040
                                 1206  *                                                          32512050
02AB   4B7F 4F97                 1207  HWRD      RH    WMDR,NULL,DW2        READ HALFWORD, STORE       32512060
02AC   2294 3FB2                 1208  HWRT1     AINCX COUNT,COUNT,NULL,COMMON                      32512070

02AD   2B94 1D80                 1210  BYTEIO    A     MAR,COUNT,RMDR       BUFFER END ADRS + COUNT    32512090
02AE   33F2 5004                 1211            NI    NULL,CCW,RWBIT       TEST R/W BIT               32512100
02AF   13E0 AC4B                 1212            BALZ  FRD(NULL),DR1        BRANCH:READ BYTE           32512110
02B0   43FF 1DB2                 1213  FWT       WDX   NULL,RMDR,COMMON     OUTPUT DATA BYTE           32512120
                                 1214  *                                                          32512130
02B1   4B7F 0FDB                 1215  FRD       RDR   WMDR,NULL,DW1        INPUT DATA BYTE, STORE IT. 32512140
                                 1216  *                                                          32512150
02B2   3384 1002                 1217  COMMON    AI    MAR,R4,2             ADDRESS BYTE COUNT         32512160
02B3   2B74 3F97                 1218            AINC  WMDR,COUNT,NULL,DW2 ADJUST COUNT, STORE       32512170
02B4   17E8 AF00                 1219            BALNG EXAUTO(NULL)         EXIT IF NOT >0             32512180

                                 1221  * EXIT TO SUBROUTINE AT BUFFER END (POSITIVE BYTE COUNT)        32512200
02B5   221F 3FB7                 1222  EXSUB2    AINCX MR0,NULL,NULL,EXSUB QUEUE G FLAG              32512210

                                 1224  * EXIT TO SUBROUTINE ON STATUS ERROR                          32512230
02B6   2A1F 2F80                 1225  EXSUB1    SDEC  MR0,NULL,NULL        QUEUE L FLAG               32512240

                                 1227  * UNCONDITIONAL EXIT TO SUBROUTINE (EXECUTE BIT = 0)           32512260
0000 02B7                        1228  EXSUB0    EQU   *                    MR0 CONTAINS ZERO          32512270

02B7   367F 1039                 1230  EXSUB     LI    MR3,BI16.30,I        MR3 = '0000FFFE'           32512290
02B8   3384 1014                 1231            AI    MAR,R4,20            MAR = A(SUBROUTINE ADDRESS) 32512300
02B9   2BFF 1827                 1232            L     NULL,MR0,DR2,E       FETCH SUBR ADDRESS, ADJUST CC 32512310
02BA   2B53 5D80                 1233            N     CLOC,MR3,RMDR                                   32512320
02BB   4BFF 6FD2                 1234  EXIT27    LWFF  NULL,NULL,IRD        FETCH USER INSTRUCTION.    32512330

                                 1236  * NORMAL EXIT FROM AUTO DRIVER CHANNEL                        32512350
02BC   2B5F 1080                 1237  EXAUTO    L     CLOC,R1              GET UNINCREMENTED LOC      32512360
02BD   2BBF 1011                 1238            L     PSW,R0,@LOC          RESTORE ENTRY PSW          32512370
02BE   13FC 9050                 1239            BALA  TWAIT1(NULL),D       GO TEST WAIT BIT. (P.22)   32512380
```

```
                  *            N O R M A L   M O D E             *                  32512400

02BF   3212 5008  NFAST    NI    TEMP,CCW,BBIT       TEST BUFFER SWITCH BIT            32512420
02C0   3210 7002           OI    TEMP,TEMP,2         FORM BYTE COUNT DISPLACEMENT      32512430
                  *                                                                   32512440
02C1   2B84 1807           A     MAR,R4,TEMP,DR2     FETCH BUFFER BYTE COUNT           32512450
02C2   2A04 1800           A     TEMP,R4,TEMP        TEMP = ADRS OF BUFFER BYTE COUNT  32512460
02C3   3390 1002           AI    MAR,TEMP,2                                            32512470
02C4   2A9F 1D8F           L     COUNT,RMDR,DR4      FETCH BUFFER END ADDRESS          32512480
02C5   13E8 AF00           BALG  EXAUTO(NULL)        EXIT IF COUNT POSITIVE (P.27)     32512490
02C6   2A34 1D80           A     MR1,COUNT,RMDR      BUFFER END ADRS + COUNT           32512500
                  *                                                                   32512510
                  * BUFFER BYTE  COUNT IN REGISTER "COUNT"                             32512520
                  * ADDRESS OF BUFFER BYTE COUNT IN "TEMP"                             32512530
                  * BUFFER END ADRS + BYTE COUNT IN "MR1"                              32512540
                  *                                                                   32512550
                  * NOTE: IN NON-FAST MODE, ONLY BYTE TRANSFERS ARE ALLOWED            32512560
                  *                                                                   32512570
02C7   2B9F 1880           L     MAR,MR1             MAR = BYTE ADDRESS                32512580
02C8   33F2 5004           NI    NULL,CCW,RWBIT      TEST R/W BIT                      32512590
02C9   13E0 B8CB           BALZ  NFREAD(NULL),DR1    BRANCH: R/W = 0 = READ (P.29)     32512600
02CA   33F2 5002  NFWRIT   NI    NULL,CCW,TBIT       TRANSLATION SPECIFIED ?           32512610
02CB   16A0 BA80           BALNZ WTRANSL(RETURN)     BRANCH: MUST TRANSLATE (P.29)     32512620
02CC   4BFF 1DC0           WDR   NULL,RMDR           OUTPUT APPROPRIATE BYTE           32512630
02CD   4A7F 5DC0           LBR   MR3,RMDR            COPY BYTE USED IN I/O             32512640
02CE   12B8 B400           BAL   REDCHK(RETURN)      FORM CHECKSUM                     32512650
02CF   239F 181E           LX    MAR,TEMP,COMMON3    GO UPDATE BYTE COUNT.             32512660
                  *                                                                   32512670
                  * ONLY THE BYTE ACTUALLY TRANSFERRED IS INCLUDED IN THE              32512680
                  * LRC OR CRC.  SPECIAL CHARACTERS ARE NOT INCLUDED.                  32512690
                  *                                                                   32512700
02D0   3384 1008  REDCHK   AI    MAR,R4,8            MAR = A(CHECKWORD)                32512710
02D1   32F2 5030           NI    MR7,CCW,CBIT+SBIT   CHECK TYPE BITS                   32512720
02D2   17E0 B546           BALNZ CRCCK(NULL),PR2     BRANCH: CRC REQUIRED              32512730
02D3   2B73 6D96           X     WMDR,MR3,RMDR,PW2   DO LONGITUDINAL CHECK             32512740
02D4   03F8 0A80           BAL   (RETURN)(NULL)      RETURN TO CALLER                  32512750
                  *                                                                   32512760
02D5   4BFF 7F86  CRCCK    SMCR  NULL,NULL,PR2       IS CRC ASSIST UNIT EQUIPPED ?     32512770
02D6   17E9 5280           BALNG CRC16B(NULL)        BRANCH: NO (P.47) - USES 'RETURN' 32512780
0000 0006         CRC      EQU   6                   CRC HARDWARE ASSIST DEVICE ADDRESS 32512790
02D7   32DF 1006  HWASSIST LI    MR6,CRC             ASSIST UNIT ADDRESS               32512800
02D8   32F7 8005           SRLI  MR7,MR7,5           POSITION CHECKTYPE BITS           32512810
                  *                                  0 = CRC12; 1 = CRC SDLC.          32512820
02D9   4BF6 BBC0           OCRA  NULL,MR6,MR7        COMMAND CHECKTYPE                 32512830
02DA   4BFF 5D80           WH    NULL,RMDR           OLD RESIDUAL                      32512840
02DB   4BFF 19C0           WDR   NULL,MR3            UNTRANSLATED DATA BYTE            32512850
02DC   4B7F 4F96           RH    WMDR,NULL,PW2       NEW RESIDUAL                      32512860

02DD   2B9F 1800  COMMON2  L     MAR,TEMP            MAR = A(BYTE COUNT)               32512880
02DE   2B74 3F96  COMMON3  AINC  WMDR,COUNT,NULL,PW2 INCREMENT & STORE COUNT           32512890
02DF   17E8 AF00           BALNG EXAUTO(NULL)        EXIT IF NOT POSITIVE (P.27)       32512900
02E0   3372 6008  BUFSW    XI    WMDR,CCW,BBIT       COMPLEMENT BUFFER BIT             32512910
02E1   2B9F 1217           L     MAR,R4,DW2          AND UPDATE CCW                    32512920
02E2   13F8 AD40           BAL   EXSUB2(NULL)        EXIT TO SUBROUTINE (P.27)         32512930
```

ROM SEGMENTS A, B - I/O INTERRUPT PROCESSOR

```
02E3    4A7F 0FC0           1296  NFREAD   RDR   MR3,NULL           INPUT THE BYTE                         32512950
02E4    2B7F 1980           1297           L     WMDR,MR3           PREPARE TO STORE IT -                  32512960
02E5    33F2 5002           1298           NI    NULL,CCW,TBIT      TRANSLATION REQUIRED ?                 32512970
02E6    16A0 BAC0           1299           BALNZ RTRANSL(RETURN)    DO TRANSLATION.                        32512980
02E7    2B9F 189B           1300           L     MAR,MR1,DW1        WRITE TO MEMORY.                       32512990
02E8    12B8 B400           1301           BAL   REDCHK(RETURN)     INCLUDE DATA IN CHECKSUM (P.28)        32513000
02E9    239F 181E           1302           LX    MAR,TEMP,COMMON3   GO UPDATE BYTE COUNT.                  32513010

0000 02EA                   1304  TRANSL   EQU   *                  CHANNEL TO TRANSLATE I/O BYTES         32513030
0000 02EA                   1305  WTRANSL  EQU   *                  TRANSLATION WHILE WRITING              32513040
02EA    4A7F 5DC0           1306           LBR   MR3,RMDR           BYTE TO TRANSLATE                      32513050
0000 02EB                   1307  RTRANSL  EQU   *                  TRANSLATION WHILE READING              32513060
02EB    3384 1010           1308           AI    MAR,R4,16          A(TRANSLATION TABLE ADRS)              32513070
02EC    2AD3 198C           1309           A     MR6,MR3,MR3,PR4    DOUBLE DATA BYTE FOR INDEX             32513080
02ED    2B96 1D86           1310           A     MAR,MR6,RMDR,PR2   FETCH HALFWORD TABLE ENTRY.            32513090
02EE    2B7F 1D80           1311           L     WMDR,RMDR          COPY IN CASE NEGATIVE & READING        32513100
02EF    2ADF 1D80           1312           L     MR6,RMDR           TEST IF NEGATIVE                       32513110
02F0    03E4 0A80           1313           BALL  (RETURN)(NULL)     BRANCH: WE HAVE A CHARACTER            32513120
02F1    2B56 1D80           1314           A     CLOC,MR6,RMDR      ENTRY IS (ROUTINE ADRS)/2;             32513130
02F2    287F 1980           1315           L     R3,MR3             UNTRANSLATED BYTE                      32513140
02F3    33BD 5FF0           1316           NI    PSW,PSW,'FF0'      SET CC = 0                             32513150
02F4    4BFF 6FD2           1317  EXIT29   LWFF  NULL,NULL,IRD      RESET WAIT INDICATOR, EXIT, CC = 0.    32513160
0000 02F6                   1318  CHANEND  EQU   *+1                USED TO SORT FAULTS               R02  32513170

02F5    37BF 1051           1321  MMFINT2  LI    PSW,BIT18,I        ENABLE MMFINT, ONLY               R02  32513200
02F6    2B7F 1D1D           1322           L     WMDR,ILOC,I4DW4    '24-27' = UNINCREMENTED LOC       R02  32513210
02F7    339F 1038           1323           LI    MAR,'38'           A(MMFINT NEW PSW)                      32513220
02F8    2B7F 1B8F           1324           L     WMDR,MR7,DR4       PREPARE FOR FAULTED 'LM'               32513230
                            1325  *                                 FETCH MMFINT NEW PSW                   32513240
02F9    2A7F 1D8E           1326           L     MR3,RMDR,I4DR4     MR3 = NEW PSW                          32513250
02FA    339F 102C           1327           LI    MAR,'2C'           A(MMFINT 'LM' FAULT ADDRESS)           32513260
02FB    2B5F 1D9C           1328           L     CLOC,RMDR,PW4      STORE IT;                              32513270
02FC    2BBF 1991           1329           L     PSW,MR3,@LOC       NEW PSW; UPDATE ILOC.                  32513280
02FD    13FC 9010           1330           BALA  TWAIT(NULL),D      (P.22)                                 32513290
0000 02FE                   1331  MMFEND   EQU   *                  USED TO TEST DOUBLE FAULT              32513300

02FE    335F 1040           1333  HARDSTOP LI    CLOC,'040'         POINT TO MALFUNCTION STATUS            32513320
02FF    2BFF 1F91           1334           L     NULL,NULL,@LOC     AND STOP MACHINE                       32513330
                            1335  *                                 BY ENTERING 'CONSER'.             R02  32513340
```

```
0300                     1337          ORG    '300'                                        32513360

0000 0010                1339  INDEV    EQU    '10'                 FDX RECEIVER            32513380
0000 0011                1340  OUTDEV   EQU    '11'                 FDX TRANSMITTER         32513390
0000 0021                1341  INCMD    EQU    '21'                 DTR, READ               32513400
0000 0023                1342  OUTCMD   EQU    '23'                 DTR, WRITE MODE         32513410
0000 00EE                1343  FMTCMD   EQU    'EE'                 ASYNC FORMAT COMMAND - 7 DATA BITS,  32513420
                         1344  *                                    2 STOP BITS, EVEN PARITY, FAST CLK.  32513430
0000 003C                1345  PROMPTC  EQU    C'<'                 PROMPT CHARACTER        32513440
                         1346  *                                                            32513450
                         1347  * (MR2-MR3) = ACCUMULATOR        MR0 = I/O CHARACTER        32513460
                         1348  * MR4 = DIGIT  COUNTER FOR 'PRNTREG'                         32513470
                         1349  *                                                            32513480
0000 0300                1350  CONSER   EQU    *                    CONSOLE SERVICE ROUTINE 32513490
0300   335E 5FFE         1351           NI     CLOC,ILOC,'FFE'      NEXT INSTRUCTION TO EXECUTE  32513500
0301   4AFF 7F85         1352           SMCR   MR7,NULL,RFAULT      PPF ? RESET RX FLOPS.   32513510
0302   13E5 5800         1353           BALL   PWRDWN(NULL)         BRANCH: YES. (P.48)     32513520
0303   4BFF 7BC0         1354           CMCR   NULL,MR7             RESET ALL BITS          32513530
0304   2A1F 2F80         1355           SDEC   MR0,NULL,NULL                                32513540
0305   4BFF 6840         1356           LWFF   NULL,MR0             SET WAIT INDICATOR      32513550
0306   323F 1010         1357           LI     MR1,INDEV                                    32513560
0307   53F1 B0EE         1358           OCAI   NULL,MR1,FMTCMD      SET BAUD RATE AND FORMAT  32513570
0308   337F 8008         1359           SRLI   WMDR,NULL,8          DELAY                   32513580
0309   53F1 B021         1360           OCAI   NULL,MR1,INCMD       COMMAND READ MODE       32513590
030A   33FF 8008         1361           SRLI   NULL,NULL,8          DELAY                   32513600
030B   325F 1011         1362           LI     MR2,OUTDEV                                   32513610
030C   53F2 B023         1363           OCAI   NULL,MR2,OUTCMD      COMMAND WRITE MODE      32513620
030D   33FF 8008         1364           SRLI   NULL,NULL,8          DELAY                   32513630
030E   4BF1 8FC0         1365           RDRA   NULL,MR1,NULL        DUMMY READ TO SET BSY   32513640
030F   4A1F 7F80         1366  CLOOP    SMCR   MR0,NULL             SENSE MCR               32513650
0310   33F0 5021         1367           NI     NULL,MR0,'021'       EXE/HALT, OR PPF ?      32513660
0311   17E0 F440         1368           BALNZ  IDLE(NULL)           BRANCH: YES. (P.35)     32513670
0312   4BFF 2FC0         1369           SSR    NULL,NULL            DEVICE STATUS           32513680
0313   17F0 C3C0         1370           BALNC  CLOOP(NULL)          WAIT FOR BSY.           32513690
                         1371  * FULL-DUPLEX DEVICE IS ASSUMED.                             32513700
0314   339F 1028         1372           LI     MAR,'28'             A(CONSOLE STATUS)       32513710
0315   2B7F 2F9C         1373           SDEC   WMDR,NULL,NULL,PW4   SET NEGATIVE FLAG       32513720

0316   1298 E140         1375  ENTRY    BAL    CRLF(MR4)            DO CARRIAGE RETN, LINE FEED (P.33)  32513740

0317   2A7D 1F91         1377  SHOWPSW  A      MR3,PSW,NULL,@LOC    WILL PRINT PSW; UPDATE ILOC.  32513760
0318   12B8 DD80         1378           BAL    PRNTREG6(MR5)        PRINT PSW VALUE (P.33)  32513770

0319   327E 5FFE         1380  SHOWLOC  NI     MR3,ILOC,'FFE'       GET CURRENT LOC, FORCED EVEN  32513790
031A   12B8 DD80         1381           BAL    PRNTREG6(MR5)        PRINT LOC VALUE (P.33)  32513800
031B   1298 E140         1382  PROMPT   BAL    CRLF(MR4)            DO CARRIAGE RETN, LINE FEED (P.33)  32513810
031C   321F 103C         1383           LI     MR0,PROMPTC          PROMPT CHARACTER        32513820
031D   12D8 E240         1384           BAL    OUTCHR(MR6)          OUTPUT CHARACTER (P.33) 32513830
031E   12D8 CAC0         1385           BAL    INCHR(MR6)           GET FIRST CHARACTER (P.31)  32513840
031F   329F 135A         1386  DECODE   LI     MR4,DECTABE-1        END OF TABLE            32513850
0320   2E3F 1A00         1387  DECODE1  L      MR1,MR4,I            GET TABLE ENTRY         32513860
0321   32B1 B008         1388           RLLI   MR5,MR1,8            POSITION -              32513870
0322   4ABF 5AC0         1389           LBR    MR5,MR5              EXTRACT CHARACTER       32513880
```

ROM SEGMENTS C, D, E .... CONSOLE SUPPORT ROUTINE

```
0323  2BF5 0800     1390          S     NULL,MR5,MR0       WHAT WAS INPUT ?                          32513890
0324  03E0 0880     1391          BALZ  (MR1)(NULL)        IF SO, GO TO IT.                          32513900
0325  2A94 2F80     1392          SDEC  MR4,MR4,NULL       DECREMENT COUNT                           32513910
0326  33F4 034B     1393          SI    NULL,MR4,DECTAB    DONE ?                                    32513920
0327  17F0 C800     1394          BALNC DECODE1(NULL)      BRANCH: NOT YET. (P.30)                   32513930

0328  1298 E140     1396 QUESTN   BAL   CRLF(MR4)          DO CARRIAGE RETN, LINE FEED (P.33)        32513950
0329  53FF 103F     1397          WDI   NULL,C'?'          QUESTION INPUT                            32513960
032A  13F8 C6C0     1398          BAL   PROMPT(NULL)       GET NEXT REQUEST (P.30)                   32513970


                    1400 * READ CHARACTER FROM CONSOLE DEVICE                                        32513990
032B  323F 1010     1401 INCHR    LI    MR1,INDEV                                                    32514000
032C  4A1F 7F80     1402          SMCR  MR0,NULL           GET MCR REGISTER                          32514010
032D  13E5 5800     1403          BALL  PPFINT(NULL)       BRANCH: POWER GONE. (P.48)                32514020
032E  33F0 5020     1404          NI    NULL,MR0,'020'     EXE/HLT ?                                 32514030
032F  17E0 F240     1405          BALNZ IS.PRMPT(NULL)     IF BAD-STATUS HANG (P.35)                 32514040
                    1406 *                                 FROM INPUT DEVICE, DEPRESSING THE         32514050
                    1407 *                                 RUN SWITCH CAUSES THE PROCESSOR           32514060
                    1408 *                                 TO ENTER THE RUN MODE AT THE              32514070
                    1409 *                                 ADDRESS SPECIFIED BY CLOC.                32514080
0330  4BF1 AFC0     1410          SSRA  NULL,MR1,NULL      DEVICE STATUS                             32514090
0331  13F0 CAC0     1411          BALC  INCHR(NULL)        WAIT FOR NOT BSY.                         32514100
0332  4A1F 0FC0     1412          RDR   MR0,NULL           INPUT CHARACTER                           32514110
0333  4BFF 2FC0     1413          SSR   NULL,NULL                                                    32514120
0334  17F0 CCC0     1414          BALNC *-1(NULL)          WAIT FOR BSY AGAIN.                       32514130
0335  323F 1011     1415          LI    MR1,OUTDEV                                                   32514140
0336  4BF1 AFC0     1416          SSRA  NULL,MR1,NULL                                                32514150
0337  13F0 CD80     1417          BALC  *-1(NULL)          WAIT FOR NOT BSY                          32514160
0338  4BFF 1840     1418          WDR   NULL,MR0           ECHO RECEIVED CHARACTER TO OUTPUT         32514170
0339  4BFF 2FC0     1419          SSR   NULL,NULL                                                    32514180
033A  13F0 CE40     1420          BALC  *-1(NULL)          WAIT FOR NOT BSY                          32514190
033B  3210 507F     1421          NI    MR0,MR0,'7F'       MASK TO 7-BITS                            32514200
033C  33F0 0020     1422          SI    NULL,MR0,'20'      SPACE ?                                   32514210
033D  13E0 CAC0     1423          BALZ  INCHR(NULL)        BRANCH: YES.                              32514220
033E  33F0 0060     1424          SI    NULL,MR0,'60'      LOWER-CASE ?                              32514230
033F  13F0 D040     1425          BALC  IF.DELE(NULL)      BRANCH: NO.                               32514240
0340  3210 0020     1426          SI    MR0,MR0,'20'       YES. DECREMENT BY 20.                     32514250
0341  33F0 005F     1427 IF.DELE  SI    NULL,MR0,'5F'      BACK-ARROW, UNDERLINE, DELETE ?           32514260
0342  13E0 D140     1428          BALZ  IS.DELE(NULL)      BRANCH: YES.                              32514270
0343  33F0 0008     1429          SI    NULL,MR0,'08'      BACKSPACE ?                               32514280
0344  07E0 0B00     1430          BALNZ (MR6)(NULL)        BRANCH: NO. RETURN TO CALLER.             32514290
0345  3273 8004     1431 IS.DELE  SRLI  MR3,MR3,4          DELETE LAST ACCUMULATED DIGIT             32514300
0346  3232 500F     1432          NI    MR1,MR2,'0F'       KEEP LS DIGIT, MS ACCUM.                  32514310
0347  3231 A004     1433          RRLI  MR1,MR1,4          POSITION TO BITS 0:3                      32514320
0348  2A73 7880     1434          O     MR3,MR3,MR1        AND MAKE MS DIGIT OF LS ACCUMULATOR       32514330
0349  3252 8004     1435          SRLI  MR2,MR2,4          SHIFT MS ACCUMULATOR                      32514340
034A  13F8 CAC0     1436          BAL   INCHR(NULL)        AND TRY AGAIN.                            32514350
```

```
                   * TABLES USED FOR CONVERSION                                                      32514370
                   * HEXASCII, ASCIIHEX, INTERPRETER BRANCHES.                                       32514380
                   *                                                                                 32514390
034B   3C30 03C9   DECTAB   DC    IS.PRMPT+'3C300000' <  +  0  +  ROUTINE (P.35)                     32514400
034C   4031 0395            DC    IS.AT+'40310000'    @  +  1  +  ROUTINE (P.33)                     32514410
034D   2B32 0399            DC    IS.PLUS+'2B320000'  +  +  2  +  ROUTINE (P.33)                     32514420
034E   2D33 0398            DC    IS.MINUS+'2D330000' -  +  3  +  ROUTINE (P.33)                     32514430
034F   3F34 0328            DC    QUESTN+'3F340000'   ?  +  4  +  ROUTINE (P.31)                     32514440
0350   5235 03A4            DC    IS.R+'52350000'     R  +  5  +  ROUTINE (P.34)                     32514450
0351   4636 03AA            DC    IS.F+'46360000'     F  +  6  +  ROUTINE (P.34)                     32514460
0352   4437 03B1            DC    IS.D+'44370000'     D  +  7  +  ROUTINE (P.34)                     32514470
0353   5038 03BE            DC    IS.P+'50380000'     P  +  8  +  ROUTINE (P.34)                     32514480
0354   5F39 032B            DC    INCHR+'5F390000'    DEL+  9  +  ROUTINE (P.31)                     32514490
0355   0841 032B            DC    INCHR+'08410000'    BS +  A  +  ROUTINE (P.31)                     32514500
0356   2042 032B            DC    INCHR+'20420000'    SP +  B  +  ROUTINE (P.31)                     32514510
0357   3D43 0328            DC    QUESTN+'3D430000'   =  +  C  +  ROUTINE (P.31)                     32514520
0358   0044 0328            DC    QUESTN+'00440000'   ......D  +  ROUTINE (P.31)                     32514530
0359   0045 0328            DC    QUESTN+'00450000'   ......E  +  ROUTINE (P.31)                     32514540
035A   0046 0328            DC    QUESTN+'00460000'   ......F  +  ROUTINE (P.31)                     32514550
0000 035B          DECTABE  EQU   *                   END OF TABLE                                   32514560

035B   1298 E140   TRYMOD   BAL   CRLF(MR4)           DO CARRIAGE RETN, LINE FEED (P.33)             32514580
035C   321F 103C            LI    MR0,PROMPTC         PROMPT CHARACTER                               32514590
035D   12D8 E240            BAL   OUTCHR(MR6)         OUTPUT CHARACTER (P.33)                        32514600
035E   12D8 CAC0            BAL   INCHR(MR6)          READ 1ST CHARACTER (P.31)                      32514610
035F   33F0 003D            SI    NULL,MR0,C'='       EQUAL SIGN ?                                   32514620
0360   17E0 C7C0            BALNZ DECODE(NULL)        BRANCH: NO. (P.30)                             32514630

                   * ACCUMULATE HEXADECIMAL INPUT                                                    32514650
                   * USES CHAINED (MR2-MR3) AS 64-BIT ACCUMULATOR                                    32514660
0361   3253 E000   ACCUM    MI    MR2,MR3,0           CLEAR ACCUMULATOR                              32514670
0362   12D8 CAC0   ACCUM1   BAL   INCHR(MR6)          (P.31)                                         32514680
0363   33F0 600D            XI    NULL,MR0,'0D'       CARRIAGE RETURN ENTERED ?                      32514690
0364   03E0 0A80            BALZ  (MR5)(NULL)         RETURN TO CALLER IF YES.                       32514700
0000 0365          ASCHEX   EQU   *                   CONVERSION FROM ASCII TO HEXADECIMA            32514710
0365   329F 135A            LI    MR4,DECTABE-1       END OF TABLE                                   32514720
0366   2E3F 1A00   DECODE2  L     MR1,MR4,I           GET TABLE ENTRY                                32514730
0367   32F1 B010            RLLI  MR7,MR1,16          POSITION -                                     32514740
0368   4AFF 5BC0            LBR   MR7,MR7             EXTRACT CHARACTER                              32514750
0369   2BF7 6800            X     NULL,MR7,MR0        WHAT WAS INPUT ?                               32514760
036A   17E0 DC80            BALNZ ASCHEX1(NULL)       BRANCH: NO.                                    32514770
036B   32F4 034B            SI    MR7,MR4,DECTAB      CONVERT TO DIGIT                               32514780
036C   3252 9004            SLLI  MR2,MR2,4           HIGH HALF                                      32514790
036D   3273 B004            RLLI  MR3,MR3,4           LOW HALF                                       32514800
036E   3293 500F            NI    MR4,MR3,'0F'        EXTRACT OLD HIGH DIGIT, LOW HALF               32514810
036F   2A52 7A00            O     MR2,MR2,MR4         AND MOVE TO LOW DIGIT, HIGH HALF               32514820
0370   2A73 6A00            X     MR3,MR3,MR4         REMOVE FROM LOW HALF                           32514830
0371   2273 7BA2            OX    MR3,MR3,MR7,ACCUM1  APPEND NEW DIGIT, TRY AGAIN.                   32514840
0000 0372          ASCHEX1  EQU   *                   NO MATCH                                       32514850
0372   3294 0001            SI    MR4,MR4,1           DECREMENT COUNTER                              32514860
0373   33F4 034B            SI    NULL,MR4,DECTAB     FAILED TO MATCH ?                              32514870
0374   13F0 CA00            BALC  QUESTN(NULL)        BRANCH: YES. (P.31)                            32514880
0375   13F8 D980            BAL   DECODE2(NULL)       TRY AGAIN.                                     32514890
```

```
                           1492  * PRINT REGISTER CONTENTS                                            32514910
0376    329F 1005          1493  PRNTREG6 LI    MR4,5             SET DIGIT COUNT                       32514920
0377    23FF 1FBA          1494           LX    NULL,NULL,PRNTREG                                       32514930
0378    1298 E140          1495  PRNTLF8  BAL   CRLF(MR4)         DO CARRIAGE RETN, LINE FEED (P.33)    32514940
0379    329F 1007          1496  PRNTREG8 LI    MR4,7             SET DIGIT COUNT                       32514950
037A    3254 9002          1497  PRNTREG  SLLI  MR2,MR4,2         SET UP SHIFT COUNTER                  32514960
037B    2A13 8900          1498           SRL   MR0,MR3,MR2       SHIFT DIGIT TO MR0(27:31)             32514970
037C    3210 500F          1499           NI    MR0,MR0,'0F'      EXTRACT DIGIT                         32514980
037D    3210 134B          1500  HEXASC   AI    MR0,MR0,DECTAB    FORM INDEX                            32514990
037E    2E1F 1800          1501           L     MR0,MR0,I         FETCH ENTRY                           32515000
037F    3210 B010          1502           RLLI  MR0,MR0,16        POSITION CHARACTER TO BITS 24:31      32515010
0380    12D8 E240          1503           BAL   OUTCHR(MR6)       OUTPUT CHARACTER (P.33)               32515020
0381    2A94 2F80          1504  PREG.0   SDEC  MR4,MR4,NULL      DECREMENT COUNT                       32515030
0382    17F0 DE80          1505           BALNC PRNTREG(NULL)     LOOP 'TIL DONE;                       32515040
                           1506  *                                TRANSFER IF NOT DONE.                 32515050
0383    53FF 1020          1507           WDI   NULL,C' '         OUTPUT A SPACE                        32515060
0384    03F8 0A80          1508           BAL   (MR5)(NULL)       RETURN TO CALLER.                     32515070

                           1510  * PERFORM CARRIAGE RETURN/LINE FEED                                  32515090
0385    321F 100A          1511  CRLF     LI    MR0,'0A'          LINE FEED                             32515100
0386    12D8 E240          1512           BAL   OUTCHR(MR6)       OUTPUT CHARACTER (P.33)               32515110
0387    321F 100D          1513           LI    MR0,'0D'          CARRIAGE RETURN                       32515120
0388    22DF 1A09          1514           LX    MR6,MR4,OUTCHR                                          32515130

                           1516  * OUTPUT CHARACTER TO CONSOLE                                        32515150
0389    323F 1011          1517  OUTCHR   LI    MR1,OUTDEV                                              32515160
038A    4BF1 AFC0          1518           SSRA  NULL,MR1,NULL     ADDRESS, GET STATUS                   32515170
038B    13F4 F440          1519           BALV  IDLE(NULL)        IDLE IF BAD STATUS (P.35)             32515180
038C    17E0 F440          1520           BALNZ IDLE(NULL)        IDLE IF BAD STATUS (P.35)             32515190
038D    13F0 E240          1521           BALC  OUTCHR(NULL)      WAIT FOR NOT BSY                      32515200
038E    4BFF 1840          1522           WDR   NULL,MR0          OUTPUT CHARACTER                      32515210
038F    4BFF 2FC0          1523           SSR   NULL,NULL                                               32515220
0390    13F0 E3C0          1524           BALC  *-1(NULL)         WAIT FOR NOT BSY                      32515230
0391    4BFF 1FC0          1525           WDR   NULL,NULL         OUTPUT NULL                           32515240
0392    4BFF 2FC0          1526           SSR   NULL,NULL                                               32515250
0393    13F0 E480          1527           BALC  *-1(NULL)         WAIT FOR NOT BSY                      32515260
0394    03F8 0B00          1528           BAL   (MR6)(NULL)       RETURN TO CALLER                      32515270

                           1530  * MODIFY LOCATION COUNTER                                            32515290
0395    12B8 D840          1531  IS.AT    BAL   ACCUM(MR5)        GO GET DATA (P.32)                    32515300
0396    3353 5FFE          1532           NI    CLOC,MR3,'FFE'    NEW LOC, FORCED EVEN                  32515310
0397    13F8 E680          1533           BAL   IS.PLO(NULL)      GO DISPLAY.                           32515320

                           1535  * PROCEED TO PREVIOUS CELL                                           32515340
0398    335C 0004          1536  IS.MINUS SI    CLOC,CLOC,4       DECREMENT BY 4                        32515350

                           1538  * PROCEED TO NEXT CELL                                               32515370
0399    335C 1002          1539  IS.PLUS  AI    CLOC,CLOC,2       INCREMENT BY 2                        32515380
039A    1298 E140          1540  IS.PLO   BAL   CRLF(MR4)         DO CARRIAGE RETN, LINE FEED (P.33)    32515390
```

```
039B   2A7C 1F85                  A     MR3,CLOC,NULL,RFAULT GET LOC, INVALIDATE INST BUFFER
039C   12B8 DD80                  BAL   PRNTREG6(MR5)       DISPLAY UPDATED CLOC (P.33)
039D   2B9C 1F87                  A     MAR,CLOC,NULL,DR2   READ CONTENTS OF OPEN CELL
039E   329F 1003                  LI    MR4,3               SET DIGIT COUNT = 4
039F   2A7F 1D91                  L     MR3,RMDR,@LOC       COPY TO ACCUMULATOR; UPDATE ILOC.
03A0   12B8 DE80                  BAL   PRNTREG(MR5)        DISPLAY MEMORY HALFWORD (P.33)
03A1   12B8 D6C0                  BAL   TRYMOD(MR5)         SEE IF USER WANTS CHANGE (P.32)
03A2   2B7F 1997                  L     WMDR,MR3,DW2        STORE NEW DATA;
03A3   13F8 E640                  BAL   IS.PLUS(NULL)       OPEN + CELL AND DISPLAY. (P.33)

                        * DISPLAY GENERAL REGISTER
03A4   12B8 D840        IS.R      BAL   ACCUM(MR5)          GET REGISTER NUMBER (P.32)
03A5   2BDF 1980                  L     YDI,MR3             SELECT REGISTER
03A6   2A7F 1C80                  L     MR3,YD              COPY CONTENTS TO PRINT REGISTER
03A7   12B8 DE00                  BAL   PRNTLF8(MR5)        AND GO PRINT ON NEW LINE (P.33)
03A8   12B8 D6C0        IS.R00    BAL   TRYMOD(MR5)         SEE IF USER WANTS CHANGE (P.32)
03A9   233F 19A8                  LX    YD,MR3,IS.R00       LOAD NEW DATA, GET NEXT REQUEST.

                        * DISPLAY SPFP REGISTER
03AA   12B8 F580        IS.F      BAL   TSTDFU(MR5)         SEE IF FPP EQUIPPED (P.35)
03AB   33D3 500E                  NI    YDI,MR3,'0E'        FORCE USER SELECTION EVEN
03AC   CA7F 1C80                  RRE   MR3,YD              READ REGISTER SPEC'D, INTO YD
03AD   12B8 DE00                  BAL   PRNTLF8(MR5)        AND GO PRINT ON NEW LINE (P.33)
03AE   12B8 D6C0        IS.F00    BAL   TRYMOD(MR5)         SEE IF USER WANTS CHANGE (P.32)
03AF   CBF9 29C0                  LE    YD,MR3,K            LOAD IMAGE DATA
03B0   13F8 EB80                  BAL   IS.F00(NULL)        AND TRY AGAIN

                        * DISPLAY DPFP REGISTER
03B1   12B8 F580        IS.D      BAL   TSTDFU(MR5)         SEE IF FPP EQUIPPED (P.35)
03B2   33D3 500E                  NI    YDI,MR3,'0E'        FORCE USER SELECTION EVEN
03B3   CA7F 9C80                  RRD   MR3,YD              READ SELECTED REGISTER, INTO MR3
03B4   12B8 DE00                  BAL   PRNTLF8(MR5)        AND GO PRINT ON NEW LINE (P.33)
03B5   2BDF 3F00                  AINC  YDI,NULL,YDI
03B6   CA7F 9C80                  RRD   MR3,YD              GET LOW HALF.
03B7   12B8 DE40                  BAL   PRNTREG8(MR5)       SHOW HIGH HALF (P.33)
03B8   2A1F 1F00                  L     MR0,YDI
03B9   2BD0 2F80                  SDEC  YDI,MR0,NULL        POINT BACK TO HIGH HALF
03BA   12B8 D6C0        IS.D00    BAL   TRYMOD(MR5)         SEE IF USER WANTS CHANGE (P.32)
03BB   CBF9 8900                  LW    YD,MR2              LOAD HIGH HALF,
03BC   CBF9 A9C0                  LD    YD,MR3,K            LOAD DOUBLE, IMAGE
03BD   13F8 EE80                  BAL   IS.D00(NULL)        AND TRY AGAIN.

                        * MODIFY PSW
03BE   12D8 CAC0        IS.P      BAL   INCHR(MR6)          GET NEXT INPUT CHARACTER (P.31)
03BF   33F0 600D                  XI    NULL,MR0,X'0D'      CARRIAGE RETURN ?
03C0   17E0 CA00                  BALNZ QUESTN(NULL)        BRANCH: NO. (P.31)
03C1   1298 E140                  BAL   CRLF(MR4)           DO CARRIAGE RETN, LINE FEED (P.33)
03C2   2A7D 1F80                  A     MR3,PSW,NULL
03C3   12B8 DD80                  BAL   PRNTREG6(MR5)       DISPLAY PSW (P.33)
03C4   2A7E 1F80                  A     MR3,ILOC,NULL       WILL PRINT LOC
03C5   12B8 DD80                  BAL   PRNTREG6(MR5)       (P.33)
03C6   12B8 D6C0                  BAL   TRYMOD(MR5)         SEE IF USER WANTS CHANGE (P.32)
03C7   2BBF 1980                  L     PSW,MR3             UPDATE PSW,
03C8   13F8 C580                  BAL   ENTRY(NULL)         GO DISPLAY IT. (P.30)
```

ROM SEGMENTS C, D, E .... CONSOLE SUPPORT ROUTINE

```
                                1596  * ENTER RUN MODE                                                       32515950
 0000 03C9                      1597  IS.PRMPT EQU   *                   PROMPT CHARACTER; TO RUN MODE.      32515960
 03C9   339F 1028               1598           LI    MAR,'28'            A(CONSOLE STATUS)                   32515970
 03CA   2B7F 1F9C               1599           L     WMDR,NULL,PW4       RESET FLAG                          32515980
 03CB   1298 E140               1600           BAL   CRLF(MR4)           DO CARRIAGE RETN, LINE FEED (P.33)  32515990
 03CC   37BD 5023               1601           NI    PSW,PSW,BIT160,I    RESET PSW 16                        32516000
 03CD   321F 1120               1602           LI    MR0,'120'                                               32516010
 03CE   4BFF 7840               1603           CMCR  NULL,MR0            RESET EXE/HLT INTERRUPT             32516020
 03CF   4BFF 6FC0               1604           LWFF  NULL,NULL           RESET WAIT INDICATOR, EXIT.         32516030
 0000 03D0                      1605  CONSEND  EQU   *                   USED TO SORT FAULTS                 32516040
 03D0   17F8 F412               1606           BDC   *(NULL),IRD         EXECUTE INSTRUCTION,                32516050
                                1607  *                                  DISALLOW CATN FOR ONE CYCLE.        32516060


 03D1   4AFF 7F80               1609  IDLE     SMCR  MR7,NULL                                                32516080
 03D2   13E5 5800               1610           BALL  PPFINT(NULL)        PRIMARY POWER FAIL (P.48)           32516090
 03D3   33F7 5020               1611           NI    NULL,MR7,'020'      CATN  ?                             32516100
 03D4   13E0 F440               1612           BALZ  IDLE(NULL)          NO, LOOP                            32516110
 03D5   13F8 C000               1613           BAL   CONSER(NULL)        GO TO CONSOLE SERVICE ROUTINE (P.30) 32516120

 03D6   4A1F 7F80               1615  TSTDFU   SMCR  MR0,NULL            COPY MCR TO MR0                     32516140
 03D7   37F0 5183               1616           NI    NULL,MR0,BIT20,I    FPP EQUIPPED ?                      32516150
 03D8   13E0 CA00               1617           BALZ  QUESTN(NULL)        BRANCH: NO. (P.31)                  32516160
 03D9   13F8 D840               1618           BAL   ACCUM(NULL)         RETURNS VIA (MR5) (P.32).           32516170
```

```
               * ******************************************************             32516190
               *                                                                      32516200
               *         PRIVILEGED SYSTEM FUNCTION (PSF)                              32516210
               *                                                                      32516220
               * ******************************************************             32516230

03DA  2B9A 1D80      PSF1     A     MAR,YX,RMDR        CALCULATE 2ND OPERAND ADDRESS        32516250
03DB  32D7 13DF               AI    MR6,MR7,PSFTAB     WHERE TO GET VECTOR                  32516260
03DC  33F7 0009               SI    NULL,MR7,9         LEGAL FUNCTION ?                R02  32516270
03DD  03F0 0B00               BALC  (MR6)(NULL)        BRANCH: YES.                    R02  32516280
03DE  17FC 8240               BALD  ILEGAL(NULL)       ILLEGAL FUNCTION.               R02  32516290

03DF  13F8 FA40      PSFTAB   BAL   REL(NULL)          READ ERROR LOGGER                    32516310
03E0  13F8 FB80               BAL   LPSTD(NULL)        LOAD PROCESS SEGMENT DESCRIPTOR      32516320
03E1  13F8 FC00               BAL   LSSTD(NULL)        LOAD SHARED  SEGMENT DESCRIPTOR      32516330
03E2  13F8 FC80               BAL   STPS(NULL)         STORE PROCESS STATE                  32516340
03E3  13F9 0100               BAL   LDPS(NULL)         LOAD PROCESS STATE        (P.37)     32516350
03E4  13F9 0640               BAL   ISSV(NULL)         SAVE INTERRUPTIBLE STATE (P.37)      32516360
03E5  13F9 0700               BAL   ISRST(NULL)        LOAD INTERRUPTIBLE STATE (P.37)      32516370
03E6  13F9 07C0               BAL   TEL(NULL)          TEST ERROR LOGGER         (P.37)     32516380

               * NOTE - FOLLOWING WORD IS PART OF BRANCH TABLE.                              32516400
0000 03E7            RMVF     EQU   *                  CODE 8 - RESET MEMORY VOLTAGE FAILU  32516410
03E7  321F 1004               LI    MR0,4              MASK                                 32516420
03E8  4BFF 7852               CMCR  NULL,MR0,IRD       RESET MCR BIT 13, EXIT.              32516430

0000 03E9            REL      EQU   *                  CODE 0 - READ ERROR LOGGER           32516450
03E9  2B9F 1C05               L     MAR,YS,RFAULT      ERROR LOGGER ADDRESS FROM R2    R02  32516460
03EA  2BDF 3E8D               AINC  YDI,NULL,YSI,REL   POINT R2+1, READ LOGGER         R02  32516470
03EB  2B3F 1D82               L     YD,RMDR,IR                                              32516480
03EC  3219 9010               SLLI  MR0,YD,16          NEED TO ADJUST CC BASED ON B16:31    32516490
03ED  2BFF 1830               L     NULL,MR0,D,E       SET CC, EXIT.                        32516500
               *                                       IF ERROR LOGGER STATUS BEING RETURNED32516510
               *                                       AN ERROR CAUSES L FLAG TO SET.       32516520

0000 03EE            LPSTD    EQU   *                  CODE 1 - LOAD PROCESS SEG TABLE DES  32516540
03EE  2BFF 1F99               L     NULL,NULL,LPSTD    LOAD PSTD FROM MEMORY                32516550
03EF  2BFF 1F92      EXIT36   L     NULL,NULL,IRD      EXIT.                                32516560

0000 03F0            LSSTD    EQU   *                  CODE 2 - LOAD SHARED SEG TABLE DESC  32516580
03F0  2BFF 1F98               L     NULL,NULL,LSSTD    LOAD SSTD FROM MEMORY                32516590
03F1  2BFF 1F92               L     NULL,NULL,IRD      EXIT.                                32516600

0000 03F2            STPS     EQU   *                  CODE 3 - STORE PROCESS STATE         32516620
03F2  2B7F 171F               L     WMDR,R14,DW4       STORE PROCESS' OLD PSW @+0           32516630
03F3  2B7F 179D               L     WMDR,R15,I4DW4     OLD LOC @+4                          32516640
03F4  2A9D 1F95               A     MR4,PSW,NULL,I4    SAVE EXECUTIVE PSW                   32516650
03F5  2A3D 6700               X     MR1,PSW,R14        GET PROCESS REGISTER SET             32516660
03F6  3231 50F0               NI    MR1,MR1,'F0'       ONLY THESE BITS CHANGE               32516670
03F7  2BBD 6895               X     PSW,PSW,MR1,I4     SELECT NEW SET.                      32516680
03F8  12D9 29C0               BAL   STM@(MR6)          STORE GENERAL REG SET @+12 (P.42)    32516690
03F9  2BBF 1A15               L     PSW,MR4,I4         RESELECT ENTRY SET                   32516700
03FA  37CE 513D               NI    YDI,R14,BIT14,I    INTERRUPTIBLE STATE EXISTS ? YDI=0.  32516710
```

PRIVILEGED SYSTEM FUNCTION (PSF)

```
03FB   16C1 8240          1673        BALNZ STM71(MR6)        BRANCH: YES.(P.54)                     32516720
03FC   37EE 5009          1674        NI    NULL,R14,BIT13,I   FLOATING POINT LEGAL ?                 32516730
03FD   13E0 FF92          1675        BALZ  *+1(NULL),IRD     BRANCH: YES.                           32516740
03FE   4ABF 7F80          1676        SMCR  MR5,NULL          TEST MACHINE CONTROL REGISTER          32516750
03FF   37F5 5183          1677        NI    NULL,MR5,BIT20,I   DFU EQUIPPED ?                         32516760
0400   17E1 0052          1678        BALNZ *+1(NULL),IRD     BRANCH: YES. ELSE, EXIT.               32516770
0401   12D8 7E00          1679        BAL   STME@(MR6)        STORE SPFP REGISTERS (P.17)            32516780
0402   12D9 87C0          1680        BAL   STMD@(MR6)        STORE DPFP REGISTERS (P.54)            32516790
0403   2BFF 1F92          1681 EXIT37 L     NULL,NULL,IRD     EXIT.                                  32516800

0000 0404                 1683 LDPS   EQU   *                 CODE 4 - LOAD PROCESS STATE            32516820
0404   2AFF 1E0F          1684        L     MR7,MAR,DR4       COPY BASE ADDRESS, FETCH PSW @+0       32516830
0405   2A7F 1D95          1685        L     MR3,RMDR,I4       COPY PROCESS PSW                       32516840
0406   2A1D 6995          1686        X     MR0,PSW,MR3,I4    SELECT PROCESS REGISTER SET -          32516850
0407   3210 50F0          1687        NI    MR0,MR0,'F0'      ONLY THESE BITS CHANGE                 32516860
0408   2BBD 6815          1688        X     PSW,PSW,MR0,I4    SELECT REGISTER SET.                   32516870
0409   12D9 23C0          1689        BAL   LM@(MR6)          LOAD GENERAL REG SET @+12 (P.42)       32516880
040A   37D3 513D          1690        NI    YDI,MR3,BIT14,I   INTERRUPTIBLE STATE EXISTS ? YDI=0.    32516890
040B   16C1 8380          1691        BALNZ LM71(MR6)         BRANCH: YES.(P.54)                     32516900
040C   37F3 5009          1692        NI    NULL,MR3,BIT13,I  FLOATING POINT LEGAL ?                 32516910
040D   17E1 04C0          1693        BALNZ LDPS1(NULL)       BRANCH: NO.                            32516920
040E   4ABF 7F80          1694        SMCR  MR5,NULL          TEST MACHINE CONTROL REGISTER          32516930
040F   37F5 5183          1695        NI    NULL,MR5,BIT20,I  DFU EQUIPPED ?                         32516940
0410   13E1 04C0          1696        BALZ  LDPS1(NULL)       BRANCH: NOT EQUIPPED.                  32516950
0411   12D8 5940          1697        BAL   LME@(MR6)         LOAD SPFP REGISTERS (P.13)             32516960
0412   12D9 8A80          1698        BAL   LMD@(MR6)         LOAD DPFP REGISTERS (P.54)             32516970
0000 0413                 1699 LDPS1  EQU   *                                                        32516980
0413   2B9F 1B85          1700        L     MAR,MR7,RFAULT    POINT TO PSW @+0,                  R02 32516990
0414   2BFF 1F8E          1701        L     NULL,NULL,I4DR4   READ LOC @+4 (WE HAVE PSW)         R02 32517000
0415   2B5F 1D95          1702        L     CLOC,RMDR,I4      LOAD PROCESS LOC, POINT TO PSTD @+8    32517010
0416   33F3 5400          1703        NI    NULL,MR3,'400'    TASK ENABLES MAT ?                 R03 32517020
0417   13E0 91D9          1704        BALZ  LPSW2(NULL),LPSTD BRANCH: NO. LOAD PSW.     (P.22)R03 32517030
0418   13F8 91C0          1705        BAL   LPSW2(NULL)       PSTD LOADED; GO LOAD PSW. (P.22)R03 32517040

0000 0419                 1707 ISSV   EQU   *                 CODE 5 - SAVE INTERRUPTIBLE STATE      32517060
0419   33DF 1000          1708        LI    YDI,0             START WITH REGISTER 0,                 32517070
041A   12D9 8240          1709        BAL   STM71(MR6)        STORE SCRATCHPADS (P.54)               32517080
041B   2BFF 1F92          1710        L     NULL,NULL,IRD     THEN EXIT.                             32517090

0000 041C                 1712 ISRST  EQU   *                 CODE 6 - RESTORE INTERRUPTIBLE STAT    32517110
041C   33DF 1000          1713        LI    YDI,0             START WITH REGISTER 0                  32517120
041D   12D9 8380          1714        BAL   LM71(MR6)         LOAD SCRATCHPADS (P.54)                32517130
041E   2BFF 1F92          1715        L     NULL,NULL,IRD     THEN EXIT.                             32517140

0000 041F                 1717 TEL    EQU   *                 CODE 7 - TEST ERROR LOGGER             32517160
041F   2B7F 101E          1718        L     WMDR,R0,TEL       STORE WITH NO ECC                      32517170
0420   2BFF 1F92          1719        L     NULL,NULL,IRD     EXIT.                                  32517180
```

MISCELLANEOUS

```
0421   2A7F 1E05                1721  AL1     L     MR3,MAR,RFAULT      SAVE END ADDRESS,                        32517200
                                1722  *                                 RESET RX FLOPS.                          32517210
0422   339F 1078                1723          LI    MAR,'78'                                                     32517220
0423   33F3 0080                1724          SI    NULL,MR3,'80'       IS CALCULATED END ADDRESS VALID ?        32517230
0424   13F1 180B                1725          BALC  SETCCO(NULL),DR1    IF CARRY, INVALID, ELSE (P.40)           32517240
0425   2A1F 1D8A                1726          L     MR0,RMDR,I1DR1      DEVICE ADDRESS INTO MR0.                 32517250
0426   33DF 1007                1727          LI    YDI,7               STATUS MASK                              32517260
0427   339F 107F                1728          LI    MAR,'7F'            MAR = START - 1                          32517270
0428   4BF0 BDC0                1729          OCRA  NULL,MR0,RMDR       ADDRESS DEVICE, SEND COMMAND.            32517280
0429   4BFF 2FE0                1730  AL2     SSR   NULL,NULL,E         SENSE STATUS, ADJUST CC                  32517290
042A   13ED 0AD2                1731          BALF  *+1(NULL),IRD       EXIT ON BAD STATUS                       32517300
042B   13F1 0A40                1732          BALC  AL2(NULL)           WAIT FOR BSY = 0                         32517310
042C   4A9F 0FC0                1733          RDR   MR4,NULL            READ 1ST BYTE                            32517320
042D   23FF 0A69                1734          SX    NULL,NULL,MR4,AL2,C BRANCH: A LEADING ZERO; IGNORE.          32517330
042E   2B7F 1A1A                1735          L     WMDR,MR4,I1DW1      STORE 1ST NON-ZERO BYTE                  32517340
042F   23F3 2E71                1736  AL3     SDECX NULL,MR3,MAR,AL4,C  TEST LIMITS:                             32517350
0430   2BFF 1FB2                1737          L     NULL,NULL,IRD,E     ALL DONE.                                32517360
0431   4BFF 2FE0                1738  AL4     SSR   NULL,NULL,E         TEST DEVICE STATUS                       32517370
0432   13ED 0CD2                1739          BALF  *+1(NULL),IRD       EXIT IF BAD, ELSE                        32517380
0433   13F1 0C40                1740          BALC  AL4(NULL)           WAIT FOR NOT BUSY.                       32517390
0434   4B7F 0FDA                1741          RDR   WMDR,NULL,I1DW1     INPUT & STORE SUBSEQUENT BYTE            32517400
0435   13F9 0BC0                1742          BAL   AL3(NULL)           AND LOOP.                                32517410
```

MISCELLANEOUS

```
                          1744  * COMMON SUBROUTINE FOR TBT, SBT, RBT & CBT     *                        32517430
                          1745  * MR0 = MATRIX START ADDRESS                                             32517440
                          1746  * R1 CONTAINS DISPLACEMENT TO DESIRED BIT                                32517450
0436   3219 8003          1747  COMBIT    SRLI  MR0,YD,3            ON BYTE BOUNDARY,                    32517460
0437   2B9A 1D80          1748            A     MAR,YX,RMDR         CALCULATE BASE ADDRESS               32517470
0438   2B90 1E0B          1749            A     MAR,MR0,MAR,DR1     ADDRESS ARRAY & FETCH BYTE           32517480
0439   3259 5007          1750            NI    MR2,YD,'7'          MASK LS 3 BITS TO TEST               32517490
                          1751  *                                   A BIT IN THE BYTE                    32517500
043A   3252 143F          1752            AI    MR2,MR2,BTABLE      FORM VECTOR ADDRESS                  32517510
043B   2E5F 1900          1753            L     MR2,MR2,I           FETCH BIT MASK                       32517520
043C   2A72 5DA0          1754            N     MR3,MR2,RMDR,E      TEST THE BIT, SET CC                 32517530
043D   0BF8 0B00          1755            EXL   (MR6)(NULL)         PERFORM OPERATION ON BIT OR EXIT     32517540
043E   2BFF 1F92          1756  EXIT39    L     NULL,NULL,IRD       EXIT.                                32517550

                          1758  * BIT TABLE USED BY TBT, SBT, RBT, & CBT                                 32517570
                          1759  *                                                                        32517580
0000 043F                 1760  BTABLE    EQU   *                                                        32517590
043F   0000 0080          1761  BIT24     DC    '00000080'                                               32517600
0440   0000 0040          1762  BIT25     DC    '00000040'                                               32517610
0441   0000 0020          1763  BIT26     DC    '00000020'                                               32517620
0442   0000 0010          1764  BIT27     DC    '00000010'                                               32517630
0443   0000 0008          1765  BIT28     DC    '00000008'                                               32517640
0444   0000 0004          1766  BIT29     DC    '00000004'                                               32517650
0445   0000 0002          1767  BIT30     DC    '00000002'                                               32517660
0446   0000 0001          1768  BIT31     DC    '00000001'                                               32517670


                          1770  * COMMON CONDITION CODE ADJUST ROUTINES                                  32517690
                          1771  *                                                                        32517700
0447   2BFF 1FA2          1772  SETCC4    L     NULL,NULL,IR,E      RESET CC BITS                   R02  32517710
0448   33BD 7004          1773            OI    PSW,PSW,4           SET V FLAG                           32517720
0449   2BFF 1F90          1774            L     NULL,NULL,D         CC = 4                          R02  32517730

044A   321F 1001          1776  SETCC8    LI    MR0,1               SET FOR CARRY OUT                    32517750
044B   2BF0 8832          1777            SRL   NULL,MR0,MR0,IRD,E  SET CC = 1000, EXIT.                 32517760
```

LIST INSTRUCTIONS

```
                     * ROUTINE IS COMMON PREPROCESSOR FOR ATL, ABL, RTL, RBL.
                     *
0000 044C            LIST    EQU   *
044C   2B9A 1D87                A     MAR,YX,RMDR,DR2      CALCULATE LIST ADDRESS           R02
044D   36BF 1017                LI    MR5,BI16.31,I        MR5 = '0000FFFF'                 R02
044E   2A15 5D8F                N     MR0,MR5,RMDR,DR4     MR0 = MAX SLOTS                  R02
044F   2AFF 1E05                L     MR7,MAR,RFAULT       MR7 = A(LIST); RESET RX FLOPS.   R02
0450   03F8 0B00                BAL   (MR6)(NULL)          BRANCH TO 2ND LEVEL HANDLER


0000 0451            ATL1    EQU   *
0451   3397 1004                AI    MAR,MR7,4            POINT TO CURRENT TOP
0452   2A35 5D80                N     MR1,MR5,RMDR         MR1 = SLOTS USED                 R02
0453   2BF0 0880                S     NULL,MR0,MR1         MAX SLOTS LESS SLOTS USED
0454   17E9 11C7                BALNG SETCC4(NULL),DR2     BRANCH: NO ROOM AT THE INN. (P.39)
0455   2B7F 1C80                L     WMDR,YD              DATA TO BE STORED                R02
0456   2A55 5D80                N     MR2,MR5,RMDR         MR2 = CURRENT TOP POINTER
0457   2252 2FD9                SDECX MR2,MR2,NULL,ATL.010,C BRANCH: NO LIST WRAP.
0458   3250 0001                SI    MR2,MR0,1            LIST WRAP - SET CURR TOP TO MAX.
0459   32B2 1002     ATL.010    AI    MR5,MR2,2            COMPUTE SLOT ADDRESS
045A   3395 9002                SLLI  MAR,MR5,2            .
045B   2B97 1E1F                A     MAR,MR7,MAR,DW4      ADD ELEMENT TO LIST              R02
045C   3397 1004                AI    MAR,MR7,4
045D   2B7F 1917                L     WMDR,MR2,DW2         STORE NEW CURRENT TOP
045E   3397 1002     ATL.020    AI    MAR,MR7,2
045F   2B71 3F97                AINC  WMDR,MR1,NULL,DW2    STORE NEW SLOTS USED
0460   2BFF 1FB2     SETCC0     L     NULL,NULL,IRD,E      SET CC = 0, EXIT.
```

LIST INSTRUCTIONS

```
0000 0461                    1807  ABL1     EQU   *                                                  32518060
0461    3397 1006            1808           AI    MAR,MR7,6           ADDRESS NEXT BOTTOM POINTER        32518070
0462    2A35 5D80            1809           N     MR1,MR5,RMDR        MR1 = SLOTS USED               R02 32518080
0463    2BF0 0880            1810           S     NULL,MR0,MR1        MAX SLOTS LESS SLOTS USED          32518090
0464    17E9 11C7            1811           BALNG SETCC4(NULL),DR2    BRANCH: NO ROOM AT THE INN. (P.39) 32518100
0465    2B7F 1C80            1812           L     WMDR,YD             DATA TO BE STORED              R02 32518110
0466    2A75 5D80            1813           N     MR3,MR5,RMDR        MR3 = NEXT BOTTOM POINTER          32518120
0467    32B3 1002            1814           AI    MR5,MR3,2           COMPUTE SLOT ADDRESS               32518130
0468    3395 9002            1815           SLLI  MAR,MR5,2           .                                  32518140
0469    2B97 1E1F            1816           A     MAR,MR7,MAR,DW4     ADD ELEMENT TO LIST            R02 32518150
046A    2A73 3F80            1817           AINC  MR3,MR3,NULL        INCREMENT NEXT BOTTOM              32518160
046B    23F0 29ED            1818           SDECX NULL,MR0,MR3,ABL.010,C BRANCH: NO LIST WRAP.           32518170
046C    2A7F 1F80            1819           L     MR3,NULL            LIST WRAP - SET NEXT BOTT TO 0.    32518180
046D    3397 1006            1820  ABL.010  AI    MAR,MR7,6                                              32518190
046E    2B7F 1997            1821           L     WMDR,MR3,DW2        STORE NEW NEXT BOTT                32518200
046F    13F9 1780            1822           BAL   ATL.020(NULL)       GO UPDATE SLOTS USED    (P.40)     32518210


0000 0470                    1824  RTL1     EQU   *                                                  32518230
0470    3397 1004            1825           AI    MAR,MR7,4           READ CURRENT TOP                   32518240
0471    2A35 5D80            1826           N     MR1,MR5,RMDR        MR1 = SLOTS USED               R02 32518250
0472    13E1 11C7            1827           BALZ  SETCC4(NULL),DR2    BRANCH: NO SLOTS USED (P.39)   R02 32518260
0473    2A55 5D80            1828           N     MR2,MR5,RMDR        MR2 = CURRENT TOP POINTER          32518270
0474    32B2 1002            1829           AI    MR5,MR2,2           CALCULATE SLOT ADDRESS             32518280
0475    3395 9002            1830           SLLI  MAR,MR5,2           .                                  32518290
0476    2B97 1E0F            1831           A     MAR,MR7,MAR,DR4     READ LIST ELEMENT                  32518300
0477    2A52 3F80            1832           AINC  MR2,MR2,NULL        INCREMENT CURR TOP                 32518310
0478    23F0 297A            1833           SDECX NULL,MR0,MR2,RTL.010,C BRANCH: NO LIST WRAP.           32518320
0479    2A5F 1F80            1834           L     MR2,NULL            LIST WRAP - SET CURR TOP TO 0.     32518330
047A    3397 1004            1835  RTL.010  AI    MAR,MR7,4                                              32518340
047B    2B7F 1917            1836           L     WMDR,MR2,DW2        STORE NEW CURRENT TOP              32518350
047C    3397 1002            1837  RTL.020  AI    MAR,MR7,2                                              32518360
047D    2B71 2FB7            1838           SDEC  WMDR,MR1,NULL,DW2,E STORE NEW SLOTS USED, UPDATE CC    32518370
047E    2B3F 1D92            1839           L     YD,RMDR,IRD         COPY DATA TO YD, EXIT.             32518380


0000 047F                    1841  RBL1     EQU   *                                                  32518400
047F    3397 1006            1842           AI    MAR,MR7,6           READ NEXT BOTTOM                   32518410
0480    2A35 5D80            1843           N     MR1,MR5,RMDR        MR1 = SLOTS USED               R02 32518420
0481    13E1 11C7            1844           BALZ  SETCC4(NULL),DR2    BRANCH: NO SLOTS USED (P.39)   R02 32518430
0482    2A75 5D80            1845           N     MR3,MR5,RMDR        MR3 = NEXT BOTTOM POINTER          32518440
0483    2273 2FC5            1846           SDECX MR3,MR3,NULL,RBL.010,C BRANCH: NO LIST WRAP            32518450
0484    3270 0001            1847           SI    MR3,MR0,1           LIST WRAP - SET NEXT BOTT TO MAX.  32518460
0485    32B3 1002            1848  RBL.010  AI    MR5,MR3,2           COMPUTE SLOT ADDRESS               32518470
0486    3395 9002            1849           SLLI  MAR,MR5,2           .                                  32518480
0487    2B97 1E0F            1850           A     MAR,MR7,MAR,DR4     READ LIST ELEMENT                  32518490
0488    3397 1006            1851           AI    MAR,MR7,6                                              32518500
0489    2B7F 1997            1852           L     WMDR,MR3,DW2        STORE NEW NEXT BOTTOM              32518510
048A    13F9 1F00            1853           BAL   RTL.020(NULL)       GO UPDATE SLOTS USED               32518520
```

LOAD/STORE MULTIPLE GENERAL REGISTERS

```
048B   2B9A 1D80         1855  LM1      A     MAR,YX,RMDR        CALCULATE ADDRESS               32518540
048C   2AFF 1E0F         1856           L     MR7,MAR,DR4        SAVE FOR FAULT RECOVERY    R02  32518550
048D   321F 1490         1857           LI    MR0,LMTAB          BASE ADDRESS OF TABLE           32518560
048E   2210 1F25         1858           AX    MR0,MR0,YDI,LM2    CALCULATE ENTRY.                32518570

048F   2BFF 1F8F         1860  LM@      L     NULL,NULL,DR4      HERE FOR 16 LOADS.              32518590
                         1861  *                                                                 32518600
0490   281F 1D8E         1862  LMTAB    L     R0,RMDR,I4DR4                                      32518610
0491   283F 1D8E         1863           L     R1,RMDR,I4DR4                                      32518620
0492   285F 1D8E         1864           L     R2,RMDR,I4DR4                                      32518630
0493   287F 1D8E         1865           L     R3,RMDR,I4DR4                                      32518640
0494   289F 1D8E         1866           L     R4,RMDR,I4DR4                                      32518650
0495   28BF 1D8E         1867           L     R5,RMDR,I4DR4                                      32518660
0496   28DF 1D8E         1868           L     R6,RMDR,I4DR4                                      32518670
0497   28FF 1D8E         1869           L     R7,RMDR,I4DR4                                      32518680
0498   291F 1D8E         1870           L     R8,RMDR,I4DR4                                      32518690
0499   293F 1D8E         1871           L     R9,RMDR,I4DR4                                      32518700
049A   295F 1D8E         1872           L     R10,RMDR,I4DR4                                     32518710
049B   297F 1D8E         1873           L     R11,RMDR,I4DR4                                     32518720
049C   299F 1D8E         1874           L     R12,RMDR,I4DR4                                     32518730
049D   29BF 1D8E         1875           L     R13,RMDR,I4DR4                                     32518740
049E   29DF 1D8E         1876           L     R14,RMDR,I4DR4                                     32518750
049F   29FF 1D95         1877           L     R15,RMDR,I4                                        32518760
04A0   03F8 0B00         1878           BAL   (MR6)(NULL)                                        32518770
0000 04A1                1879  LMTABE   EQU   *                  USED FOR FAULT DECODE           32518780

04A1   2B7F 1C80         1881  STM1     L     WMDR,YD            FIRST DATA TO STORE             32518800
04A2   2B9A 1D9F         1882           A     MAR,YX,RMDR,DW4    CALCULATE ADDRESS, STORE.  R02  32518810
04A3   321F 14A8         1883           LI    MR0,STMTAB         TABLE BASE ADDRESS              32518820
04A4   2A10 1F00         1884           A     MR0,MR0,YDI        COMPUTE ENTRY                   32518830
04A5   02D8 0800         1885  LM2      BAL   (MR0)(MR6)                                         32518840
04A6   2BFF 1F92         1886  EXIT42   L     NULL,NULL,IRD      EXIT.                           32518850

0000 04A7                1888  STM@     EQU   *                  HERE FOR 16 STORES.             32518870
04A7   2B7F 101F         1889           L     WMDR,R0,DW4        STORE FIRST DATA                32518880
                         1890  *                                                                 32518890
04A8   2B7F 109D         1891  STMTAB   L     WMDR,R1,I4DW4                                      32518900
04A9   2B7F 111D         1892           L     WMDR,R2,I4DW4                                      32518910
04AA   2B7F 119D         1893           L     WMDR,R3,I4DW4                                      32518920
04AB   2B7F 121D         1894           L     WMDR,R4,I4DW4                                      32518930
04AC   2B7F 129D         1895           L     WMDR,R5,I4DW4                                      32518940
04AD   2B7F 131D         1896           L     WMDR,R6,I4DW4                                      32518950
04AE   2B7F 139D         1897           L     WMDR,R7,I4DW4                                      32518960
04AF   2B7F 141D         1898           L     WMDR,R8,I4DW4                                      32518970
04B0   2B7F 149D         1899           L     WMDR,R9,I4DW4                                      32518980
04B1   2B7F 151D         1900           L     WMDR,R10,I4DW4                                     32518990
04B2   2B7F 159D         1901           L     WMDR,R11,I4DW4                                     32519000
04B3   2B7F 161D         1902           L     WMDR,R12,I4DW4                                     32519010
04B4   2B7F 169D         1903           L     WMDR,R13,I4DW4                                     32519020
04B5   2B7F 171D         1904           L     WMDR,R14,I4DW4                                     32519030
04B6   2B7F 179D         1905           L     WMDR,R15,I4DW4                                     32519040
04B7   03F8 0B00         1906           BAL   (MR6)(NULL)                                        32519050
```

MISCELLANEOUS

```
04B8   3658 5075          1908  CHVR1   NI    MR2,YS,BI17.31,I    CAPTURE SIGNIFICANCE                        32519070
04B9   3678 5019          1909          NI    MR3,YS,BIT16,I      AND HALFWORD SIGN BIT.                      32519080
04BA   2A72 0980          1910          S     MR3,MR2,MR3         EXTEND SIGN IN MR3                          32519090
04BB   2A33 6C02          1911          X     MR1,MR3,YS,IR       RECREATE HALFWORD OVERFLOW BIT              32519100
04BC   37F1 500F          1912          NI    NULL,MR1,BIT15,I    BY CHECKING B15 OF DATA AND RESULT          32519110
04BD   13E1 2FC0          1913          BALZ  CHVR2(NULL)                                                     32519120
04BE   3210 7004          1914          OI    MR0,MR0,4           OVERFLOW                                    32519130
04BF   2B3F 19A0          1915  CHVR2   L     YD,MR3,E            LOAD YD, ADJUST G & L FLAGS                 32519140
04C0   2BBD 7810          1916          O     PSW,PSW,MR0,D       OR IN C & V STATES, EXIT.                   32519150


04C1   2A1F 3F00          1918  BXLE1   AINC  MR0,NULL,YDI        MR0 POINTS TO R1+1                          32519170
04C2   2BDF 1800          1919          L     YDI,MR0             POINT TO R1+1                               32519180
04C3   2A3F 1C80          1920          L     MR1,YD              MR1 = INCREMENT                             32519190
04C4   2BD0 3F80          1921          AINC  YDI,MR0,NULL        POINT TO R1+2                               32519200
04C5   2A5F 1C80          1922          L     MR2,YD              MR2 = COMPARAND                             32519210
04C6   2BD0 2F80          1923          SDEC  YDI,MR0,NULL        POINT TO R1                                 32519220
04C7   2A11 1C80          1924          A     MR0,MR1,YD          MR0 = OLD R1 + INCREMENT                    32519230
04C8   23F0 2953          1925          SDECX NULL,MR0,MR2,BXLE3,C BRANCH: MR0 > COMPARAND                    32519240
04C9   2B9A 1D80          1926  BXLE2   A     MAR,YX,RMDR         CALCULATE BRANCH ADDRESS                    32519250
04CA   235F 1E13          1927          LX    CLOC,MAR,BXLE3      LOAD NEW LOC                                32519260


04CB   2A1F 3F00          1929  BXH1    AINC  MR0,NULL,YDI        MR0 POINTS TO R1+1                          32519280
04CC   2BDF 1800          1930          L     YDI,MR0             POINT TO R1+1                               32519290
04CD   2A3F 1C80          1931          L     MR1,YD              MR1 = INCREMENT                             32519300
04CE   2BD0 3F80          1932          AINC  YDI,MR0,NULL        POINT TO R1+2                               32519310
04CF   2A5F 1C80          1933          L     MR2,YD              MR2 = COMPARAND                             32519320
04D0   2BD0 2F80          1934          SDEC  YDI,MR0,NULL        POINT TO R1                                 32519330
04D1   2A11 1C80          1935          A     MR0,MR1,YD          MR0 = OLD R1 + INCREMENT                    32519340
04D2   23F0 2949          1936          SDECX NULL,MR0,MR2,BXLE2,C BRANCH: MR0 > COMPARAND                    32519350
04D3   2B3F 1812          1937  BXLE3   L     YD,MR0,IRD          YD = NEW VALUE; EXIT.                       32519360


04D4   325F 1010          1939  AHM1    LI    MR2,16              SHIFT COUNT                            R02  32519380
04D5   2A19 9900          1940          SLL   MR0,YD,MR2          YD LEFT 16 FOR FLAGS                   R02  32519390
04D6   2A3F 1D80          1941          L     MR1,RMDR            .                                      R02  32519400
04D7   2A31 9900          1942          SLL   MR1,MR1,MR2         DATA LEFT 16                           R02  32519410
04D8   2A31 1820          1943          A     MR1,MR1,MR0,E       ADD, SET FLAGS                         R02  32519420
04D9   2B71 8917          1944          SRL   WMDR,MR1,MR2,DW2    SUM RIGHT 16, STORE.                   R02  32519430
04DA   2BFF 1F92          1945  EXIT43  L     NULL,NULL,IRD       EXIT.                                       32519440
```

MISCELLANEOUS

```
04DB    32D9 50FF          1947  TLATE1    NI    MR6,YD,'FF'          BYTE TO TRANSLATE                        32519460
04DC    2AD6 1B05          1948            A     MR6,MR6,MR6,RFAULT   2X THE BYTE PLUS ADRS                    32519470
                           1949  *                                    (RESET RX FLOPS)                         32519480
04DD    2B96 1D87          1950            A     MAR,MR6,RMDR,DR2     OF TRANSLATION TABLE                     32519490
04DE    3239 5F00          1951            NI    MR1,YD,'F00'         FETCH HALFWORD ENTRY                     32519500
04DF    2ADF 1D80          1952            L     MR6,RMDR                                                      32519510
04E0    17E5 388A          1953            BALNL TLATE2(NULL),I1DR1   EXIT IF NOT NEGATIVE                     32519520
04E1    2B31 7D92          1954            O     YD,MR1,RMDR,IRD      OR INTO R1, EXIT.                        32519530
04E2    2356 1B1A          1955  TLATE2    AX    CLOC,MR6,MR6,EXIT43  EXECUTE AT ROUTINE.                  R02 32519540


04E3    229F 1DA7          1957  MH1       LX    MR4,RMDR,MH2         GET MULTIPLIER                           32519560
04E4    3658 5075          1958  MHR1      NI    MR2,YS,BI17.31,I     EXTRACT SIGNIFICANCE                     32519570
04E5    3618 5019          1959            NI    MR0,YS,BIT16,I       GET HALFWORD SIGN BIT                    32519580
04E6    2A92 0800          1960            S     MR4,MR2,MR0          AND EXTEND                               32519590
04E7    3679 5075          1961  MH2       NI    MR3,YD,BI17.31,I     EXTRACT SIGNIFICANCE                     32519600
04E8    3619 5019          1962            NI    MR0,YD,BIT16,I       GET HALFWORD SIGN BIT                    32519610
04E9    2A73 0802          1963            S     MR3,MR3,MR0,IR       AND EXTEND                               32519620
04EA    2A53 EA00          1964            M     MR2,MR3,MR4          MULTIPLY                                 32519630
04EB    2B3F 1990          1965            L     YD,MR3,D             LS 32 BIT PRODUCT TO R1; EXIT.           32519640


04EC    229F 1DB0          1967  DH1       LX    MR4,RMDR,DH2         GET DIVISOR                              32519660
04ED    3658 5075          1968  DHR1      NI    MR2,YS,BI17.31,I     EXTRACT SIGNIFICANCE                     32519670
04EE    3618 5019          1969            NI    MR0,YS,BIT16,I       GET HALFWORD SIGN BIT                    32519680
04EF    2A92 0800          1970            S     MR4,MR2,MR0          EXTEND IN MR4                            32519690
04F0    13E0 97C0          1971  DH2       BALZ  AFAUL0(NULL)         BRANCH: DIV-BY-ZERO. (P.23)              32519700
04F1    2A5F 1F80          1972            L     MR2,NULL                                                      32519710
04F2    2A7F 1C80          1973            L     MR3,YD               MR3=DIVIDEND                             32519720
04F3    17E5 3D40          1974            BALNL *+2(NULL)                                                     32519730
04F4    2A5F 2F80          1975            SDEC  MR2,NULL,NULL        MR2 = SIGN OF DIVIDEND                   32519740
04F5    2A53 FA00          1976            D     MR2,MR3,MR4          DO DIVIDE                                32519750
04F6    13F4 9800          1977            BALV  AFAUL1(NULL)         BRANCH: QUOTIENT OVERFLOW. (P.23)        32519760
04F7    3693 5053          1978            NI    MR4,MR3,BI00.16,I    CAPTURE SIGN/EXTENDED SIGN               32519770
04F8    3613 5019          1979            NI    MR0,MR3,BIT16,I      CAPTURE HALFWORD SIGN BIT                32519780
04F9    2BF4 1800          1980            A     NULL,MR4,MR0         ALL BITS ALIKE ?                         32519790
04FA    17E0 9800          1981            BALNZ AFAUL1(NULL)         BRANCH: QUOTIENT OVERFLOW. (P.23)        32519800
04FB    2B3F 1902          1982            L     YD,MR2,IR            REMAINDER TO R1                          32519810
04FC    2BDF 3F00          1983            AINC  YDI,NULL,YDI         COMPUTE R1+1                             32519820
04FD    2B3F 1990          1984            L     YD,MR3,D             QUOTIENT TO R1+1; EXIT.                  32519830
```

MISCELLANEOUS

```
0000 04FE              1986  LRA1     EQU   *                     LOAD REAL ADDRESS                   32519850
04FE   36F9 5055       1987           NI    MR7,YD,BI08.15,I      = PRESENTED SEGMENT NUMBER          32519860
04FF   2A57 1B80       1988           A     MR2,MR7,MR7           FOR SEGMENT NUMBER ALIGN        R02 32519870
0500   2A1F 1D8E       1989           L     MR0,RMDR,I4DR4        MR0 = PSTD; FETCH SHARED TABLE DES  32519880
0501   36D0 513B       1990           NI    MR6,MR0,BI02.14,I     = PROCESS SEGMENT TAB SIZE - 1  R02 32519890
0502   2A3F 1D85       1991           L     MR1,RMDR,RFAULT       MR1 = SSTD; RESET RX FLOPS.     R02 32519900
0503   23F6 0945       1992           SX    NULL,MR6,MR2,LRA2,C TEST IF IN TABLE:                 32519910
0504   13F9 1280       1993  LRA1.5   BAL   SETCC8(NULL)          TABLE SIZE EXCEEDED; UNMAPPED(P.39) 32519920
0505   3257 800D       1994  LRA2     SRLI  MR2,MR7,13            CREATE OFFSET                       32519930
0506   367F 1057       1995           LI    MR3,BI15.31,I         MASK = '0001FFFF'                   32519940
0507   2AD0 5980       1996           N     MR6,MR0,MR3           GET CODED SEG TAB ADRS              32519950
0508   32D6 9007       1997           SLLI  MR6,MR6,7             AND DECODE                          32519960
0509   2B96 190F       1998           A     MAR,MR6,MR2,DR4       ADD OFFSET, FETCH HSTE              32519970
050A   2A9F 1D80       1999           L     MR4,RMDR              MR4 = PROCESS HSTE                  32519980
050B   37F4 504F       2000  LRA3     NI    NULL,MR4,BIT01,I      PRESENCE BIT SET ?                  32519990
050C   13E1 11C0       2001           BALZ  SETCC4(NULL)          BRANCH: NOT PRESENT. (P.39)         32520000
050D   37F4 5059       2002           NI    NULL,MR4,BIT08,I      SHARED ?                            32520010
050E   13E1 4600       2003           BALZ  LRA.PRI(NULL)         BRANCH: CAN EVALUATE AS PRIVATE.    32520020
                       2004  *                                                                        32520030
050F   32B1 800E       2005  LRA.SHAR SRLI  MR5,MR1,14            GET SST SIZE                    R03 32520040
0510   2AF4 5980       2006           N     MR7,MR4,MR3           SRF BECOMES SST OFFSET          R03 32520050
0511   23F7 2AC4       2007           SDECX NULL,MR7,MR5,LRA1.5,C BRANCH: PST SIZE EXCEEDED.  R03     32520060
0512   2AD1 5980       2008           N     MR6,MR1,MR3           GET ENCODED SST ADDRESS             32520070
0513   32D6 9007       2009           SLLI  MR6,MR6,7             AND DECODE.                         32520080
0514   2B96 1B8F       2010           A     MAR,MR6,MR7,DR4       FETCH SHARED HSTE                   32520090
0515   3694 700B       2011           OI    MR4,MR4,BI03.050,I    SET ALL BUT ACCESS MODE BITS    R01 32520100
0516   3694 5065       2012           NI    MR4,MR4,BIT080,I      ZERO S BIT                      R01 32520110
0517   2294 5D8B       2013           NX    MR4,MR4,RMDR,LRA3     "AND" ACCESS KEYS WITH SST HSTE R02 32520120
                       2014  *                                                                        32520130
0518   3614 5013       2015  LRA.PRI  NI    MR0,MR4,BI10.14,I     MASK SEG LIMIT FIELD FROM STE       32520140
0519   3659 5071       2016           NI    MR2,YD,BI16.20,I      EXTRACT SEGMENT FIELD           R01 32520150
051A   3210 8006       2017           SRLI  MR0,MR0,6             .                               R01 32520160
051B   23F0 095D       2018           SX    NULL,MR0,MR2,LRA.PR2,C BRANCH: ADDRESS NOT > LIMIT.       32520170
051C   13F9 1280       2019           BAL   SETCC8(NULL)          BRANCH: LIMIT VIOLATION (P.39)      32520180
051D   2A14 59A0       2020  LRA.PR2  N     MR0,MR4,MR3,E         GET SEG RELOC FIELD             R03 32520190
051E   3210 9007       2021           SLLI  MR0,MR0,7             SCALED - MULTIPLY BY 2**7           32520200
051F   3739 5017       2022           NI    YD,YD,BI16.31,I       GET LEAST SIGNIFICANT HALF OF ADDRESS32520210
0520   2339 183A       2023           AX    YD,YD,MR0,LRA.PR3     TRANSLATE. E-BIT LATENCY ON.    R03 32520220

0521   0800 0000       2025  BIT04    DC    '08000000'            CONSTANT                        R03 32520240
0522   33BD 7001       2026  ADDCC1   OI    PSW,PSW,1             TURN ON L FLAG                  R03 32520250
0523   2BFF 1F92       2027           L     NULL,NULL,IRD         EXIT, CC SET.                   R03 32520260
```

MISCELLANEOUS

```
0524                     2029          ORG   '524'              .                                   R03  32520280
0524   2A5F 1E05         2030  SCP1    L     MR2,MAR,RFAULT     MR2 = A(CCW); RESET RX FLOPS        R02  32520290
0525   2A7F 1D80         2031          L     MR3,RMDR           MR3 = CCW                                32520300
0526   32B3 5008         2032          NI    MR5,MR3,BBIT       MR5 = 0 OR 8 (BUFFER BIT)                32520310
0527   32B5 7002         2033          OI    MR5,MR5,2          MR5 = '02' OR '0A'                       32520320
0528   2B95 1907         2034  SCP2    A     MAR,MR5,MR2,DR2    FETCH BUFFER BYTE COUNT                  32520330
0529   2A9F 1D80         2035          L     COUNT,RMDR         IF COUNT IS POSITIVE                     32520340
052A   13E9 11C0         2036          BALG  SETCC4(NULL)       SET V FLAG & EXIT (P.39)                 32520350
052B   2AF4 3F80         2037          AINC  MR7,COUNT,NULL     INCREMENT COUNT                          32520360
052C   2B7F 1BB7         2038          L     WMDR,MR7,DW2,E     STORE IT, SET CC:                        32520370
                         2039  *                                IF POSITIVE, CC = 2                      32520380
                         2040  *                                IF ZERO      CC = 0                      32520390
                         2041  *                                IF NEGATIVE  CC = 1                      32520400
052D   17E9 4C80         2042          BALNG SCP3(NULL)         BRANCH: NOT YET AT BUFFER LIMIT.         32520410
052E   33F3 5001         2043          NI    NULL,MR3,FBIT      FAST MODE ?                              32520420
052F   17E1 4C80         2044          BALNZ SCP3(NULL)         BRANCH: YES.                             32520430
0530   3373 6008         2045          XI    WMDR,MR3,BBIT      COMPLEMENT BUFFER BIT                    32520440
0531   2B9F 1917         2046          L     MAR,MR2,DW2        RESTORE CCW                              32520450
0532   3395 1002         2047  SCP3    AI    MAR,MR5,2                                                   32520460
0533   2B92 1E0F         2048          A     MAR,MR2,MAR,DR4    FETCH BUFFER END ADRS                    32520470
0534   2B94 1D80         2049          A     MAR,COUNT,RMDR     ADD COUNT                                32520480
0535   33F3 5004         2050          NI    NULL,MR3,RWBIT     TEST R/W BIT                             32520490
0536   13E1 4E0B         2051          BALZ  SCP4(NULL),DR1     BRANCH: R/W = 1 = WRITE                  32520500
0537   2B3F 1D92         2052          L     YD,RMDR,IRD        .       R/W = 0 = READ .                 32520510
                         2053  *                                                                         32520520
0538   2B7F 1C9B         2054  SCP4    L     WMDR,YD,DW1        STORE BYTE                               32520530
0539   2BFF 1F92         2055  EXIT46  L     NULL,NULL,IRD      EXIT.                                    32520540
                         2056  *                                NOTE:  BUFFER 1 MAY BE USED, FAST        32520550
                         2057  *                                MODE. HOWEVER, BUFFERS NOT SWITCHED.     32520560

053A   3294 6FFF         2059  LRA.PR3 XI    MR4,MR4,-1         CHANGE ACCESS PRIVS TO PROT KEYS R03     32520580
053B   37F4 5521         2060          NI    NULL,MR4,BIT04,I   SET G FLAG IF WRITE-PROTECTED       R03  32520590
053C   2BFF 1F80         2061          L     NULL,NULL          TURN OFF E-BIT LATENCY              R03  32520600
053D   37F4 553F         2062          NI    NULL,MR4,BI0305,I  TEST READ, XEQ PROTECT              R03  32520610
053E   17E1 4892         2063          BALNZ ADDCC1(NULL),IRD   BRANCH: READ OR XEQ PROTECTED.           32520620
053F   1400 0000         2064  BI0305  DC    '14000000'         CONSTANT                            R03  32520630


                         2069          ENDC                                                              32520680
```

MISCELLANEOUS

```
0540    32D9 503F             2071  CRC121   NI    MR6,YD,'3F'         MASK 6 BITS                32520700
0541    363F 1121             2072           LI    MR1,C0F01,I         POLY CHECK                 32520710
0542    2AD6 6D80             2073           X     MR6,MR6,RMDR        XOR IN RESIDUAL            32520720
0543    36D6 5017             2074           NI    MR6,MR6,BI16.31,I   MASK LS 16 BITS            32520730
0544    32FF 1001             2075           LI    MR7,1                                          32520740
0545    12B9 5480             2076           BAL   CRC12B(RETURN)                                 32520750
0546    2BFF 1F92             2077  EXIT47   L     NULL,NULL,IRD                                  32520760

0547    3279 50FF             2079  CRC161   NI    DAT,YD,'FF'         MASK 8 BITS                32520780
0548    12B9 5280             2080           BAL   CRC16B(RETURN)      TO COMMON ROUTINE          32520790
0549    2BFF 1F92             2081           L     NULL,NULL,IRD                                  32520800

                              2083  * SUBROUTINE SHARED BY AUTO DRIVER CHANNEL                32520820
                              2084  *                                                         32520830
054A    363F 1123             2085  CRC16B   LI    MR1,CA001,I         POLY CHECK                 32520840
054B    2AD3 6D80             2086           X     MR6,DAT,RMDR        XOR IN RESIDUAL            32520850
054C    36D6 5017             2087           NI    MR6,MR6,BI16.31,I   MASK LS 16 BITS            32520860
054D    32FF 1001             2088           LI    MR7,1                                          32520870
                              2089  *                                                         32520880
054E    22D6 8BD0             2090           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520890
054F    2AD6 6880             2091           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32520900
0550    22D6 8BD2             2092           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520910
0551    2AD6 6880             2093           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32520920
0552    22D6 8BD4             2094  CRC12B   SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520930
0553    2AD6 6880             2095           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32520940
0554    22D6 8BD6             2096           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520950
0555    2AD6 6880             2097           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32520960
0556    22D6 8BD8             2098           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520970
0557    2AD6 6880             2099           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32520980
0558    22D6 8BDA             2100           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32520990
0559    2AD6 6880             2101           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32521000
055A    22D6 8BDC             2102           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32521010
055B    2AD6 6880             2103           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32521020
055C    22D6 8BDE             2104           SRLX  MR6,MR6,MR7,*+2,C   DATA & RESIDUAL EQUAL?     32521030
055D    2AD6 6880             2105           X     MR6,MR6,MR1         YES, XOR IN FEEDBACK       32521040
                              2106  *                                                         32521050
055E    2B7F 1B17             2107           L     WMDR,MR6,DW2        STORE RESULT               32521060
055F    03F8 0A80             2108           BAL   (RETURN)(NULL)      RETURN                     32521070
```

POWER FAIL/RESTORE ROUTINES

```
                                  2110  * ************************************************************         32521090
                                  2111  *                                                                         32521100
                                  2112  *          PRIMARY POWER FAIL ( POWER DOWN ) SEQUENCE                     32521110
                                  2113  *                                                                         32521120
                                  2114  * ************************************************************         32521130

0000 0560                         2116  PWRDWN   EQU   *                      PPF HAS OCCURRED                    32521150
0560    4AFF 7F85                 2117  PPFINT   SMCR  MR7,NULL,RFAULT        RESET FAULTS, COLLECT CONTROL FLAGS 32521160
0561    339F 1084                 2118           LI    MAR,'84'               ADRS OF CURRENT PSW SAVE POINTER    32521170
0562    2BFF 1F8C                 2119           L     NULL,NULL,PR4          FETCH POINTER                       32521180
0563    321F 1FFC                 2120           LI    MR0,'FFC'                                                  32521190
0564    2B70 5D9C                 2121           N     WMDR,MR0,RMDR,PW4      FORCE ALIGNMENT                     32521200
0565    2B90 5D80                 2122           N     MAR,MR0,RMDR           LOAD ALIGNED ADDRESS                32521210
0566    2B7D 1F9C                 2123           A     WMDR,PSW,NULL,PW4      STORE PSW @ +0                      32521220
0567    2BBF 1F95                 2124           L     PSW,NULL,I4            SELECT REG SET 0                    32521230
0568    2B7F 1D1F                 2125           L     WMDR,ILOC,DW4          SAVE ILOC @+4                       32521240
0569    2BDF 1F95                 2126           L     YDI,NULL,I4            YDI = 0                             32521250
                                  2127  *                                                                         32521260
056A    12D9 29C0                 2128  UNLOAD.1 BAL   STM@(MR6)              STORE GEN REGISTER SET (P.42)       32521270
056B    2BFF 1F95                 2129           L     NULL,NULL,I4           READY FOR NEXT                      32521280
056C    33BD 1010                 2130           AI    PSW,PSW,'010'          INCREMENT REGISTER SET NUMBER       32521290
056D    33FD 5080                 2131           NI    NULL,PSW,'80'          TEST IF LAST SET STORED             32521300
056E    13E1 5A80                 2132           BALZ  UNLOAD.1(NULL)         LOOP FOR ALL GENERAL SETS           32521310
                                  2133  *                                                                         32521320
056F    12D9 8240                 2134  UNLOAD.2 BAL   STM71(MR6)             STORE SCRATCHPADS (P.54)            32521330
0570    37F7 5183                 2135           NI    NULL,MR7,BIT20,I       TEST MCR BIT 4                      32521340
0571    13E1 5D00                 2136           BALZ  UNLOAD.3(NULL)         SKIP IF NO FPP                      32521350
0572    12D8 7E00                 2137           BAL   STME@(MR6)             STORE SPFP REGISTERS (P.17)         32521360
0573    12D9 87C0                 2138           BAL   STMD@(MR6)             STORE DPFP REGISTERS (P.54)         32521370
0000 0574                         2139  UNLOAD.3 EQU   *                                                          32521380
0574    17FD 5D00                 2140  POW      BALD  *(NULL)                WAIT FOR POWER DOWN.                32521390
```

POWER FAIL/RESTORE ROUTINES

```
                   * ***********************************************************
                   *
                   *          POWER RESTORE SEQUENCE
                   *
                   * ***********************************************************

                   * IT IS ASSUMED THAT THE OPERATING SYSTEM WILL TAKE CARE
                   * OF MAINTAINING PROCESS AND SHARED SEGMENT TABLE DESCRIPTORS.

0000 0575          SELFTEST EQU   *                   BASIC PROCESSOR DIAGNOSTIC
0000 0575          SELFTST1 EQU   *                   BACK HERE ON FAILURE -
0575   2BDF 2FA0            SDEC  YDI,NULL,NULL,E     SET ALL T/F MASK BITS
0576   17E5 5D40            BALNL SELFTST1(NULL)
0577   17F1 5D40            BALNC SELFTST1(NULL)
0578   13ED 5D40            BALF  SELFTST1(NULL)
0579   13F5 5D40            BALV  SELFTST1(NULL)
057A   13E9 5D40            BALG  SELFTST1(NULL)

057B   361F 107B            LI    MR0,FIVES,I         ='55555555'
057C   3630 6079            XI    MR1,MR0,TENS,I      ='FFFFFFFF'
057D   2A51 0800            S     MR2,MR1,MR0         ='AAAAAAAA'
057E   3672 007B            SI    MR3,MR2,FIVES,I     ='55555555'
057F   3693 7079            OI    MR4,MR3,TENS,I      ='FFFFFFFF'
0580   2AB4 0900            S     MR5,MR4,MR2         ='55555555'
0581   2AD5 3A00            AINC  MR6,MR5,MR4         ='55555555'
0582   2AF6 2880            SDEC  MR7,MR6,MR1         ='55555555'
0583   2BB7 6800            X     PSW,MR7,MR0         ='00000000'
0584   17E1 5D40            BALNZ SELFTST1(NULL)      NO FLAGS SHOULD BE SET.          R03
                   *
0585   4ABF 7F80            SMCR  MR5,NULL            COLLECT CONTROL FLAGS
0586   13F5 6800            BALV  COLDSTRT(NULL)      BRANCH: MEMORY POWER WAS LOST (P.50)

0000 0587          WARMSTRT EQU   *                   MEMORY POWER NOT LOST
0587   12F9 6E40            BAL   MEMTEST(MR7)        TEST BASIC MEMORY (P.50)
                   *
0588   339F 1084   RELOAD   LI    MAR,'84'            MAR = A(REGISTER SAVE POINTER)
0589   2BBF 1F8C            L     PSW,NULL,PR4        FETCH SAVE POINTER
058A   321F 1FFC            LI    MR0,'FFC'           .                                R02
058B   2B90 5D8F            N     MAR,MR0,RMDR,DR4    ALIGN ADDRESS, FETCH PSW         R02
058C   2AFF 1D8E   RELOADB  L     MR7,RMDR,I4DR4      HOLD PSW TEMPORARILY
058D   2B5F 1D95            L     CLOC,RMDR,I4        LOAD POWERDOWN LOC
                   *
058E   12D9 23C0   WARM.1   BAL   LM@(MR6)            LOAD GENERAL REGISTER SET (P.42)
058F   33BD 1010            AI    PSW,PSW,'10'        INCREMENT REGISTER SET NUMBER
0590   33FD 5080            NI    NULL,PSW,'80'       TEST IF LAST SET LOADED
0591   13E1 6380            BALZ  WARM.1(NULL)        LOOP UNTIL ALL SETS LOADED
0592   12D9 8380            BAL   LM71(MR6)           LOAD SCRATCHPAD REGISTERS (P.54)
                   *
0593   37F5 5183            NI    NULL,MR5,BIT20,I    TEST MCR BIT 4
0594   13E1 65C0            BALZ  WARM.2(NULL)        SKIP IF NO DFU
0595   12D8 5940            BAL   LME@(MR6)           LOAD SPFP REGISTERS (P.13)
0596   12D9 8A80            BAL   LMD@(MR6)           LOAD DPFP REGISTERS (P.54)
                   *
```

POWER FAIL/RESTORE ROUTINES

```
0597   4BFF 7FD1        2195 WARM.2   CMCR  NULL,NULL,@LOC        TURN FAULT LAMP OFF, UPDATE ILOC.    32521940
0598   2BBF 1B80        2196          L     PSW,MR7               LOAD RUNNING PSW                     32521950
0599   12F9 7380        2197          BAL   TLSU(MR7)             TEST IF LSU ENABLED (P.51)           32521960
059A   339F 1028        2198          LI    MAR,'28'              A(CONSOLE STATUS)                    32521970
059B   2BFF 1F86        2199          L     NULL,NULL,PR2         FETCH STATUS                         32521980
059C   2BFF 1D80        2200          L     NULL,RMDR             WAS IN CONSOLE MODE ?                32521990
059D   13E4 C000        2201          BALL  CONSER(NULL)          BRANCH: YES. GO BACK TO IT. (P.30)   32522000
059E   327F 1001        2202 PWRUPINT LI    MR3,1                 SET FOR CODE '40000000'              32522010
059F   17FC 8C00        2203          BALD  MMFINT(NULL)          TEST IF MMFINT ENABLED (P.20)        32522020

0000 05A0               2205 COLDSTRT EQU   *                     MEMORY POWER WAS LOST.               32522040
05A0   339F 1000        2206          LI    MAR,0                 START WITH ZERO                      32522050
05A1   363F 1009        2207          LI    MR1,BIT13,I           ='00040000'                          32522060
05A2   327F 1004        2208          LI    MR3,4                 INCREMENT                            32522070
05A3   2B7F 1E1C        2209 COLD.1   L     WMDR,MAR,PW4          WRITE ADDRESS TO CELL                32522080
05A4   2B93 1E00        2210          A     MAR,MR3,MAR           ADVANCE TO NEXT                      32522090
05A5   23F1 0E63        2211          SX    NULL,MR1,MAR,COLD.1,C DO WHOLE 256 KB                      32522100
05A6   12F9 6E40        2212          BAL   MEMTEST(MR7)          NOW TEST BASIC MEMORY.               32522110
05A7   323F 1FF1        2213          LI    MR1,'FF1'             MR1 = 'FFFFFFF1'                     32522120
05A8   37F5 5183        2214          NI    NULL,MR5,BIT20,I      DFU EQUIPPED ?                       32522130
05A9   13E1 6B80        2215          BALZ  COLD.3(NULL)          BRANCH: NOT EQUIPPED.                32522140
05AA   CBF9 2FC0        2216 COLD.2   LE    YD,NULL,K             INITIALIZE SPFP REGISTERS            32522150
05AB   CBF9 8F80        2217          LW    YD,NULL               AND DPFP REGISTERS                   32522160
05AC   CBF9 AFC0        2218          LD    YD,NULL,K             .                                    32522170
05AD   23D1 1F6A        2219          AX    YDI,MR1,YDI,COLD.2,C                                       32522180
05AE   F33F 1209        2220 COLD.3   LI    M7YD,ILEGAL           INITIALIZE SCRATCHPADS               32522190
05AF   2B3D 7F00        2221          O     YD,PSW,YDI            AND GENERAL REGISTERS                32522200
05B0   23D1 1F6E        2222          AX    YDI,MR1,YDI,COLD.3,C                                       32522210
05B1   33BD 1010        2223          AI    PSW,PSW,'010'         GO TO NEXT GENERAL REG SET           32522220
05B2   33FD 5080        2224          NI    NULL,PSW,'080'        ALL DONE ?                           32522230
05B3   13E1 6B80        2225          BALZ  COLD.3(NULL)          BRANCH: NO.                          32522240

05B4   37BF 1019        2227          LI    PSW,BIT16,I           PSW = '00008000'                     32522260
05B5   2B5F 2F80        2228          SDEC  CLOC,NULL,NULL        CLOC = '00FFFFF'                     32522270
05B6   4BFF 7FD1        2229          CMCR  NULL,NULL,@LOC        TURN FAULT LAMP OFF, UPDATE ILOC     32522280
05B7   12F9 7380        2230          BAL   TLSU(MR7)             TEST IF LSU ENABLED (P.51)           32522290
                        2231 *                                    MVF INDICATION RESET BY SOFTWARE;    32522300
                        2232 *                                    CONSER RESETS ALL SET MCR BITS.      32522310
05B8   13F8 C000        2233          BAL   CONSER(NULL)          GO DISPLAY PROMPT. (P.30)            32522320

0000 05B9               2235 MEMTEST  EQU   *                                                          32522340
05B9   321F 1000        2236          LI    MR0,0                 LOW MEMORY LIMIT                     32522350
05BA   2B9F 180C        2237 MEMLOOP  L     MAR,MR0,PR4           TEST CELL ADDRESS                    32522360
05BB   2B5F 1E00        2238          L     CLOC,MAR              FOR INSTRUCTION BUFFER TEST          32522370
05BC   2A3F 1D91        2239          L     MR1,RMDR,@LOC         AND SAVE CONTENTS                    32522380
05BD   3650 007B        2240          SI    MR2,MR0,FIVES,I       CREATE DATA PATTERN                  32522390
0000 05BE               2241 MEM.1    EQU   *                     START OF ERROR LOOP                  32522400
05BE   2B7F 1900        2242          L     WMDR,MR2              DATA PATTERN                         32522410
05BF   2BDF 1F9C        2243          L     YDI,NULL,PW4          STORE PATTERN, SELECT R0             32522420
05C0   3390 60FC        2244          XI    MAR,MR0,'FC'          WILL DO DUMMY READ                   32522430
05C1   2B5F 1D0C        2245          L     CLOC,ILOC,PR4         252 BYTES AWAY.                      32522440
```

POWER FAIL/RESTORE ROUTINES

```
05C2   2B9F 180C              2246          L     MAR,MR0,PR4         DID TEST CELL HOLD DATA ?                  32522450
05C3   2BF2 6D80              2247          X     NULL,MR2,RMDR       TEST IT:                                   32522460
05C4   17E1 6F83              2248          BALNZ MEM.1(NULL),DR4IB   BRANCH: DATA WRONG.      (P.50)            32522470
05C5   2BF2 6D80              2249          X     NULL,MR2,RMDR       BUFFER DATA SHOULD BE SAME                 32522480
05C6   17E1 6F80              2250          BALNZ MEM.1(NULL)         BRANCH: IT IS NOT.       (P.50)            32522490
05C7   2B7F 189C              2251          L     WMDR,MR1,PW4        RESTORE ORIGINAL CONTENTS                  32522500
05C8   339F 1040              2252          LI    MAR,'40'            ADDRESS MALFUNCTION STATUS WORD            32522510
05C9   377F 104F              2253          LI    WMDR,BIT01,I        ='40000000'                                32522520
05CA   3210 1004              2254          AI    MR0,MR0,4           ADVANCE TO NEXT CELL                       32522530
05CB   37F0 0009              2255          SI    NULL,MR0,BIT13,I    AT LIMIT ?                                 32522540
05CC   13F1 6E9C              2256          BALC  MEMLOOP(NULL),PW4   BRANCH: NO. ELSE,        (P.50)            32522550
                              2257 *                                  SET 'POWER RESTORE' CODE @ '40',           32522560
05CD   03F8 0B80              2258          BAL   (MR7)(NULL)         RETURN TO CALLER.                          32522570

                              2260 * LOADER STORAGE UNIT INPUT                                                   32522590
                              2261 *                                                                             32522600
0000 05CE                     2262 TLSU     EQU   *                   CHECK IF LSU PRESENT, ENABLED.             32522610
05CE   33F5 5200              2263          NI    NULL,MR5,'200'      INIT KEY DEPRESSED ?                       32522620
05CF   17E0 C000              2264          BALNZ CONSER(NULL)        BRANCH: YES. (P.30)                        32522630
05D0   321F 1005              2265          LI    MR0,5               LSU ADDRESS                                32522640
05D1   4BF0 AFC0              2266          SSRA  NULL,MR0,NULL       ADDRESS IT, GET STATUS                     32522650
05D2   03F4 0B80              2267          BALV  (MR7)(NULL)         RETURN IF FALSE SYNC.                      32522660
0000 05D3                     2268 BOOT     EQU   *                   WE'RE GONNA BOOT IN LSU CONTENTS           32522670
05D3   339F 1001              2269          LI    MAR,1                                                          32522680
05D4   12D9 7880              2270          BAL   READIT(MR6)                                                    32522690
05D5   2AFF 1880              2271          L     MR7,MR1             PSW 16:31                                  32522700
05D6   12D9 7880              2272          BAL   READIT(MR6)                                                    32522710
05D7   2B5F 1880              2273          L     CLOC,MR1            LOC 16:31                                  32522720
05D8   12D9 7880              2274          BAL   READIT(MR6)                                                    32522730
05D9   2ABF 1891              2275          L     MR5,MR1,@LOC        MR5 = START ADRS; UPDATE ILOC.             32522740
05DA   12D9 7880              2276          BAL   READIT(MR6)                                                    32522750
05DB   3395 0001              2277          SI    MAR,MR5,1           START ADRS - 1                             32522760
                              2278 *                                  MR1 = END ADDRESS                          32522770
05DC   23F1 2E5E              2279          SDECX NULL,MR1,MAR,AUTO1,C                                           32522780
05DD   13F8 C000              2280          BAL   CONSER(NULL)        TO CONSER IF START > END (P.30)            32522790
05DE   4B7F 0FDA              2281 AUTO1    RDR   WMDR,NULL,I1DW1     INPUT BYTE, STORE                          32522800
05DF   23F1 0E5E              2282          SX    NULL,MR1,MAR,AUTO1,C LOOP UNTIL END ADRS REACHED               32522810
05E0   2BBF 1B80              2283          L     PSW,MR7             LOAD NEW PSW,                         R02  32522820
05E1   13F9 6780              2284          BAL   PWRUPINT(NULL)      TEST IF MMF ENABLED (P.50)            R02  32522830


05E2   4A30 8FC0              2286 READIT   RDRA  MR1,MR0,NULL        INPUT MS BYTE                              32522850
05E3   4A3F E8C0              2287          EXB   MR1,MR1             LEFT 8                                     32522860
05E4   4A30 8880              2288          RDA   MR1,MR0,MR1         INPUT LS BYTE                              32522870
05E5   03F8 0B00              2289          BAL   (MR6)(NULL)                                                    32522880
```

READ/WRITE CONTROL STORE

```
                               2291  * ROUTINE IS ENTERED HERE, WITH THE FOLLOWING POINTERS:               32522900
                               2292  * (R1)=WCS ADDRESS,(R1+1)=COUNT, (R2)=MAIN MEMORY ADRS                 32522910
                               2293  *                                                                     32522920
05E6    321B 9002              2294  CCS1      SLLI  MR0,YDP1,2            MR0 = COUNT TIMES 4              32522930
05E7    2A10 1C00              2295            A     MR0,MR0,YS            PLUS MEMORY ADDRESS              32522940
05E8    2A3B 1C80              2296            A     MR1,YDP1,YD           MR1=COUNT PLUS WCS ADDRESS       32522950
05E9    225F 0F74              2297            SX    MR2,NULL,YDI,WDCS,C   BRANCH: 0 = WRITE WCS       R02  32522960
05EA    33F2 6FFE              2298            XI    NULL,MR2,-2           YDI MUST BE 2, THEN..       R02  32522970
05EB    13F1 7B40              2299            BALZ  RDCS(NULL)            BRANCH: 2 = READ WCS        R02  32522980
05EC    17FC 8240              2300            BALD  ILEGAL(NULL)          ILLEGAL FUNCTION. (P.18)         32522990

0000 05ED                      2302  RDCS      EQU   *                                                  32523010
05ED    2063 3FB2              2303            AINCX 3,3,NULL,RDCS2        PRE-INCREMENT COUNT         R02  32523020
05EE    2B9F 1800              2304  RDCS1     L     MAR,MR0                                            32523030
05EF    2F7F 189F              2305            L     WMDR,MR1,I,DW4        MOVE WCS DATA TO MAIN MEMORY     32523040
05F0    3210 0004              2306            SI    MR0,MR0,4             DECREMENT MEMORY ADDRESS         32523050
05F1    2A31 2F80              2307            SDEC  MR1,MR1,NULL          DECREMENT WCS ADDRESS            32523060
05F2    2063 2FEE              2308  RDCS2     SDECX 3,3,NULL,RDCS1,C      DECREMENT COUNT                  32523070
05F3    2BFF 1F92              2309            L     NULL,NULL,IRD                                      32523080

0000 05F4                      2311  WDCS      EQU   *                                                  32523100
05F4    2021 3FBA              2312            AINCX 1,1,NULL,WDCS2        PRE-INCREMENT COUNT         R02  32523110
05F5    2B9F 180F              2313  WDCS1     L     MAR,MR0,DR4           FETCH FULLWORD                   32523120
05F6    3210 0004              2314            SI    MR0,MR0,4             DECREMENT MEMORY ADDRESS         32523130
05F7    2A5F 1D80              2315            L     MR2,RMDR              COPY DATA TO MR2                 32523140
05F8    3FF2 0880              2316            STR   MR2,MR1               AND STORE IN WCS                 32523150
05F9    2A31 2F80              2317            SDEC  MR1,MR1,NULL          DECREMENT WCS ADDRESS            32523160
05FA    2021 2FF5              2318  WDCS2     SDECX 1,1,NULL,WDCS1,C      DECREMENT COUNT                  32523170
05FB    2BFF 1F92              2319  EXIT52    L     NULL,NULL,IRD         EXIT IF DONE                     32523180
```

HIGH-SPEED DATA HANDLING INSTRUCTIONS

```
                      * PROCESS BYTE RX                                                32523200
                      *                                                                32523210
05FC   3219 8010      PB1       SRLI  MR0,YD,16          BITS 8:15 OF THE REGISTER     32523220
                      *                                  SPECIFIED BY R1 CONTAIN A     32523230
                      *                                  CONTROL CODE INDICATING       32523240
                      *                                  TYPE OF ERROR CHECKING TO     32523250
05FD   323F 1006                LI    MR1,CRC            BE PERFORMED.  ADDRESS THE    32523260
05FE   4BF1 B840                OCRA  NULL,MR1,MR0       CRC HARDWARE & OUTPUT THE     32523270
                      *                                  CONTROL INFORMATION TO IT.    32523280
05FF   4BFF 5D80                WH    NULL,RMDR          OUTPUT OLD RESIDUAL           32523290
0600   4BFF 1CC0                WDR   NULL,YD            OUTPUT THE DATA BYTE IN R1    32523300
                      *                                  TO BE INCLUDED IN THE ERROR   32523310
0601   4B7F 4F97                RH    WMDR,NULL,DW2      CHECK.  INPUT THE RESULT      32523320
                      *                                  AND STORE IT.                 32523330
0602   2BFF 1F92      EXIT53    L     NULL,NULL,IRD      FETCH NEXT INSTRUCTION        32523340


                      * PROCESS BYTE RR                                                32523360
                      *                                                                32523370
0603   3219 8010      PBR1      SRLI  MR0,YD,16          BITS 0:15 OF THE REGISTER     32523380
                      *                                  SPECIFIED BY R1 CONTAIN A     32523390
                      *                                  CONTROL CODE INDICATING       32523400
                      *                                  TYPE OF ERROR CHECKING TO     32523410
0604   323F 1006                LI    MR1,CRC            BE PERFORMED. ADDRESS THE     32523420
0605   4BF1 B840                OCRA  NULL,MR1,MR0       CRC HARDWARE & OUTPUT THE     32523430
                      *                                  CONTROL INFORMATION TO IT     32523440
0606   4BFF 5C00                WH    NULL,YS            OUTPUT OLD RESIDUAL FROM      32523450
                      *                                  R2 BITS 16:31.                32523460
0607   4BFF 1CC0                WDR   NULL,YD            OUTPUT DATA BYTE FROM R2      32523470
                      *                                  TO BE INCLUDED IN THE         32523480
0608   4B1F 4F92                RH    YS,NULL,IRD        ERROR CHECK. GET RESULT, EXIT.  32523490
```

```
0609    323F 1FF1            2352  STM71    LI    MR1,'FF1'          MR1 = 'FFFFFFF1'                   32523510
060A    EB7F 1C9F            2353  STM72    L     WMDR,M7YD,DW4      STORE SCRATCHPAD REGISTER          32523520
060B    2BFF 1F95            2354           L     NULL,NULL,I4       INCREMENT MAR                      32523530
060C    23D1 1F4A            2355           AX    YDI,MR1,YDI,STM72,C CONTINUE THROUGH M7R15            32523540
060D    03F8 0B00            2356           BAL   (MR6)(NULL)        RETURN TO CALLER                   32523550

060E    323F 1FF1            2358  LM71     LI    MR1,'FF1'          MR1 = 'FFFFFFF1'                   32523570
060F    03F0 0B0F            2359  LM72     BALC  (MR6)(NULL),DR4    RETURN IF DONE, ELSE READ;         32523580
0610    EB3F 1D95            2360           L     M7YD,RMDR,I4       LOAD SCRATCHPAD, INCREMENT MAR     32523590
0611    23D1 1F0F            2361           AX    YDI,MR1,YDI,LM72   AND TRY AGAIN.                     32523600

0612    CBF9 8D8E            2363  LD1      LW    YD,RMDR,I4DR4      LOAD HIGH HALF DPFP DATA       R02 32523620
0613    CBF9 ADB2            2364           LD    YD,RMDR,IRD,E      LOAD LOW, EXIT.                R02 32523630

0614    CB7F 9C80            2366  STD1     RRD   WMDR,YD            FETCH MS 32 BITS                   32523650
0615    2B9A 1D9F            2367           A     MAR,YX,RMDR,DW4    STORE HIGH HALF                R02 32523660
0616    2BDF 3F15            2368           AINC  YDI,NULL,YDI,I4    POINT TO LOW HALF                  32523670
0617    CB7F 9C9F            2369           RRD   WMDR,YD,DW4        READ LS 32 BITS, AND STORE         32523680
0618    2BFF 1F92            2370  EXIT54   L     NULL,NULL,IRD                                         32523690

0619    2B9A 1D80            2372  STMD1    A     MAR,YX,RMDR        CALCULATE START ADDRESS            32523710
061A    323F 000F            2373           SI    MR1,NULL,15        MR1='FFFFFFF1'                     32523720
061B    CB7F 9C9F            2374  STMD2    RRD   WMDR,YD,DW4        STORE HALF A DOUBLE REGISTER       32523730
061C    2BFF 1F95            2375           L     NULL,NULL,I4                                          32523740
061D    23D1 1F5B            2376           AX    YDI,MR1,YDI,STMD2,C BRANCH: MORE TO DO.               32523750
061E    2BFF 1F92            2377           L     NULL,NULL,IRD      EXIT.                          R02 32523760
                             2378  *                                                                    32523770
061F    323F 000F            2379  STMD@    SI    MR1,NULL,15        MR1='FFFFFFF1'                     32523780
0620    CB7F 9C9F            2380  STMD@1   RRD   WMDR,YD,DW4        STORE HALF A DOUBLE REGISTER       32523790
0621    2BFF 1F95            2381           L     NULL,NULL,I4                                          32523800
0622    23D1 1F60            2382           AX    YDI,MR1,YDI,STMD@1,C BRANCH: MORE TO DO.              32523810
0623    03F8 0B00            2383           BAL   (MR6)(NULL)        RETURN TO CALLER.                  32523820

0624    2B9A 1D80            2385  LMD1     A     MAR,YX,RMDR        CALCULATE ADDRESS                 332523840
0625    323F 1FF2            2386           LI    MR1,'FF2'          MR1 = 'FFFFFFF2'               R02 32523850
0626    13F1 860F            2387  LMD2     BALC  EXIT54(NULL),DR4   EXIT AS YDI BECOMES ZERO       R02 32523860
0627    CBF9 8D8E            2388           LW    YD,RMDR,I4DR4      LOAD HIGH HALF, READ LOW      R02  32523870
0628    CBF9 ADD5            2389           LD    YD,RMDR,K,I4       LOAD LOW HALF, INCREMENT MAR  R02  32523880
0629    23D1 1F26            2390           AX    YDI,MR1,YDI,LMD2   INCREMENT BY 2                 R02 32523890

                             2392  *                                                                    32523910
062A    323F 1FF2            2393  LMD@     LI    MR1,'FF2'          MR1 = 'FFFFFFF2'                   32523920
062B    03F0 0B0F            2394  LMD@1    BALC  (MR6)(NULL),DR4    BRANCH: RETURN TO CALLER.          32523930
062C    CBF9 8D8E            2395           LW    YD,RMDR,I4DR4      LOAD HIGH HALF, READ LOW           32523940
062D    CBF9 ADD5            2396           LD    YD,RMDR,K,I4       LOAD LOW HALF, INCREMENT MAR       32523950
062E    23D1 1F2B            2397           AX    YDI,MR1,YDI,LMD@1  TRY AGAIN...                       32523960
```

```
 062F   2BFF 1822              2399  FXDR1    L     NULL,MR0,IR,E        SET CC= 0000,0001, OR 0010              32523980
 0630   2ADF 1F00              2400           L     MR6,YDI                                                      32523990
 0631   2BDF 3E80              2401           AINC  YDI,NULL,YSI         POINT TO R2+1                           32524000
 0632   CA7F 9C80              2402           RRD   MR3,YD               MR0,MR3=ARGUMENT                        32524010
 0633   3230 B008              2403           RLLI  MR1,MR0,8            ROTATE MS 32 BITS                       32524020
 0634   3273 B008              2404           RLLI  MR3,MR3,8            ROTATE LS 32 BITS                       32524030
 0635   3251 5F00              2405           NI    MR2,MR1,'F00'        FRACTION BITS 0:23                      32524040
 0636   3273 50FF              2406           NI    MR3,MR3,'0FF'        FRACTION BITS 24:31                     32524050
 0637   2A52 7980              2407           O     MR2,MR2,MR3          COMBINED (8 HEX DIGITS)                 32524060
 0638   23DF 1B3D              2408           LX    YDI,MR6,FXDR2        RESTORE R1 SELECT.                      32524070


 0639   2BFF 1822              2410  FXR1     L     NULL,MR0,IR,E        SET CC= 0000,0001, OR 0010              32524090
 063A   2BFF 1F80              2411           L     NULL,NULL            TURN OFF E-BIT LATENCY                  32524100
 063B   3230 B008              2412           RLLI  MR1,MR0,8                                                    32524110
 063C   3251 5F00              2413           NI    MR2,MR1,'F00'        MR2 = FRACTION LEFT 8                   32524120
 0000 053D                     2414  FXDR2    EQU   *                                                            32524130
 063D   321D 5001              2415           NI    MR0,PSW,'01'         CAPTURE PSW 'L' FLAG                    32524140
 063E   3231 507F              2416           NI    MR1,MR1,'07F'        MR1 = EXPONENT                          32524150
 063F   327F 1048              2417           LI    MR3,'48'                                                     32524160
 0640   2233 08C4              2418           SX    MR1,MR3,MR1,FXR3,C  COMPARE EXPONENT TO LIMIT                32524170
                               2419  *        MR0 CORRESPONDS TO FLOAT VALUE:                                    32524180
                               2420  *                                   +: 00000000    -: 00000001             32524190
 0641   3610 6081              2421  OVERFLO  XI    MR0,MR0,BI0031,I     +: 80000001    -: 80000000             32524200
 0642   33BD 7004              2422           OI    PSW,PSW,'04'         SET V FLAG                              32524210
 0643   2B3F 0810              2423           S     YD,NULL,MR0,D        +: 7FFFFFFF    -: 80000000             32524220

 0644   33F1 0008              2425  FXR3     SI    NULL,MR1,8           COMPARE COUNT TO LIMIT                  32524240
 0645   17E5 9380              2426           BALNL UNDERFLO(NULL)       BRANCH: LOSES ALL PRECISION.            32524250
 0646   3231 9002              2427           SLLI  MR1,MR1,2            SET FOR HEX SHIFTS                      32524260
 0647   2B32 8880              2428           SRL   YD,MR2,MR1           AND SHIFT.                              32524270
 0648   2279 1CCB              2429           AX    MR3,YD,YD,FXR4,C     BRANCH: NO SIGN CONFLICT                32524280
 0649   23FF 0841              2430           SX    NULL,NULL,MR0,OVERFLO,C BRANCH: POSITIVE, CAN'T DO.          32524290
 064A   23F3 2FC1              2431           SDECX NULL,MR3,NULL,OVERFLO,C BRANCH: NEG, MAG > '80000000'        32524300

 064B   23FF 084D              2433  FXR4     SX    NULL,NULL,MR0,FXR5,C BRANCH: REALLY POSITIVE                 32524320
 064C   2B3F 0C90              2434           S     YD,NULL,YD,D         2'S COMPLEMENT, EXIT.                   32524330
 064D   2BFF 1F90              2435  FXR5     L     NULL,NULL,D          EXIT.                                   32524340

 064E   2B3F 1FB0              2437  UNDERFLO L     YD,NULL,D,E          LOAD 00000000, SET CC = 0, EXIT.        32524360
```

MIXED MODE FLOATING POINT INSTRUCTIONS

```
0000 064F          LGDR1   EQU   *                   (16)                                   32524380
064F   CA9F 9C02           RRD   MR4,YS,IR           HIGH HALF                              32524390
0650   2BDF 3E80           AINC  YDI,NULL,YSI                                               32524400
0651   CABF 9C80           RRD   MR5,YD              LOW HALF                               32524410
0652   2BDF 1800           L     YDI,MR0                                                    32524420
0653   2B3F 1A20           L     YD,MR4,E            HIGH HALF, SET CC                      32524430
0654   2BDF 3F00           AINC  YDI,NULL,YDI                                               32524440
0655   2B3F 1A90           L     YD,MR5,D            LOW HALF; EXIT.                        32524450

0000 0656          LPDR1   EQU   *                   (33)                                   32524470
0000 0656          LCDR1   EQU   *                   (37)                                   32524480
0656   2A1F 1F00           L     MR0,YDI             REMEMBER R1 SELECT                     32524490
0657   CA9F 9C00           RRD   MR4,YS              HIGH HALF, DPFP DATA                   32524500
0658   2BDF 3E80           AINC  YDI,NULL,YSI                                               32524510
0659   CA7F 9C80           RRD   MR3,YD              LOW HALF                               32524520
065A   2BDF 1800           L     YDI,MR0             RESTORE R1 SELECT                      32524530
065B   0BF8 0B00           EXL   (MR6)(NULL)         ADJUST SIGN BIT                        32524540
065C   CBF9 8A00           LW    YD,MR4              LOAD HIGH HALF;                    R02 32524550
065D   CBF9 A9B2           LD    YD,MR3,IRD,E        LOAD LOW, EXIT.                    R02 32524560

0000 065E          STDE1   EQU   *                   (82)                                   32524580
065E   2ADD 1F80           A     MR6,PSW,NULL        REMEMBER OLD CC                        32524590
065F   CA7F 9C80           RRD   MR3,YD              GET DATA                               32524600
0660   2BDF 3F00           AINC  YDI,NULL,YDI                                               32524610
0661   CA9F 9C80           RRD   MR4,YD              LOW HALF                               32524620
0662   CAFF 1C80           RRE   MR7,YD              SAVE 'OLD' SINGLE FLOAT REGISTER       32524630
0663   CBF9 8980           LW    YD,MR3              HIGH HALF, NEW DATA                    32524640
0664   CBF9 2A20           LE    YD,MR4,E            LOW HALF; ROUND, SET CC.               32524650
0665   CB7F 1C80           RRE   WMDR,YD             GET ROUNDED RESULT.                    32524660
0666   CBF9 2BC0           LE    YD,MR7,K            RESTORE 'OLD' DATA IMAGE               32524670
0667   2AFD 1F80           A     MR7,PSW,NULL        GET FLAGS FROM LOAD/ROUND              32524680
0668   2BBF 1B00           L     PSW,MR6             AND RESTORE OLD -                      32524690
0669   33F7 5004           NI    NULL,MR7,4          ANY ERROR REPORTED ?                   32524700
066A   13E1 9B00           BALZ  STDE2(NULL)         BRANCH: NO.                            32524710
066B   33F7 5003           NI    NULL,MR7,3          OVERFLOW RESULTED ?                    32524720
066C   17E0 98DF   STDE2   BALNZ AFAUL4(NULL),DW4    BRANCH: YES. (P.23)                    32524730
066D   2BFF 1F92   EXIT57  L     NULL,NULL,IRD       EXIT, WITH ENTRY CC.                   32524740
```

FREE SPACE

```
066E                          2477          IFP   RXRX1-*          .                     R02  32524760
066E                          2478          DO    RXRX1-*          .                     R02  32524770
066E    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
066F    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0670    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0671    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0672    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0673    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0674    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0675    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0676    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0677    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0678    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0679    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067A    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067B    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067C    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067D    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067E    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
067F    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0680    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0681    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0682    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0683    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0684    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
0685    0000 0000             2479          DC    FREEWORD         .                     R02  32524780
                              2480          ENDC                   .                     R02  32524790
```

STRING INSTRUCTIONS

```
0686                        2482          ORG   '686'                .                              R02  32524810
0000 0586                   2483  RXRX1   EQU   *                    STORAGE-STORAGE PREPROCESSOR        32524820
0686    2B9A 1D80           2484          A     MAR,YX,RMDR          CALCULATE ADDRESS              R02  32524830
0687    E83F 1E00           2485          L     M7R1,MAR             M7R1 = 1ST OP ADDRESS          R02  32524840
0688    E89F 1F01           2486          L     M7R4,YDI,DR2IB       IMMEDIATE LEN1 (?)                  32524850
                            2487  *                                  FETCH XOP HALFWORD FROM 2ND RX      32524860
0689    4A1F EDC0           2488          EXB   MR0,RMDR             MR0 = YD2, YS2, XOP FIELDS          32524870
068A    E81F 1800           2489          L     M7R0,MR0             COPY TO R0.                         32524880
068B    F3E0 5080           2490          NI    NULL,M7R0,'80'       IMM LEN1 SPEC'D ?                   32524890
068C    17E1 A3C0           2491          BALNZ RXRX2(NULL)          BRANCH: YES.                        32524900
068D    2ABF 1C80           2492          L     MR5,YD               LENGTH IS IN REGISTER               32524910
068E    E89F 1A80           2493          L     M7R4,MR5             LEN1                                32524920
068F    F3C0 800C           2494  RXRX2   SRLI  YDI,M7R0,12          SELECT YD2 REGISTER                 32524930
0690    E8BF 1F00           2495          L     M7R5,YDI             IMMEDIATE LEN2 (?)                  32524940
0691    F3E0 5040           2496          NI    NULL,M7R0,'40'       IMM LEN2 SPEC'D ?                   32524950
0692    17E1 A540           2497          BALNZ RXRX3(NULL)          BRANCH: YES.                        32524960
0693    2ABF 1C80           2498          L     MR5,YD               LENGTH IS IN REGISTER               32524970
0694    E8BF 1A80           2499          L     M7R5,MR5             LEN2                                32524980
0000 0595                   2500  RXRX3   EQU   *                                                        32524990
0695    2BDF 1D81           2501          L     YDI,RMDR,DR2IB       FIRST INDEX; FETCH 2ND HALFWORD OF RX32525000
0696    227F 0F58           2502          SX    MR3,NULL,YDI,NOX1,C BRANCH: NO FIRST INDEX SPEC'D        32525010
0697    2A7F 1C80           2503          L     MR3,YD               FIRST INDEX VALUE                   32525020
0698    2A3F 1D80           2504  NOX1    L     MR1,RMDR                                                 32525030
0699    13E5 A940           2505          BALL  RX2(NULL)            BRANCH: RX2 FORMAT                  32525040
069A    37F1 500D           2506          NI    NULL,MR1,BIT17,I     RX3, THEN ?                         32525050
069B    13E1 A901           2507          BALZ  RXEXIT(NULL),DR2IB   BRANCH: RX1.                        32525060
                            2508  *                                                                      32525070
069C    4ABF 58C0           2509  RX3     LBR   MR5,MR1              MR5 = INSTR BITS 24:31              32525080
069D    4BDF E8C0           2510          EXB   YDI,MR1              SELECT SX2 (?)                      32525090
069E    23FF 0F60           2511          SX    NULL,NULL,YDI,NOX2,C BRANCH: NO SX2 SPEC'D.              32525100
069F    2A73 1C80           2512          A     MR3,MR3,YD           ADD SX2 VALUE                       32525110
06A0    3235 9010           2513  NOX2    SLLI  MR1,MR5,16           SLIDE PREVIOUS HALF OF A2           32525120
06A1    361F 1017           2514          LI    MR0,BI16.31,I                                            32525130
06A2    2A10 5D80           2515          N     MR0,MR0,RMDR         LOW HALF OF A2                      32525140
06A3    2A31 7800           2516          O     MR1,MR1,MR0          COMBINE HIGH & LOW HALF OF RX3 ADDRES32525150
06A4    E053 18A9           2517  RXEXIT  AX    M7R2,MR3,MR1,RXDCODE EFFECTIVE ADDRESS 2                 32525160
                            2518  *                                                                      32525170
0000 06A5                   2519  RX2     EQU   *                                                        32525180
06A5    37F1 500D           2520          NI    NULL,MR1,BIT17,I     BACKWARDS DISPLACEMENT ?            32525190
06A6    17E1 AA00           2521          BALNZ RX2.BCK(NULL)        BRANCH: YES.                        32525200
06A7    3631 5075           2522  RX2.FWD NI    MR1,MR1,BI17.31,I    KEEP POSITIVE DISPLACEMENT          32525210
06A8    223C 18A4           2523  RX2.BCK AX    MR1,CLOC,MR1,RXEXIT GET DISPLACEMENT FROM LOC            32525220

0000 06A9                   2525  RXDCODE EQU   *                                                        32525240
06A9    E95C 1F85           2526          A     M7R10,CLOC,NULL,RFAULT INCR'D LOC; RESET RX FLOPS.       32525250
06AA    F200 501F           2527          NI    MR0,M7R0,'1F'        GET FUNCTION CODE                   32525260
06AB    3230 16B1           2528          AI    MR1,MR0,RXRXTAB                                          32525270
06AC    33F0 0005           2529          SI    NULL,MR0,5           VALID FUNCTION CODE ?               32525280
06AD    03F0 0880           2530          BALC  (MR1)(NULL)          START INSTRUCTION.                  32525290
06AE    17FC 8240           2531          BALD  ILEGAL(NULL)         ILLEGAL FUNCTION. (P.18)            32525300
```

STRING INSTRUCTIONS

```
                         * GLOBAL REGISTER USAGE:                                                   32525320
                         *                                                                          32525330
                         * M7R0 = XOP                                                               32525340
                         * M7R1 = OP1 STRING POINTER                                                32525350
                         * M7R2 = OP2 STRING POINTER                                                32525360
                         * M7R4 = OP1 LENGTH                                                        32525370
                         * M7R5 = OP2 LENGTH                                                        32525380
                         * M7R10 = INCREMENTED LOC                                                  32525390
                         * M7R11 = INTERRUPT RETURN ADDRESS                                         32525400

                         * CONSTANTS USED IN ROUTINE 'STBP'                                         32525420
06AF    05F5 E100        TENUP8   DC    '05F5E100'          CONSTANT 10**8                          32525430
06B0    02FA F080        HALFTEN8 DC    '02FAF080'          CONSTANT (10**8)/2                      32525440

06B1    13F9 AF80        RXRXTAB  BAL   MVTU(NULL)          8C/00 (P.60)                            32525460
06B2    13F9 AF80                 BAL   MOVE(NULL)          8C/01 (P.60)                            32525470
06B3    13F9 B700                 BAL   CPAN(NULL)          8C/02 (P.61)                            32525480
06B4    13F9 BEC0                 BAL   PMV(NULL)           8C/03 (P.63)                            32525490
06B5    13F9 D4C0                 BAL   UMV(NULL)           8C/04 (P.66)                            32525500

                         * ROUTINE GETS REQUIRED PAD CHARACTER INTO R0; RETURNS                     32525520
                         * TO USER VIA 'GET.END' ROUTINE.                                           32525530
06B6    F000 5020        GETPAD   NI    M7R0,M7R0,'20'      DEFAULT PAD SPEC'D ?                    32525540
06B7    17E1 AE80                 BALNZ GET.END(NULL)       BRANCH: YES. WE HAVE '20'.              32525550
06B8    3240 50FF                 NI    MR2,R0,'FF'         PADC FROM GR0 IS USED;                  32525560
06B9    E81F 1900                 L     M7R0,MR2            SAVE IN R0.                             32525570

                         * ROUTINE COMPUTES END ADDRESSES FOR STRINGS 1, 2                          32525590
06BA    E821 1200        GET.END  A     M7R1,M7R1,M7R4      END1 = A(STR1) + LEN1                   32525600
06BB    E842 1280                 A     M7R2,M7R2,M7R5      END2 = A(STR2) + LEN2                   32525610
06BC    33BD 5FF0                 NI    PSW,PSW,'FF0'       PRE-ZERO CONDITION CODE                 32525620
06BD    03F8 0B00                 BAL   (MR6)(NULL)         RETURN.                                 32525630
```

STRING INSTRUCTIONS

```
                  * FOR THE FOLLOWING MOVE ROUTINES, THE FOLLOWING CONVENTIONS HOLD:       32525650
                  * MVTU: AT INPUT, GR0 CONTAINS AN 'ESCAPE' OR 'UNTIL' CHARACTER.         32525660
                  *                GR2 MAY CONTAIN THE ADDRESS OF A TRANS. TABLE.          32525670

                  * ALL: AT OUTPUT, GR1 IS LOADED WITH THE ADDRESS OF THE NEXT             32525690
                  *                BYTE TO BE PROCESSED IN THE SECOND OPERAND STRING.      32525700
                  *                IF THIS STRING IS OF ZERO LENGTH, THE ADDRESS           32525710
                  *                OF THE SPECIFIED STRING START IS RETURNED.              32525720

0000 05BE         MVTU     EQU   *                      (8C/00)                            32525740
0000 05BE         MOVE     EQU   *                      (8C/01)                            32525750
0000 05BE         MOVEP    EQU   *                      (8C/21)                            32525760
06BE   E87F 1800           L     M7R3,MR0               FUNCTION CODE TO R3                32525770
06BF   12D9 AD80           BAL   GETPAD(MR6)            GET PADC, CC=0, STRING ENDS. (P.59) 32525780
06C0   12D8 8F00           BAL   SET.RTN(MR6)           SET INTERRUPT RETURN IN R11 (P.21)  32525790

06C1   EBFF 1280  MOVE1    L     NULL,M7R5              TEST LEN2:                         32525810
06C2   17E9 B440           BALNG PADIT(NULL)            BRANCH: LEN1 NOT > 0               32525820
06C3   EB82 028B           S     MAR,M7R2,M7R5,DR1      FETCH BYTE @ (END2-LEN2)           32525830
06C4   E3E3 2FC9           SDECX NULL,M7R3,NULL,MOVE2,C TEST FLAG IN R3:                   32525840
                  *                                     BRANCH IF MOVE OR MOVEP.           32525850
06C5   23FF 0147  MVTU1    SX    NULL,NULL,R2,MVTU2,C MVTU: BRANCH IF A(TRTBL) = 0         32525860
06C6   2B82 1D8B           A     MAR,R2,RMDR,DR1        FETCH BYTE FROM TRTBL SPEC'D BY GR2 32525870
06C7   2BE0 6D80  MVTU2    X     NULL,R0,RMDR           SAME AS 'UNTIL' BYTE IN GR0 ?      32525880
06C8   13E1 B600           BALZ  TERMCHAR(NULL)         BRANCH: YES.                       32525890

06C9   EBFF 1200  MOVE2    L     NULL,M7R4              TEST LEN1                          32525910
06CA   17E9 B640           BALNG OUTPUT.Z(NULL)         BRANCH: LEN1 NOT > 0               32525920
06CB   EB81 0200           S     MAR,M7R1,M7R4          A(BYTE1) = END1 - LEN1             32525930
06CC   2B7F 1D9B           L     WMDR,RMDR,DW1          STORE OUTPUT BYTE                  32525940
06CD   E8A5 2F80           SDEC  M7R5,M7R5,NULL         DECREMENT LEN2,                    32525950
06CE   E884 2F80           SDEC  M7R4,M7R4,NULL         DECREMENT LEN1,                    32525960
06CF   12DD B410           BALA  MOVE2A(MR6),D          SERVICE ANY INTERRUPT              32525970
06D0   17FD B040  MOVE2A   BALD  MOVE1(NULL)            AND LOOP.                          32525980

0000 06D1         PADIT    EQU   *                      FILLS OUT DESTINATION WITH PADC    32526000
06D1   E3FF 01DA           SX    NULL,NULL,M7R3,OUTPUT.A,C BRANCH: MVTU DOESN'T PAD.       32526010
06D2   EB7F 1000           L     WMDR,M7R0              PADC FROM R0                       32526020
0000 06D3         PAD.0    EQU   *                                                         32526030
06D3   EBFF 1200           L     NULL,M7R4              TEST LEN1                          32526040
06D4   17E9 B680           BALNG OUTPUT.A(NULL)         BRANCH: LEN1 EXHAUSTED             32526050
06D5   EB81 021B           S     MAR,M7R1,M7R4,DW1      STORE PAD @ (END1-LEN1)            32526060
06D6   E884 2F80           SDEC  M7R4,M7R4,NULL         DECREMENT LEN1 BY 1                32526070
06D7   12DD B4D0           BALA  PAD.0(MR6),D           LOOP.                              32526080

06D8   33BD 7004  TERMCHAR OI    PSW,PSW,4              WILL SET C FLAG                    32526100
06D9   33BD 1004  OUTPUT.Z AI    PSW,PSW,4              ENTER HERE TO SET V FLAG           32526110
06DA   EA02 0280  OUTPUT.A S     MR0,M7R2,M7R5          (END2-LEN2)                        32526120
06DB   203F 182E           LX    R1,MR0,PAD.1           THE ADDRESS OF THE NEXT STR2 BYTE TO 32526130
                  *                                     BE PROCESSED IS RETURNED IN GR1.(P.6132526140
```

STRING INSTRUCTIONS

```
                     * FOR THE FOLLOWING COMPARE ROUTINES, THE FOLLOWING CONVENTIONS HOLD:   32526160
                     * ALL: AT OUTPUT, GR1 IS LOADED WITH THE OFFSET  OF THE LAST BYTE       32526170
                     *                PROCESSED IN THE SECOND OPERAND STRING. FOR            32526180
                     *                A NON-EQUAL COMPARE, THIS INDICATES THE FIRST          32526190
                     *                (LOWEST-ADDRESSED) STRING 2 BYTE WHICH DOES NOT        32526200
                     *                EQUAL THE CORRESPONDING STRING 1 BYTE; PROVIDED        32526210
                     *                THAT STRING 2 IS NOT SHORTER THAN STRING 1.            32526220
                     *                IF STRING 2 IS OF LENGTH ZERO BYTES, ZERO              32526230
                     *                IS RETURNED.                                           32526240

0000 06DC            CPAN     EQU   *                    (8C/02)                             32526260
0000 06DC            CPANP    EQU   *                    (8C/22)                             32526270
06DC   E87F 1100              L     M7R3,M7R2            SAVE A(STR2) IN R3                  32526280
06DD   EA1F 1100              L     MR0,M7R2             AND IN R1                           32526290
06DE   283F 1800              L     R1,MR0               .                                   32526300
06DF   12D9 AD80              BAL   GETPAD(MR6)          GET PADC, CC=0, STRING ENDS (P.59)  32526310
06E0   12D8 8F00              BAL   SET.RTN(MR6)         SET INTERRUPT RETURN IN R11 (P.21)  32526320

0000 06E1            CPAN1    EQU   *                                                        32526340
06E1   EBFF 1280              L     NULL,M7R5            TEST LEN2                           32526350
06E2   17E9 BDC0              BALNG CPAN20(NULL)         BRANCH: LEN2 EXHAUSTED.     (P.62)  32526360
06E3   EBFF 1200              L     NULL,M7R4            TEST LEN1                           32526370
06E4   17E9 BCC0              BALNG CPAN10(NULL)         BRANCH: LEN1 EXHAUSTED.             32526380
06E5   EB82 028B              S     MAR,M7R2,M7R5,DR1    A(BYTE2) = END2-LEN2                32526390
06E6   283F 1E00              L     R1,MAR               A(BYTE2) FOR OFFSET                 32526400
06E7   2A3F 1D80              L     MR1,RMDR             BYTE2                               32526410
06E8   EB81 020B              S     MAR,M7R1,M7R4,DR1    A(BYTE1) = END1-LEN1                32526420
06E9   2A1F 1D80     CPAN2    L     MR0,RMDR             BYTE1                               32526430
06EA   2BF0 08A0     CPAN3    S     NULL,MR0,MR1,E       (BYTE1-BYTE2); CC = 0, 2, OR 9.     32526440
06EB   13E1 BBC0              BALZ  CPAN4(NULL)          BRANCH: STILL EQUAL.                32526450

0000 06EC            MISMATCH EQU   *                    AT TERMINATION OF CPAN/CPANP,       32526470
                     *                                   THE ADDRESS OF THE STR2 BYTE        32526480
                     *                                   AT MISMATCH IS RETURNED IN GR1.     32526490
                     *                                   THIS REFLECTS THE FIRST MISMATCH    32526500
                     *                                   IN A 'FIND STR1 IN STR2' SEARCH,    32526510
                     *                                   BUT REQUIRES LEN2 >= LEN1 FOR       32526520
                     *                                   A SENSIBLE RESULT.                  32526530
06EC   EA1F 1180              L     MR0,M7R3             A(STR2 START) FROM R3               32526540
06ED   2821 0800              S     R1,R1,MR0            OFFSET = A(BYTE2) - A(STR2)         32526550
06EE   17FD DCC0     PAD.1    BALD  IIPEXIT(NULL)        ZERO IIP BIT, EXIT (P.67)           32526560

06EF   E884 2F80     CPAN4    SDEC  M7R4,M7R4,NULL       DECREMENT LEN1,                     32526580
06F0   E8A5 2F80              SDEC  M7R5,M7R5,NULL       DECREMENT LEN2                      32526590
06F1   12DD 3C90              BALA  CPAN5(MR6),D         SERVICE ANY INTERRUPT               32526600
06F2   17FD 3840     CPAN5    BALD  CPAN1(NULL)          LOOP.                               32526610

0000 06F3            CPAN10   EQU   *                    STRING 1 EXHAUSTED (DEST'N)         32526630
06F3   EB82 028B              S     MAR,M7R2,M7R5,DR1    A(BYTE2) = END2 - LEN2              32526640
06F4   283F 1E00              L     R1,MAR               A(BYTE2) TO RETURN OFFSET           32526650
06F5   EA1F 1000              L     MR0,M7R0             PADC TO MR0 FOR BYTE1               32526660
06F6   223F 1DAA              LX    MR1,RMDR,CPAN3       BYTE2; GO COMPARE.                  32526670
```

STRING INSTRUCTIONS

```
0000 06F7                2670  CPAN20   EQU   *                     STRING 2 EXHAUSTED (SOURCE)            32526690
06F7    EBFF 1200        2671           L     NULL,M7R4             TEST LEN1                              32526700
06F8    17E9 BB00        2672           BALNG MISMATCH(NULL)        BRANCH: LEN1 ALSO EXHAUSTED. (P.61)    32526710
06F9    EB81 020B        2673           S     MAR,M7R1,M7R4,DR1     A(BYTE1) = END1 - LEN1                 32526720
06FA    E23F 1029        2674           LX    MR1,M7R0,CPAN2        PADC TO MR1 FOR BYTE2; GO CHECK.       32526730
```

```
                                2676  * FOR THE FOLLOWING PACK ROUTINES, THE FOLLOWING CONVENTIONS HOLD:       32526750
                                2677  * ALL: AT INPUT, STRING LENGTHS ONE GREATER THAN SPECIFIED IN            32526760
                                2678  *                THE USER-LEVEL INSTRUCTION ARE ASSUMED. NO             32526770
                                2679  *                MAXIMUM IS IMPOSED. PACK PROCEEDS RIGHT-TO-LEFT.       32526780
                                2680  *                AN ODD NUMBER OF PACKED DATA DIGITS IS ALWAYS          32526790
                                2681  *                PRODUCED.                                              32526800

                                2683  *                NO INTERRUPT RESULTS FROM AN INVALID SIGN OR           32526820
                                2684  *                DATA DIGIT.                                            32526830

                                2686  *                STANDARD SIGN DIGITS 'C' OR 'D' ARE PRODUCED.          32526850
                                2687  *                A STRING REPRESENTING ZERO HAS SIGN DIGIT 'C'.         32526860


 0000 06FB                      2689  PMV     EQU   *                   (8C/03)                              32526880
 0000 06FB                      2690  PMVA    EQU   *                   (8C/23)                              32526890
 06FB    12D8 8F00              2691          BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)   32526900
 06FC    E87F 1FA0              2692          L     M7R3,NULL,E         PRESET FLAGS = 0                     32526910
 06FD    12D9 C680              2693          BAL   GETBYTE(MR6)        GET UNPACKED STR2 SIGNED BYTE (P.64) 32526920
 06FE    3213 8004              2694          SRLI  MR0,MR3,4           MOVE SIGN DIGIT RIGHT                32526930
 06FF    12D9 CA80              2695          BAL   SGN.CHK(MR6)        CHECK FOR VALID SIGN (P.64)          32526940
 0700    12D9 C980              2696          BAL   DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64) 32526950
 0701    3252 9004              2697          SLLI  MR2,MR2,4           MOVE DATA DIGIT LEFT                 32526960
 0702    2A92 7800              2698          O     MR4,MR2,MR0         FORM PACKED SIGNED BYTE              32526970
 0703    12D9 C780              2699          BAL   STORBYTE(MR6)       AND STORE @ (STR1+LEN1) (P.64)       32526980
 0704    E8A5 2F80              2700          SDEC  M7R5,M7R5,NULL      DECREMENT LEN2 IN R5,                32526990
 0705    E884 2F80              2701          SDEC  M7R4,M7R4,NULL      LEN1 IN R4.                          32527000
 0706    12D8 8F00              2702          BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)   32527010
                                2703  *                                 IF FAULT, RETRY FROM SIGNED BYTE.    32527020

 0000 0707                      2705  PMVLP1  EQU   *                   PACK LOOP                            32527040
                                2706  *                                 OPERATES RIGHT-TO-LEFT.              32527050
 0707    E3E5 12C9              2707          AX    NULL,M7R5,M7R5,PMVL.1,C BRANCH: LEN2 NOT EXHAUSTED.      32527060
 0708    17FD CE80              2708          BALD  PUSHP(NULL)         LEN2 EXHAUSTED. OUTPUT '00'. (P.65)  32527070
 0709    16DD C680              2709  PMVL.1  BALD  GETBYTE(MR6)        GET (LOW) ZONED BYTE (P.64)          32527080
 070A    12D9 C980              2710          BAL   DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64) 32527090
 070B    2A9F 1900              2711          L     MR4,MR2             RETAIN DIGIT IN MR4                  32527100
 070C    EA65 2F80              2712          SDEC  MR3,M7R5,NULL       GET DECREMENTED LEN2                 32527110
 070D    13E5 C500              2713          BALL  PMVL.2(NULL)        BRANCH: LEN2 EXHAUSTED               32527120
 070E    EB93 110B              2714          A     MAR,MR3,M7R2,DR1    GET (HIGH) ZONED BYTE                32527130
 070F    325F 100F              2715          LI    MR2,'0F'            DIGIT MASK                           32527140
 0710    2A52 5D80              2716          N     MR2,MR2,RMDR        EXTRACT DATA DIGIT                   32527150
 0711    12D9 C980              2717          BAL   DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64) 32527160
 0712    3252 9004              2718          SLLI  MR2,MR2,4           POSITION HIGH DIGIT                  32527170
 0713    2A92 7A00              2719          O     MR4,MR2,MR4         PACK                                 32527180
 0714    E3E4 1256              2720  PMVL.2  AX    NULL,M7R4,M7R4,PMVL.3,C BRANCH: LEN1 >= 0                32527190
 0715    13F9 D1C0              2721          BAL   FLUSHP(NULL)        LEN1 < 0. (MR4) MUST BE ZERO. (P.65) 32527200
 0716    12D9 C780              2722  PMVL.3  BAL   STORBYTE(MR6)       STORE PACKED OUTPUT IN STR1 (P.64)   32527210
 0717    F0A5 0002              2723          SI    M7R5,M7R5,2         DECREMENT LEN2 BY 2,                 32527220
 0718    E884 2F80              2724          SDEC  M7R4,M7R4,NULL      DECREMENT LEN1 BY 1                  32527230
 0719    13FD C1D0              2725          BALA  PMVLP1(NULL),D      LOOP, ALLOW INTERRUPT.               32527240
```

STRING INSTRUCTIONS

```
                  * ROUTINE GETS BYTE FROM MEMORY @ END2                                  32527260
0000 071A         GETBYTE  EQU   *                                                        32527270
071A   EB82 128B                A     MAR,M7R2,M7R5,DR1   A(BYTE2) = A(STR2) + LEN2       32527280
071B   2A9F 1D80                L     MR4,RMDR            EXTRACT BYTE,                   32527290
071C   3274 50F0                NI    MR3,MR4,'F0'        HIGH DIGIT TO MR3,              32527300
071D   2253 6A20                XX    MR2,MR3,MR4,GETB1   LOW DIGIT TO MR2.               32527310

                  * ROUTINE STORES BYTE IN MEMORY @ END1                                  32527330
0000 071E         STORBYTE EQU   *                                                        32527340
071E   2B7F 1A00                L     WMDR,MR4            DATA TO STORE                   32527350
071F   EB81 121B                A     MAR,M7R1,M7R4,DW1   A(BYTE1) = A(STR1) + LEN1       32527360
0720   03F8 0B00  GETB1         BAL   (MR6)(NULL)         RETURN.                         32527370

                  * ROUTINE ADJUSTS CC ACCORDING TO DATA QUEUED IN R3                     32527390
0000 0721         PMVFLAGS EQU   *                                                        32527400
0721   E3FF 01E0                SX    NULL,NULL,M7R3,GETB1,C BRANCH: NO SIGNIF IN R3      32527410
0722   33FD 5001                NI    NULL,PSW,1          ALREADY NEGATIVE ?              32527420
0723   07E0 0B00                BALNZ (MR6)(NULL)         BRANCH: YES.                    32527430
0724   33BD 7002                OI    PSW,PSW,2           SET G FLAG.                     32527440
0725   03F8 0B00                BAL   (MR6)(NULL)         RETURN.                         32527450

                  * ROUTINE ACCUMULATES SIGNIFICANCE, TESTS FOR VALID DATA                32527470
0000 0726         DIGITCK  EQU   *                                                        32527480
0726   33F2 0009                SI    NULL,MR2,'09'       VALID DECIMAL NIBBLE ?          32527490
0727   17E9 CA40                BALNG DIGC1(NULL)         BRANCH: YES.                    32527500
0728   33BD 7008                OI    PSW,PSW,8           QUEUE C-FLAG                    32527510
0729   E063 7920  DIGC1         OX    M7R3,M7R3,MR2,GETB1 ACCUMULATE SIGNIFICANCE, B27:31. 32527520

                  * ROUTINE CHECKS (MR2) FOR VALID SIGN DIGIT, QUEUES FLAGS.              32527540
                  * STANDARD SIGN DIGIT RETURNED IN MR0.                                  32527550
0000 072A         SGN.CHK  EQU   *                                                        32527560
072A   33F0 6003                XI    NULL,MR0,'03'       PLUS ?                          32527570
072B   13E1 CE00                BALZ  SGPLUS(NULL)        BRANCH: YES.                    32527580
072C   33F0 0009                SI    NULL,MR0,'09'       ILLEGAL SIGN DIGIT ?            32527590
072D   13E9 CBC0                BALG  SGN.CK1(NULL)       BRANCH: NO.                     32527600
072E   33BD 7008                OI    PSW,PSW,'08'        SET C FLAG.                     32527610
072F   F3E0 5020  SGN.CK1       NI    NULL,M7R0,'20'      "ABSOLUTE" SPEC'D BY C BIT OF XOP ? 32527620
0730   17E1 CE00                BALNZ SGPLUS(NULL)        BRANCH: YES.                    32527630
0731   33F0 600B                XI    NULL,MR0,'0B'       NEGATIVE ?                      32527640
0732   13E1 CD40                BALZ  SGMINUS(NULL)       BRANCH: YES.                    32527650
0733   33F0 600D                XI    NULL,MR0,'0D'       NEGATIVE ?                      32527660
0734   17E1 CE00                BALNZ SGPLUS(NULL)        BRANCH: NO.                     32527670
                  *                                       ILLEGAL TREATED AS PLUS HERE.   32527680
0735   321F 100D  SGMINUS       LI    MR0,'0D'            SIGN DIGIT                      32527690
0736   33BD 7001                OI    PSW,PSW,1           QUEUE L FLAG IN R3              32527700
0737   03F8 0B00                BAL   (MR6)(NULL)         RETURN                          32527710
0738   321F 100C  SGPLUS        LI    MR0,'0C'            SIGN DIGIT                      32527720
0739   03F8 0B00                BAL   (MR6)(NULL)         RETURN.                         32527730
```

STRING INSTRUCTIONS

```
                          2776  * ROUTINES STORE REPRESENTATION OF ZERO TO OUTPUT STRING               32527750
073A   12D8 8F00          2777  PUSHP    BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)   32527760
073B   229F 1FBE          2778           LX    MR4,NULL,PUSH       DATA '00'                            32527770

073C   12D8 8F00          2780  PUSHU    BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)   32527790
073D   329F 1030          2781           LI    MR4,'30'            DATA '30'                            32527800

0000 073E                 2783  PUSH     EQU   *                                                        32527820
073E   12D9 C840          2784           BAL   PMVFLAGS(MR6)       COLLECT QUEUED FLAGS (P.64)          32527830
073F   E3E4 1241          2785  PUSH.0   AX    NULL,M7R4,M7R4,PUSH.1,C BRANCH: LEN1 NOT < 0             32527840
0740   17FD DCC0          2786           BALD  IIPEXIT(NULL)       ZERO IIP BIT, EXIT (P.67)            32527850
0741   16DD C780          2787  PUSH.1   BALD  STORBYTE(MR6)       STORE DATA BYTE @ END1 (P.64)        32527860
0742   E884 2F80          2788           SDEC  M7R4,M7R4,NULL      DECREMENT LEN1                       32527870
0743   12DD CFD0          2789           BALA  PUSH.0(MR6),D       SERVICE ANY INTERRUPT                32527880

                          2791  * ROUTINES TEST INPUT DATA FOR ZERO WHEN LEN2 >= 0, LEN1 < 0            32527900
0000 0744                 2792  FLUSHU   EQU   *                   CALLED BY UMV. FLUSHES INPUT DATA.   32527910
0744   12D9 C840          2793           BAL   PMVFLAGS(MR6)       COLLECT QUEUED FLAGS (P.64)          32527920
0745   23F2 2FF2          2794           SDECX NULL,MR2,NULL,ADDCC4,C BRANCH: OVERFLOW CASE.            32527930
0746   E01F 2F8A          2795           SDECX M7R0,NULL,NULL,FLUSH BYTE MASK 'FF'                      32527940

0000 0747                 2797  FLUSHP   EQU   *                   CALLED BY PMV. FLUSHES INPUT DATA.   32527960
0747   12D9 C840          2798           BAL   PMVFLAGS(MR6)       COLLECT QUEUED FLAGS (P.64)          32527970
0748   23F4 2FF2          2799           SDECX NULL,MR4,NULL,ADDCC4,C BRANCH: OVERFLOW CASE     (P.67)  32527980
0749   F01F 100F          2800           LI    M7R0,'0F'           DIGIT MASK                           32527990

074A   E8A5 2F80          2802  FLUSH    SDEC  M7R5,M7R5,NULL      THIS BYTE HAS BEEN FETCHED           32528010
074B   12D8 8F00          2803           BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)   32528020
074C   E3E5 12CE          2804  FLUSH.0  AX    NULL,M7R5,M7R5,FLUSH.1,C BRANCH: LEN2 NOT < 0            32528030
074D   17FD DCC0          2805           BALD  IIPEXIT(NULL)       ZERO IIP BIT, EXIT (P.67)            32528040
074E   16DD C680          2806  FLUSH.1  BALD  GETBYTE(MR6)        GET BYTE @ END2 (P.64)               32528050
074F   EBF4 5000          2807           N     NULL,MR4,M7R0       TEST INPUT DATA WITH MASK            32528060
0750   17E1 DC80          2808           BALNZ ADDCC4(NULL)        BRANCH: OVERFLOW CASE. (P.67)        32528070
0751   E8A5 2F80          2809           SDEC  M7R5,M7R5,NULL      DECREMENT LEN2 BY 1                  32528080
0752   12DD D310          2810           BALA  FLUSH.0(MR6),D      LOOP, ALLOW INTERRUPT.               32528090
```

STRING INSTRUCTIONS

```
                         * FOR THE FOLLOWING UNPACK ROUTINES, THE FOLLOWING CONVENTIONS HOLD:    32528110
                         * ALL: AT INPUT, STRING LENGTHS ONE GREATER THAN SPECIFIED IN           32528120
                         *                THE USER-LEVEL INSTRUCTION ARE ASSUMED. NO             32528130
                         *                MAXIMUM IS IMPOSED. UNPACK PROCEEDS RIGHT-TO-LEFT.     32528140
                         *                AN ODD NUMBER OF PACKED DATA DIGITS IS ALWAYS          32528150
                         *                PROCESSED.                                             32528160

                         *                NO INTERRUPT RESULTS FROM AN INVALID SIGN OR           32528180
                         *                DATA DIGIT.                                            32528190

                         *                STANDARD SIGN DIGITS 'C' OR 'D' ARE PRODUCED.          32528210
                         *                A STRING REPRESENTING ZERO HAS SIGN DIGIT 'C'.         32528220

0000 0753                UMV      EQU  *                   (8C/04)                               32528240
0000 0753                UMVA     EQU  *                   (8C/24)                               32528250
0753   12D8 8F00                  BAL  SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)    32528260
0754   E87F 1FA0                  L    M7R3,NULL,E         PRESET FLAGS = 0                      32528270
0755   12D9 C680                  BAL  GETBYTE(MR6)        GET BYTE @ END2 (P.64)                32528280
0756   2A1F 1900                  L    MR0,MR2             SIGN DIGIT TO MR0                     32528290
0757   12D9 CA80                  BAL  SGN.CHK(MR6)        CHECK FOR VALID SIGN DIGIT (P.64)     32528300
0758   3210 9004                  SLLI MR0,MR0,4           MOVE SIGN DIGIT LEFT                  32528310
0759   3253 8004                  SRLI MR2,MR3,4           MOVE DATA DIGIT RIGHT, INTO MR2       32528320
075A   12D9 C980                  BAL  DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64)  32528330
075B   2A90 7900                  O    MR4,MR0,MR2         FORM UNPACKED SIGNED BYTE             32528340
075C   12D9 C780                  BAL  STORBYTE(MR6)       STORE @ END1 (P.64)                   32528350
075D   E8A5 2F80                  SDEC M7R5,M7R5,NULL      DECREMENT LEN2 BY 1 IN R5,            32528360
075E   E884 2F80                  SDEC M7R4,M7R4,NULL      DECREMENT LEN1 BY 1 IN R4.            32528370
075F   12D8 8F00                  BAL  SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21)    32528380

0000 0760                UMVLP1   EQU  *                   UNPACK DATA BYTES                     32528400
0760   E3E5 12E2                  AX   NULL,M7R5,M7R5,UMVL.1,C BRANCH: LEN2 NOT < 0              32528410
0761   17FD CF00                  BALD PUSHU(NULL)         LEN2 < 0. OUTPUT '30' (P.65)          32528420
0762   16DD C680         UMVL.1   BALD GETBYTE(MR6)        GET BYTE @ END2 (P.64)                32528430
0763   12D9 C980                  BAL  DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64)  32528440
0764   E3E4 1267                  AX   NULL,M7R4,M7R4,UMVL.2,C BRANCH: LEN1 NOT < 0              32528450
0765   2A5F 1A00                  L    MR2,MR4             GET BOTH PACKED DIGITS TO MR2         32528460
0766   13F9 D100                  BAL  FLUSHU(NULL)        LEN1 < 0. ALL DATA MUST BE 0 (P.65)   32528470
0767   3292 7030         UMVL.2   OI   MR4,MR2,'30'        APPEND ZONE.                          32528480
0768   12D9 C780                  BAL  STORBYTE(MR6)       STORE ZONED BYTE @ END1 (P.64)        32528490
0769   3253 8004                  SRLI MR2,MR3,4           POSITION (HIGH) PACKED DIGIT          32528500
076A   12D9 C980                  BAL  DIGITCK(MR6)        CHECK (MR2) FOR VALID DECIMAL (P.64)  32528510
076B   EB84 2F80                  SDEC MAR,M7R4,NULL       (LEN1 - 1)                            32528520
076C   13E5 D100                  BALL FLUSHU(NULL)        BRANCH: LEN1 EXHAUSTED. (P.65)        32528530
076D   3372 7030                  OI   WMDR,MR2,'30'       APPEND ZONE.                          32528540
076E   EB81 1E1B                  A    MAR,M7R1,MAR,DW1    STORE @ A(STR1) + (LEN1-1)            32528550
076F   F084 0002                  SI   M7R4,M7R4,2         DECREMENT LEN1 BY 2 IN R4,            32528560
0770   E8A5 2F80                  SDEC M7R5,M7R5,NULL      DECREMENT LEN2 BY 1 IN R5             32528570
0771   12DD D810                  BALA UMVLP1(MR6),D       LOOP; ALLOW INTERRUPT.                32528580
```

```
                        2862  * ROUTINE ADDS V FLAG TO CURRENT CC, EXITS FROM INTERRUPTIBLE          32528610
                        2863  * INSTRUCTION.                                                         32528620
 0772   33BD 7004       2864  ADDCC4   OI    PSW,PSW,4           SET V FLAG                          32528630

                        2866  * THIS ROUTINE ZEROS PSW BIT 14, AND LOADS CLOC WITH THE               32528650
                        2867  * INCREMENTED VALUE COMPUTED ON ORIGINAL ENTRY TO THE INTERRUPTIBLE   32528660
                        2868  * INSTRUCTION, BEFORE TERMINATING TO FETCH THE NEXT USER-LEVEL        32528670
                        2869  * INSTRUCTION.                                                         32528680

 0773   37BD 5101       2871  IIPEXIT  NI    PSW,PSW,BIT140,I    ZERO IIP BIT                        32528700
 0774   EB5F 1500       2872           L     CLOC,M7R10          GET INCREMENTED LOC                 32528710
 0775   2BFF 1F92       2873  EXIT67   L     NULL,NULL,IRD       AND EXIT.                           32528720
```

```
                     * FOR THE FOLLOWING LPB ROUTINE, THE FOLLOWING CONVENTIONS HOLD:          32528740
                     *              THE PACKED DECIMAL STRING IS ASSUMED TO BE OF LENGTH       32528750
                     *              16 BYTES, CONTAINING 31 DATA DIGITS AND A TRAILING         32528760
                     *              SIGN DIGIT.                                                32528770

                     *              LEGAL DATA DIGITS ARE ALL BCD DIGITS 0 THROUGH 9.          32528790
                     *              LEGAL POSITIVE SIGN DIGITS ARE 3, A, C, E, F.              32528800
                     *              LEGAL NEGATIVE SIGN DIGITS ARE B, D.                       32528810
                     *              A STRING REPRESENTING ZERO MAY HAVE EITHER A POSITIVE      32528820
                     *              OR NEGATIVE SIGN DIGIT.                                    32528830

                     *              IF CONVERSION OF THE PACKED DECIMAL NUMBER RESULTS         32528850
                     *              IN ARITHMETIC OVERFLOW, THE DESTINATION REGISTERS          32528860
                     *              ARE UNCHANGED, BUT NO INTERRUPT OCCURS.                    32528870

                     *              IF AN INVALID SIGN DIGIT IS ENCOUNTERED, A DATA            32528890
                     *              FORMAT FAULT, REASON = 2, OCCURS.                          32528900

                     *              IF AN INVALID DATA DIGIT IS ENCOUNTERED, A DATA            32528920
                     *              FORMAT FAULT, REASON = 3, OCCURS.                          32528930

                     *              THE LARGEST NUMBER THAT MAY BE PROCESSED IS                32528950
                     *              +/- 9 223 372 036 854 775 807.                             32528960

0000 0776            LPB1     EQU   *                       * 6F *                             32528980
0776   2B9A 1D80              A     MAR,YX,RMDR             OPERAND ADDRESS               R02  32528990
0777   E9BF 1E00              L     M7R13,MAR               OPERAND ADDRESS                    32529000
0778   E81F 1F05              L     M7R0,YDI,RFAULT         ENTRY YD SELECT; RESET PX FLOPS    32529010
0779   E9FF 1FA0              L     M7R15,NULL,E            ZERO CC, ACCUMULATOR R15           32529020
077A   E9DF 1F80              L     M7R14,NULL              AND R14                            32529030
077B   E95C 1F80              A     M7R10,CLOC,NULL         INCREMENTED LOCATION COUNTER VALUE 32529040
077C   F19F 100F              LI    M7R12,15                R12 COUNT = [(31 DIGITS+SIGN)/2]-1 32529050
                     *                                      ONLY 19 DIGITS MAY CONTAIN DATA.   32529060
077D   12D8 8F00              BAL   SET.RTN(MR6)            SET INTERRUPT RETURN IN R11 (P.21) 32529070

0000 077E            LPB1A    EQU   *                                                          32529090
077E   EB9F 168B              L     MAR,M7R13,DR1           A(OPERAND BYTE) FROM R13; FETCH.   32529100
077F   2A7F 1D80              L     MR3,RMDR                                                   32529110
0780   3253 8004              SRLI  MR2,MR3,4               GET MS DIGIT OF BYTE               32529120
0781   12D9 E7C0              BAL   TENXPLUS(MR6)           GET (ACCUMULATOR*10)+(MR2) (P.70)  32529130
0782   3253 500F              NI    MR2,MR3,'0F'            GET LS DIGIT OF BYTE               32529140
0783   E18C 2FC5              SDECX M7R12,M7R12,NULL,LPB1B,C DECREMENT DIGIT COUNT             32529150
0784   23FF 1F89              LX    NULL,NULL,LPB2          BRANCH: SIGN DIGIT COMING. (P.69)  32529160
0785   12D9 E7C0     LPB1B    BAL   TENXPLUS(MR6)           GET (ACCUMULATOR*10)+(MR2) (P.70)  32529170
0786   E9AD 3F80              AINC  M7R13,M7R13,NULL        INCREMENT A(STRING) IN R13         32529180
0787   12DD E210              BALA  LPB1C(MR6),D            ALLOW INTERRUPT                    32529190
0788   17FD DF80     LPB1C    BALD  LPB1A(NULL)             AND LOOP.                          32529200
```

STRING INSTRUCTIONS

```
0000 0789              2923  LPB2      EQU   *                   PROCESS SIGN NIBBLE, DO OUTPUT      32529220
0789   33F2 6003       2924            XI    NULL,MR2,'03'       POSITIVE SIGN ?                     32529230
078A   13E1 E500       2925            BALZ  PLUSBIN(NULL)                                           32529240
078B   33F2 0009       2926            SI    NULL,MR2,'09'       VALID SIGN ?                        32529250
078C   17E8 9C40       2927            BALNG FORFAUL2(NULL)      BRANCH: INVALID SIGN DIGIT   (P.24) 32529260
078D   33F2 600B       2928            XI    NULL,MR2,'0B'       MINUS ?                             32529270
078E   13E1 E440       2929            BALZ  MINUSBIN(NULL)      BRANCH: IS 'B'                      32529280
078F   33F2 600D       2930            XI    NULL,MR2,'0D'       MINUS ?                             32529290
0790   17E1 E500       2931            BALNZ PLUSBIN(NULL)       BRANCH: NOT 'D'                     32529300
0000 0791              2932  MINUSBIN  EQU   *                                                       32529310
0791   E1FF 07D3       2933            SX    M7R15,NULL,M7R15,MINB1,C 2'S COMP R15,                  32529320
0792   E9CE 3F80       2934            AINC  M7R14,M7R14,NULL    PROPAGATE CARRY                     32529330
0793   E9DF 0700       2935  MINB1     S     M7R14,NULL,M7R14    2'S COMP R14.                       32529340
0000 0794              2936  PLUSBIN   EQU   *                                                       32529350
0794   EA5F 1780       2937            L     MR2,M7R15           LS DATA                             32529360
0795   EA3F 1700       2938            L     MR1,M7R14           MS DATA                             32529370
0796   F3C0 1001       2939            AI    YDI,M7R0,1          FIGURE ENTRY R1+1 FROM R0           32529380
0797   2B3F 1900       2940            L     YD,MR2              LS DATA OUT                         32529390
0798   EBDF 1000       2941            L     YDI,M7R0            RESTORE ENTRY SELECT                32529400
0799   2B3F 18A0       2942            L     YD,MR1,E            MS DATA OUT. SET CC.                32529410
079A   13E5 DCC0       2943            BALL  IIPEXIT(NULL)       EXIT IF NEGATIVE. (P.67)            32529420
079B   2BF1 7900       2944            O     NULL,MR1,MR2        ANY SIGNIFICANCE ?                  32529430
079C   13E1 DCC0       2945            BALZ  IIPEXIT(NULL)       BRANCH: NO. (P.67)                  32529440
079D   33BD 7002       2946            OI    PSW,PSW,2           SET G FLAG                          32529450
079E   17FD DCC0       2947            BALD  IIPEXIT(NULL)       ZERO IIP BIT, EXIT (P.67)           32529460
```

STRING INSTRUCTIONS

```
0000 079F          TENXPLUS EQU   *                   ACCUMULATES BCD AS BINARY
079F   33F2 0009            SI    NULL,MR2,'09'       VALID DECIMAL DIGIT IN MR2 ?
07A0   13E8 9C80            BALG  FORFAUL3(NULL)      BRANCH: INV DATA DIGIT, PACKED (P.24)
07A1   2A9F 1F80            L     MR4,NULL            PREZERO FOR TENXPLO
07A2   F3EC 000B            SI    NULL,M7R12,11       FAST OVERFLOW CHECK POSSIBLE ?
07A3   13F1 E940            BALC  TENXPLA(NULL)       BRANCH: NO. BYTE COUNT LOW.
07A4   23FF 0938            SX    NULL,NULL,MR2,TENXPL2 LEADING BYTE MUST BE ZERO.
07A5   EABF 1700   TENXPLA  L     MR5,M7R14           HIGH HALF ACCUMULATED DATA
07A6   EBEE 7780            O     NULL,M7R14,M7R15    ANY DATA ACCUMULATED YET ?
07A7   13E1 ED00            BALZ  TENXPLO(NULL)       BRANCH: NO. JUST ADD THIS ONE.
07A8   3295 E00A            MI    MR4,MR5,10          TIMES 10
07A9   23F4 2FF9            SDECX NULL,MR4,NULL,TENXOVF,C BRANCH: OVERFLOW.
07AA   E9DF 1A80            L     M7R14,MR5           KEEP PARTIAL RESULT
07AB   EABF 1780            L     MR5,M7R15           LOW HALF ORIGINAL DATA
07AC   32FF 1001            LI    MR7,1
07AD   22B5 8BEF            SRLX  MR5,MR5,MR7,TENXPLB,C SHIFT LOW HALF RIGHT 1
07AE   2AFF 1F80            L     MR7,NULL            IF CARRY-OUT FROM MR5, MR7 IS ZERO.
07AF   3295 E014   TENXPLB  MI    MR4,MR5,20          ORIGINAL NUMBER TIMES 10
07B0   23F7 2FF4            SDECX NULL,MR7,NULL,TENXPLO,C BRANCH: NO ADJUST NECESSARY
07B1   32B5 100A            AI    MR5,MR5,10          ADJUST PRODUCT FOR PRE-SHIFT LOSS
07B2   17F1 ED00            BALNC TENXPLO(NULL)       BRANCH: NO CARRY-OUT
07B3   2A94 3F80            AINC  MR4,MR4,NULL        PROPAGATE CARRY TO HIGH HALF
07B4   E1F5 1976   TENXPLO  AX    M7R15,MR5,MR2,TENXPL1,C ADD NEW DIGIT
07B5   2A94 3F80            AINC  MR4,MR4,NULL        PROPAGATE CARRY
07B6   E1CE 1A78   TENXPL1  AX    M7R14,M7R14,MR4,TENXPL2,C ADD HIGH HALVES
07B7   13F9 EE40            BAL   TENXOVF(NULL)       OVERFLOW CASE.
07B8   07E4 0B00   TENXPL2  BALNL (MR6)(NULL)         RETURN IF NOT OVERFLOW
                   *
07B9   17FD DC80   TENXOVF  BALD  ADDCC4(NULL)        SET ONLY V FLAG, EXIT. (P.67)
```

STRING INSTRUCTIONS

```
                          * FOR THE FOLLOWING STBP INSTRUCTION, THE FOLLOWING CONVENTIONS APPLY:
                          *                ANY BINARY NUMBER WHICH MAY BE CONTAINED IN 64
                          *                BITS MAY BE CONVERTED TO DECIMAL AND OUTPUT TO
                          *                A PACKED DECIMAL STRING, OF LENGTH 16 BYTES.
                          *                THIS STRING CONSISTS OF 31 BCD DATA DIGITS, FOLLOWED
                          *                BY A SIGN DIGIT.

                          *                THE SIGN DIGIT IS 'C' FOR ZERO OR POSITIVE DATA,
                          *                AND 'D' FOR NEGATIVE DATA.


07BA    0000 0000                    DC    FREEWORD              ALIGNMENT FOR TRANSFERS BELOW

0000 07BB                 STBP1      EQU   *                     * 6E *
07BB    2B9A 1D80                    A     MAR,YX,RMDR           OPERAND ADDRESS                     R02
07BC    E83F 1E00                    L     M7R1,MAR              OPERAND ADDRESS
07BD    E95C 1F80                    A     M7R10,CLOC,NULL       INCREMENTED LOCATION COUNTER VALUE
07BE    E99F 1F85                    L     M7R12,NULL,RFAULT     THIS REGISTER USED ONLY TO OUTPUT ZER
                          *                                      RESET RX FLOPS.
07BF    F01F 101F                    LI    M7R0,31               COUNT FOR 31 DIGITS & SIGN TO R0

07C0    2A9B 1F80                    A     MR4,YDP1,NULL         LS HALF INPUT DATA
07C1    2A3F 1CA0                    L     MR1,YD,E              MS HALF INPUT DATA; SET CC
                          *                                      USED TO GENERATE SIGN DIGIT
07C2    17E5 F180                    BALNL STBP.POS(NULL)        BRANCH: ALREADY 'POSITIVE'
0000 07C3                 STBP.NEG   EQU   *                     MUST 2'S COMPLEMENT
07C3    229F 0A45                    SX    MR4,NULL,MR4,STBP.N1,C LOW HALF
07C4    2A31 3F80                    AINC  MR1,MR1,NULL          PROPAGATE CARRY
07C5    223F 0889         STBP.N1    SX    MR1,NULL,MR1,STBP.001 HIGH HALF

07C6    E851 7A00         STBP.POS   O     M7R2,MR1,MR4          ANY POSITIVE DATA ? ADJUST R2.
07C7    13E0 4B00                    BALZ  STBP.ZIP(NULL)        BRANCH: "FAST EXIT" FOR ZERO (P.11)
07C8    33BD 7002                    OI    PSW,PSW,2             SET G FLAG

0000 07C9                 STBP.001   EQU   *                     CONVERT TO BASE (10**8)
07C9    E8FF 1A00                    L     M7R7,MR4              LS HALF POS DATA TO R7
07CA    E8BF 1880                    L     M7R5,MR1              MS HALF POS DATA TO R5
07CB    E89F 1F80                    L     M7R4,NULL             ZERO TO R4
07CC    12D8 8F00                    BAL   SET.RTN(MR6)          SET INTERRUPT RETURN IN R11 (P.21)

07CD    F485 F6AF                    DI    M7R4,M7R5,TENUP8,I    DIVIDE (0:W1) IN (R4:R5) BY (10**8)
                          *                                      RETURNS (RE1:Q1) IN (R4:R5)
07CE    E3E4 1252                    AX    NULL,M7R4,M7R4,STBP.002,C ARITHMETIC COMPARE: IF
                          *                                      RE1 < (10**8)/2 THEN AVOID CORRECTION
07CF    F7E4 06B0                    SI    NULL,M7R4,HALFTEN8,I BOTH POSITIVE:
07D0    13F1 F480                    BALC  STBP.002(NULL)        BRANCH: RE1 < COMPARAND.
07D1    F484 06AF                    SI    M7R4,M7R4,TENUP8,I    GET (RE1-10**8), AVOID O'FLOW.

0000 07D2                 STBP.002   EQU   *
07D2    EADF 1280                    L     MR6,M7R5              SAVE Q1 TO USE BELOW
07D3    E8DF 1200                    L     M7R6,M7R4             LOAD R6 WITH RE1 FROM R4
07D4    F4C7 F6AF                    DI    M7R6,M7R7,TENUP8,I    DIVIDE (RE1:W2) IN (R6:R7)
                          *                                      BY (10**8). RETURNS
                          *                                      (RE2:Q2) IN (R6:R7).
```

```
0000 07D5                  3033 STBP.003 EQU   *                                                  32530320
07D5    E3E6 1358          3034          AX    NULL,M7R6,M7R6,STBP.004,C BRANCH: RE2 NOT NEGATIVE     32530330
07D6    F4C6 16AF          3035          AI    M7R6,M7R6,TENUP8,I  CORRECTION: RE2 = RE2 + (10**8)    32530340
07D7    E8E7 2F80          3036          SDEC  M7R7,M7R7,NULL      Q2 = Q2 - 1                        32530350

0000 07D8                  3038 STBP.004 EQU   *                                                  32530370
07D8    E9FF 1300          3039          L     M7R15,M7R6          RE2 TO R15 FOR OUTPUT..            32530380
07D9    E8DF 1B00          3040          L     M7R6,MR6            RESTORE SAVED Q1 FROM ABOVE        32530390
07DA    F4C7 F6AF          3041          DI    M7R6,M7R7,TENUP8,I  DIVIDE (Q1:Q2) IN (R6:R7) BY       32530400
                           3042 *                                  (10**8); RETURNS (RE3:Q3) IN (R6:R7).32530410

07DB    E9DF 1300          3044          L     M7R14,M7R6          RE3 TO R14                         32530430
07DC    E9BF 1380          3045          L     M7R13,M7R7          Q3 TO R13                          32530440
                           3046 **                                 FOR '7FFF FFFF FFFF FFFF' INPUT,   32530450
                           3047 **                                 R15 = 0343 CFFF                    32530460
                           3048 **                                 R14 = 0202 8830                    32530470
                           3049 **                                 R13 = 0000 039A                    32530480
07DD    12D8 8F00          3050          BAL   SET.RTN(MR6)        SET INTERRUPT RETURN IN R11 (P.21) 32530490
07DE    323F 10FF          3051          LI    MR1,'FF'            SET 'BYTE READY TO OUTPUT'         32530500


0000 07DF                  3053 STBPLP2  EQU   *                                                  32530520
07DF    E85F 1F80          3054          L     M7R2,NULL           PRE-ZERO OUTPUT REGISTER (R2)      32530530
07E0    F280 70E0          3055          OI    MR4,M7R0,'E0'       'SIGN-EXTEND' COUNT FROM R0        32530540
0000 07E1                  3056 STBPLP2B EQU   *                                                  32530550
07E1    33D4 8003          3057          SRLI  YDI,MR4,3           GET DATA REGISTER 12, 13, 14, OR 15 32530560
07E2    2A5F 1F80          3058          L     MR2,NULL            PRE-ZERO HIGH HALF                 32530570
07E3    EA7F 1C80          3059          L     MR3,M7YD            ACCESS DATA REGISTER               32530580
07E4    3253 F00A          3060          DI    MR2,MR3,10          DIVIDE BY 10                       32530590
                           3061 *                                  RETURNS (REM:QUOT) IN (MR2:MR3)    32530600
07E5    E842 7900          3062          O     M7R2,M7R2,MR2       APPEND CURRENT DECIMAL DIGIT TO R2 32530610
07E6    EB3F 1980          3063          L     M7YD,MR3            RETAIN QUOTIENT IN DATA REGISTER   32530620
0000 07E7                  3064 STBPLP2C EQU   *                                                  32530630
07E7    3231 60FF          3065          XI    MR1,MR1,'FF'        REVERSE FLAG                       32530640
07E8    13E1 FB40          3066          BALZ  STBPL2.1(NULL)      BRANCH: GOT A BYTE.                32530650
07E9    F280 0001          3067          SI    MR4,M7R0,1          COMPUTE (COUNT-1)                  32530660
07EA    3294 70E0          3068          OI    MR4,MR4,'E0'        'SIGN-EXTEND' COUNT                32530670
07EB    F042 A004          3069          RRLI  M7R2,M7R2,4         SLIDE DIGIT OUT OF PATH OF NEXT    32530680
07EC    13F9 F840          3070          BAL   STBPLP2B(NULL)      GO GET SECOND DIGIT FOR BYTE       32530690

0000 07ED                  3072 STBPL2.1 EQU   *                   OUTPUT DATA BYTE                   32530710
07ED    F17F 17EE          3073          LI    M7R11,*+1           UPDATE INTERRUPT RETURN ADDRESS    32530720
07EE    12D9 FC80          3074          BAL   STBPSTOR(MR6)       STORE LEAST SIGNIF DATA BYTE (P.73) 32530730
07EF    F17F 17F1          3075          LI    M7R11,*+2           NEW INTERRUPT RETURN               32530740
07F0    12DC 8FD0          3076          BALA  WINDOW(MR6),D       SERVICE ANY INTERRUPT              32530750
07F1    223F 1F9F          3077          LX    MR1,NULL,STBPLP2    SET 'BYTE NOT READY', LOOP.        32530760
```

STRING INSTRUCTIONS

```
                            3079  * ROUTINE DECIDES IF SIGNED BYTE OR DATA BYTE. STORES BYTE                32530780
                            3080  * AT A(STRING)+(COUNT/2). EXITS WHEN COUNT EXHAUSTED.                     32530790
                            3081  *                                                                         32530800
0000 07F2                   3082  STBPSTOR EQU   *                                                          32530810
07F2   F282 B004            3083           RLLI  MR4,M7R2,4          POSITION BYTE TO B24:31 AS REQ'D.      32530820
07F3   F3E0 601F            3084           XI    NULL,M7R0,31        SIGNED BYTE ?                          32530830
07F4   17E1 FE40            3085           BALNZ STBPS.1(NULL)       BRANCH: NO.                            32530840
0000 07F5                   3086  STBP.SGN EQU   *                   OUTPUT SIGNED BYTE                     32530850
07F5   327D 5001            3087           NI    MR3,PSW,'01'        CAPTURE CC L FLAG                      32530860
07F6   3273 700C            3088           OI    MR3,MR3,'0C'        FORM SIGN DIGIT, '0C' OR '0D'          32530870
07F7   2A93 7A00            3089           O     MR4,MR3,MR4         APPEND FIRST DATA DIGIT                32530880
07F8   E800 3F80            3090           AINC  M7R0,M7R0,NULL      SET COUNT TO 32 TEMPORARILY IN R0      32530890
07F9   F260 0001            3091  STBPS.1  SI    MR3,M7R0,1          GET (COUNT-1) FROM R0                  32530900
07FA   3273 8001            3092           SRLI  MR3,MR3,1           DIVIDE BY 2                            32530910
07FB   EB81 1980            3093           A     MAR,M7R1,MR3        A(STRING BYTE)                         32530920
07FC   2B7F 1A1B            3094           L     WMDR,MR4,DW1        STORE BYTE                             32530930
07FD   F000 0002            3095           SI    M7R0,M7R0,2         DECREMENT COUNT IN R0                  32530940
07FE   03E8 0B00            3096           BALG  (MR6)(NULL)         RETURN IF POSITIVE                     32530950
07FF   17FD DCC0            3097           BALD  IIPEXIT(NULL)       ELSE, EXIT (P.67)                      32530960

                            3102           ENDC                                                             32531010

0800                        3104           END                                                              32531030
```

STRING INSTRUCTIONS

ASSEMBLED BY MICROCAL II (32BIT) NO ASSEMBLY ERRORS

A	0000	0054					
ABL	0000	00CA					
ABL.010	0000	045D	1818				
ABL1	0000	0461	361				
ACCUM	0000	0361	1531	1552	1618		
ACCUM1	0000	0362	1485				
AD	0000	00F4					
ADDCC1	0000	0522	2063				
ADDCC4	0000	0772	2794	2799	2808	2977	
ADR	0000	0074					
AE	0000	00D4					
AER	0000	0054					
AFAUL0	0000	025F	1971				
AFAUL1	0000	0260	1044	1977	1981		
AFAUL2	0000	0261	1070				
AFAUL3	0000	0262					
AFAUL4	0000	0263	2474				
AFAULT	0000	0266	1048	1049	1050	1052	1055
AH	0000	0094					
AHI	0000	0194					
AHM	0000	00C2					
AHM1	0000	04D4	349				
AI	0000	01F4					
AIS	0000	004C					
AL	0000	01AA					
AL1	0000	0421	705				
AL2	0000	0429	1732	1734			
AL3	0000	042F	1742				
AL4	0000	0431	1736	1740			
ALGFAULT	0000	0278	1084	1085			
AM	0000	00A2					
AR	0000	0014					
ASCHEX	0000	0365					
ASCHEX1	0000	0372	1478				
ATL	0000	00C8					
ATL.010	0000	0459	1796				
ATL.020	0000	045E	1822				
ATL1	0000	0451	358				
AUTO1	0000	05DE	2279	2282			
AUTOIO	0000	0286	1117				
B	0000	0035	253	256			
BAL	0000	0082					
BAL2	0000	0085	251				
BALR	0000	0002					
BALR1	0000	0005	57				
BBIT	0000	0008	1243	1292	2032	2045	
BBS	0000	0041	151	157			
BBS1	0000	0043	152				
BDCS	0000	01CA					
BFBS	0000	0044					
BFC	0000	0086					
BFCR	0000	0006					

STRING INSTRUCTIONS

BFFS	0000 0046									
BFS	0000 0045	154	160							
BFS1	0000 0047	158								
BI00.07	0000 0049	897								
BI00.15	0000 0015	505								
BI00.16	0000 0053	1978								
BI0031	0000 0081	2421								
BI01.31	0000 0073	112	208							
BI02.14	0000 013B	1990								
BI03.050	0000 000B	2011								
BI0305	0000 053F	2062								
BI08.15	0000 0055	1987								
BI10.14	0000 0013	2015								
BI15.31	0000 0057	1995								
BI16.20	0000 0071	2016								
BI16.30	0000 0039	1159	1178	1230						
BI16.31	0000 0017	583	609	1012	1784	2022	2074	2087	2514	
BI17.31	0000 0075	1908	1958	1961	1968	2522				
BI1820	0000 004D	1144								
BIT00	0000 005B	94	219	928						
BIT01	0000 004F	2000	2253							
BIT04	0000 0521	2060								
BIT08	0000 0059	2002								
BIT080	0000 0065	2012								
BIT13	0000 0009	1674	1692	2207	2255					
BIT14	0000 013D	962	976	1672	1690					
BIT140	0000 0101	1082	2871							
BIT15	0000 000F	1912								
BIT16	0000 0019	994	1909	1959	1962	1969	1979	2227		
BIT160	0000 0023	1601								
BIT17	0000 000D	2506	2520							
BIT18	0000 0051	932	1321							
BIT19	0000 00C1	1054								
BIT20	0000 0183	1616	1677	1695	2135	2190	2214			
BIT24	0000 043F									
BIT25	0000 0440									
BIT26	0000 0441									
BIT27	0000 0442									
BIT28	0000 0443									
BIT29	0000 0444									
BIT30	0000 0445									
BIT31	0000 0446									
BOOT	0000 05D3									
BRK	0000 0110									
BRK1	0000 011A	468								
BRR	0000 0007	59	62							
BTABLE	0000 043F	1752								
BTBS	0000 0040									
BTC	0000 0084									
BTCR	0000 0004									
BTFS	0000 0042									
BUFSW	0000 02E0									
BXH	0000 0180									
BXH1	0000 04CB	640								

STRING INSTRUCTIONS

BXLE	0000 0182	
BXLE1	0000 04C1	543
BXLE2	0000 04C9	1936
BXLE3	0000 04D3	1925 1927
BYTEID	0000 02AD	1198
C	0000 00B2	
C0F01	0000 0121	2072
C1RI	0000 0031	668 813
C1RX	0000 0029	275 323
C2	0000 0031	80 115
C3	0000 002F	130
CA001	0000 0123	2085
CBIT	0000 0010	1273
CBT	0000 00EE	
CCS	0000 01D0	
CCS1	0000 05E6	763
CCW	0000 0012	1180 1181 1184 1188 1203 1211 1243 1260 1262 1273 1292 1298
CD	0000 00F2	
CDADSDMD	0000 010B	421 424 427 430 433
CDR	0000 0072	
CE	0000 00D2	
CER	0000 0052	
CH	0000 0092	
CHANEL	0000 0298	901 1153
CHANEND	0000 02F6	903
CHI	0000 0192	
CHVR	0000 0024	
CHVR1	0000 04B8	109
CHVR2	0000 04BF	1913
CI	0000 01F2	
CL	0000 00AA	
CLB	0000 01A8	
CLB1	0000 015B	702
CLH	0000 008A	
CLHI	0000 018A	
CLI	0000 01EA	
CLOOP	0000 030F	1370
CLR	0000 000A	
COLD.1	0000 05A3	2211
COLD.2	0000 05AA	2219
COLD.3	0000 05AE	2215 2222 2225
COLDSTRT	0000 05A0	2172
COMBIT	0000 0436	406 409 412 415
COMMON	0000 02B2	1208 1213
COMMON2	0000 02DD	
COMMON3	0000 02DE	1267 1302
COMSWAP	0000 0140	622 859 1059 1090
COMSWAP2	0000 0254	612
CONSEND	0000 03D0	909
CONSER	0000 0300	487 845 910 1613 2201 2233 2264 2280
CONST4E	0000 0011	475
CONSTCE	0000 0021	473
COUNT	0000 0014	1195 1202 1208 1208 1210 1218 1249 1251 1290 2035 2037 2049

STRING INSTRUCTIONS

CPAN	0000 06DC	2549									
CPAN1	0000 06E1	2662									
CPAN10	0000 06F3	2639									
CPAN2	0000 06E9	2674									
CPAN20	0000 06F7	2637									
CPAN3	0000 06EA	2668									
CPAN4	0000 06EF	2646									
CPAN5	0000 06F2	2661									
CPANP	0000 06DC										
CR	0000 0012										
CRC	0000 0006	1281	2327	2343							
CRC12	0000 00BC										
CRC121	0000 0540	338									
CRC12B	0000 0552	2076									
CRC16	0000 00BE										
CRC161	0000 0547	341									
CRC16B	0000 054A	1279	2080								
CRCCK	0000 02D5	1274									
CRLF	0000 0385	1375	1382	1396	1459	1495	1540	1587	1600		
D	0000 00BA										
D1	0000 0087	335									
D2	0000 00A5	257									
DAT	0000 0013	1139	1142	1151	1152	1159	1178	1184	1185	2079	2086
DD	0000 00FA										
DDR	0000 007A										
DE	0000 00DA										
DECODE	0000 031F	1464									
DECODE1	0000 0320	1394									
DECODE2	0000 0366	1490									
DECTAB	0000 034B	1393	1479	1488	1500						
DECTABE	0000 035B	1386	1473								
DER	0000 005A										
DEV	0000 0011	745	1120	1123	1126	1134	1134	1139	1142	1149	1150
DFAULT	0000 025B	145	297								
DFAULTX	0000 00A7	304									
DFAULTY	0000 00A1	305									
DH	0000 009A										
DH1	0000 04EC	287									
DH2	0000 04F0	1967									
DHR	0000 001A										
DHR1	0000 04ED	92									
DIGC1	0000 0729	2751									
DIGITCK	0000 0726	2696	2710	2717	2834	2845	2852				
DR	0000 003A										
EBIT	0000 0080	1181									
ECS	0000 01D2										
ENTRY	0000 0316	1594									
EPFINT	0000 020E	894									
EPSR	0000 012A										
EPSR1	0000 0127	509									
EXAUTO	0000 02BC	1196	1219	1250	1291						
EXBR	0000 0128										
EXBR1	0000 0102	506									

EXHR	0000 0068				
EXIT10	0000 0100				
EXIT12	0000 0164				
EXIT15	0000 01B8				
EXIT17	0000 01E5	758			
EXIT25	0000 0297				
EXIT27	0000 02BB				
EXIT29	0000 02F4				
EXIT3	0000 0037	63			
EXIT36	0000 03EF				
EXIT37	0000 0403				
EXIT39	0000 043E				
EXIT42	0000 04A6				
EXIT43	0000 04DA	1955			
EXIT45	0000 0539				
EXIT47	0000 0546				
EXIT5	0000 0069	155	161		
EXIT52	0000 05FB				
EXIT53	0000 0602				
EXIT54	0000 0618	2387			
EXIT57	0000 066D				
EXIT6	0000 009D	254			
EXIT67	0000 0775				
EXIT7	0000 00A4				
EXIT9	0000 00E9				
EXSUB	0000 02B7	1222			
EXSUB0	0000 02B7	1182			
EXSUB1	0000 02B6	1186			
EXSUB2	0000 02B5	1294			
FASTMODE	0000 02A3				
FAULT	0000 020F	848			
FAULT.0	0000 0211	846			
FAULT.1	0000 0220	904			
FAULT.2	0000 0224	902	905		
FAULT.3	0000 0228	914			
FAULT.4	0000 022E	918			
FBIT	0000 0001	1188	2043		
FIVES	0000 007B	2160	2163	2240	
FLDR	0000 007E				
FLDR1	0000 0111	242			
FLR	0000 005E				
FLR1	0000 0111	196			
FLR2	0000 0115	472			
FLR3	0000 0116	474			
FLUSH	0000 074A	2795			
FLUSH.0	0000 074C	2810			
FLUSH.1	0000 074E	2804			
FLUSHP	0000 0747	2721			
FLUSHU	0000 0744	2848	2854		
FMTCMD	0000 00EE	1358			
FORFAUL2	0000 0271	2927			
FORFAUL3	0000 0272	2951			
FORFAUL6	0000 0275	920			
FORFAUL7	0000 0276				

STRING INSTRUCTIONS

```
FORFAULT  0000 0277           1083
FPPFAUL   0000 026B            890
FRD       0000 02B1           1212
FREEWORD  0000 0000            147   213   214   215   216   231   395   438   453   454   489   499
                               516   519   522   634   634   634   634   641   665   690   693   696
                               707   708   717   726   751   752   810  1033  1096  1097  2479  2479
                              2479  2479  2479  2479  2479  2479  2479  2479  2479  2479  2479  2479
                              2479  2479  2479  2479  2479  2479  2479  2479  2479  2479  2989
FWT       0000 02B0
FXDR      0000 007C
FXDR1     0000 062F            240
FXDR2     0000 063D           2408
FXR       0000 005C
FXR1      0000 0639            194
FXR3      0000 0644           2418
FXR4      0000 064B           2429
FXR5      0000 064D           2433
GET.END   0000 06BA           2556
GETB1     0000 0720           2732  2742  2753
GETBYTE   0000 071A           2693  2709  2806  2829  2844
GETPAD    0000 06B6           2579  2632
GO.BY.6   0000 01D1            755   767
HALFTEN8  0000 06B0           3022
HARDSTOP  0000 02FE            907
HEXASC    0000 037D
HWASSIST  0000 02D7
HWIO      0000 02A7
HWRD      0000 02AB           1204
HWRT      0000 02AA
HWRT1     0000 02AC           1205
IDLE      0000 03D1           1368  1519  1520  1612
IF.DELE   0000 0341           1425
IIPCHECK  0000 0238            387   390   480
IIPEXIT   0000 0773           2657  2786  2805  2943  2945  2947  3097
IIPFAUL   0000 0273           1080  1081
IIPRESUM  0000 023A
ILEGAL    0000 0209             52   486   828   849  1630  2220  2300  2531
INCHR     0000 032B           1385  1411  1423  1436  1450  1451  1452  1462  1469  1584
INCMD     0000 0021           1360
INDEV     0000 0010           1357  1401
IOINT0    0000 0280            844
IOINT1    0000 0281            843
IOINT2    0000 0283            842
IOINT3    0000 0285            841
IOINTX    0000 028B           1120  1123  1126  1135
IS.AT     0000 0395           1442
IS.D      0000 03B1           1448
IS.D00    0000 03BA           1581
IS.DELE   0000 0345           1429
IS.F      0000 03AA           1447
IS.F00    0000 03AE           1566
IS.MINUS  0000 0398           1444
IS.P      0000 03BE           1449
IS.PL0    0000 039A           1533
```

IS.PLUS	0000 0399	1443 1549
IS.PRMPT	0000 03C9	1405 1441
IS.R	0000 03A4	1446
IS.R00	0000 03A8	1557
ISRST	0000 041C	1638
ISSV	0000 0419	1637
L	0000 00B0	
LA	0000 01CC	
LB	0000 01A6	
LBR	0000 0126	
LCDR	0000 006E	
LCDR1	0000 0656	218
LCER	0000 002E	
LCER1	0000 001B	123
LCS	0000 004A	
LD	0000 00F0	
LD1	0000 0612	419
LDE	0000 010E	
LDE1	0000 0117	466
LDER	0000 014E	
LDGR	0000 014C	
LDGR1	0000 0156	562
LDPS	0000 0404	1636
LDPS1	0000 0413	1693 1696
LDR	0000 0070	
LE	0000 00D0	
LED	0000 0108	
LEDR	0000 0148	
LEDR1	0000 0151	556
LEGR	0000 014A	
LER	0000 0050	
LEVEL	0000 0012	1117 1119 1120 1122 1123 1125 1126 1135 1136 1137 1137 1144
LGDR	0000 002C	
LGDR1	0000 064F	121
LGER	0000 002A	
LH	0000 0090	
LHI	0000 0190	
LHL	0000 00E6	
LHL1	0000 015D	404
LI	0000 01F0	
LIS	0000 0048	
LIST	0000 044C	357 360 363 366
LM	0000 01A2	
LM1	0000 048B	625 692
LM2	0000 04A5	1858
LM71	0000 060E	1691 1714 2188
LM72	0000 060F	2361
LM@	0000 048F	1689 2184
LMD	0000 00FE	
LMD1	0000 0624	440
LMD2	0000 0626	2390
LMD@	0000 062A	1698 2193
LMD@1	0000 062B	2397

STRING INSTRUCTIONS

LME	0000 00E4					
LME1	0000 015F	401				
LME2	0000 0161	590				
LME3	0000 0162	587				
LME@	0000 0165	1697	2192			
LME@1	0000 0166	596				
LMTAB	0000 0490	1857				
LMTABE	0000 04A1	627				
LPB	0000 00DE					
LPB1	0000 0776	391				
LPB1A	0000 077E	2921				
LPB1B	0000 0785	2916				
LPB1C	0000 0788	2920				
LPB2	0000 0789	2917				
LPDR	0000 0066					
LPDR1	0000 0656	207				
LPER	0000 0026					
LPSTD	0000 03EE	1633				
LPSW	0000 0184					
LPSW1	0000 0245	647				
LPSW2	0000 0247	1704	1705			
LPSWR	0000 0030					
LPSWR1	0000 001D	126				
LR	0000 0010					
LRA	0000 00C6					
LRA.PR2	0000 051D	2018				
LRA.PR3	0000 053A	2023				
LRA.PRI	0000 0518	2003				
LRA.SHAR	0000 050F					
LRA1	0000 04FE	355				
LRA1.5	0000 0504	2007				
LRA2	0000 0505	1992				
LRA3	0000 050B	2013				
LSSTD	0000 03F0	1634				
M	0000 00B8					
MATINT	0000 0172	917				
MD	0000 00F8					
MDR	0000 0078					
ME	0000 00D8					
MEM.1	0000 05BE	2248	2250			
MEMLOOP	0000 05BA	2256				
MEMTEST	0000 05B9	2175	2212			
MER	0000 0058					
MH	0000 0098					
MH1	0000 04E3	284				
MH2	0000 04E7	1957				
MHR	0000 0018					
MHR1	0000 04E4	89				
MINB1	0000 0793	2933				
MINUSBIN	0000 0791	2929				
MISMATCH	0000 06EC	2672				
MMFEND	0000 02FE	906				
MMFINT	0000 0230	202	880	883	2203	
MMFINT2	0000 02F5	934				

STRING INSTRUCTIONS

MOVE	0000 06BE	2548						
MOVE1	0000 06C1	2599						
MOVE2	0000 06C9	2585						
MOVE2A	0000 06D0	2598						
MOVEP	0000 06BE							
MR	0000 0038							
MVTU	0000 06BE	2547						
MVTU1	0000 06C5							
MVTU2	0000 06C7	2587						
N	0000 00A8							
NFAST	0000 02BF	1189						
NFREAD	0000 02E3	1261						
NFWRIT	0000 02CA							
NH	0000 0088							
NHI	0000 0188							
NI	0000 01E8							
NOX1	0000 0698	2502						
NOX2	0000 06A0	2511						
NR	0000 0008							
O	0000 00AC							
OC	0000 01BC							
OC1	0000 0135	732						
OCR	0000 013C							
OCR1	0000 0137	525	536					
OH	0000 008C							
OHI	0000 018C							
OI	0000 01EC							
OR	0000 000C							
OUTCHR	0000 0389	1384	1461	1503	1512	1514	1521	
OUTCMD	0000 0023	1363						
OUTDEV	0000 0011	1362	1415	1517				
OUTPUT.A	0000 06DA	2602	2606					
OUTPUT.Z	0000 06D9	2593						
OVERFLO	0000 0641	2430	2431					
PAD.0	0000 06D3	2609						
PAD.1	0000 06EE	2614						
PADIT	0000 06D1	2583						
PB	0000 00C4							
PB1	0000 05FC	352						
PBR	0000 0064							
PBR1	0000 0603	204						
PLUSBIN	0000 0794	2925	2931					
PMV	0000 06FB	2550						
PMVA	0000 06FB							
PMVFLAGS	0000 0721	2784	2793	2798				
PMVL.1	0000 0709	2707						
PMVL.2	0000 0714	2713						
PMVL.3	0000 0716	2720						
PMVLP1	0000 0707	2725						
POW	0000 0574							
PPFINT	0000 0560	847	1403	1610				
PREG.0	0000 0381							
PRNTLF8	0000 0378	1555	1563	1572				
PRNTREG	0000 037A	1494	1505	1546				

STRING INSTRUCTIONS

PRNTREG6	0000 0376	1378 1381 1542 1589 1591
PRNTREG8	0000 0379	1575
PROMPT	0000 031B	1398
PROMPTC	0000 003C	1383 1460
PSF	0000 01BE	
PSF1	0000 03DA	735
PSFTAB	0000 03DF	1627
PUSH	0000 073E	2778
PUSH.0	0000 073F	2789
PUSH.1	0000 0741	2785
PUSHP	0000 073A	2708
PUSHU	0000 073C	2843
PWRDWN	0000 0560	1353
PWRUPINT	0000 059E	2284
QTEST	0000 0248	99 503
QUESTN	0000 0328	1445 1453 1454 1455 1456 1489 1586 1617
R0	0000 0000	1146 1238 1718 1862 1889 2557 2589
R1	0000 0001	1147 1237 1863 1891 2614 2631 2641 2656 2656 2666
R10	0000 000A	1872 1900
R11	0000 000B	629 1873 1901
R12	0000 000C	624 1061 1094 1874 1902
R13	0000 000D	623 1028 1060 1091 1092 1875 1903
R14	0000 000E	551 1030 1664 1667 1672 1674 1876 1904
R15	0000 000F	552 1031 1665 1877 1905
R2	0000 0002	1149 1864 1892 2587 2588
R3	0000 0003	1150 1185 1315 1865 1893
R4	0000 0004	1179 1187 1194 1217 1231 1246 1247 1272 1293 1308 1866 1894
R5	0000 0005	1867 1895
R6	0000 0006	1868 1896
R7	0000 0007	1869 1897
R8	0000 0008	1870 1898
R9	0000 0009	1871 1899
RBL	0000 00CE	
RBL.010	0000 0485	1846
RBL1	0000 047F	367
RBT	0000 00EC	
RD	0000 01B6	
RDCS	0000 05ED	2299
RDCS1	0000 05EE	2308
RDCS2	0000 05F2	2303
RDR	0000 0136	
READIT	0000 05E2	2270 2272 2274 2276
REDCHK	0000 02D0	1266 1301
REL	0000 03E9	1632
RELOAD	0000 0588	
RELOADB	0000 058C	
RETOLOC	0000 01FD	896
RETOLOC1	0000 01FF	827
RETURN	0000 0015	1263 1266 1276 1299 1301 1313 2076 2080 2108
RH	0000 01B2	
RH1	0000 01AE	716
RHR	0000 0132	
RLL	0000 01D6	

STRING INSTRUCTIONS

RMVF	0000 03E7	
RRL	0000 01D4	
RTL	0000 00CC	
RTL.010	0000 047A	1833
RTL.020	0000 047C	1853
RTL1	0000 0470	364
RTRANSL	0000 02EB	1299
RWBIT	0000 0004	1203 1211 1260 2050
RX2	0000 06A5	2505
RX2.BCK	0000 06A8	2521
RX2.FWD	0000 06A7	
RX3	0000 069C	
RXDCODE	0000 06A9	2517
RXEXIT	0000 06A4	2507 2523
RXRX	0000 0118	
RXRX1	0000 0686	481 2477 2478
RXRX2	0000 068F	2491
RXRX3	0000 0695	2497
RXRXTAB	0000 06B1	2528
S	0000 00B6	
SBIT	0000 0020	1273
SBT	0000 00EA	
SCP	0000 01C6	
SCP1	0000 0524	749
SCP2	0000 0528	
SCP3	0000 0532	2042 2044
SCP4	0000 0538	2051
SD	0000 00F6	
SDR	0000 0076	
SE	0000 00D6	
SELFTEST	0000 0575	54
SELFTST1	0000 0575	2154 2155 2156 2157 2158 2169
SER	0000 0056	
SET.RTN	0000 023C	2580 2633 2691 2702 2777 2780 2803 2827 2839 2908 3016 3050
SETCC0	0000 0460	1725
SETCC4	0000 0447	1793 1811 1827 1844 2001 2036
SETCC8	0000 044A	1993 2019
SGMINUS	0000 0735	2766
SGN.CHK	0000 072A	2695 2831
SGN.CK1	0000 072F	2761
SGPLUS	0000 0738	2759 2764 2768
SH	0000 0096	
SHI	0000 0196	
SHOWLOC	0000 0319	
SHOWPSW	0000 0317	
SI	0000 01F6	
SINT	0000 01C4	
SINT1	0000 0287	746
SIS	0000 004E	
SLA	0000 01DE	
SLHA	0000 019E	
SLHL	0000 019A	
SLHLS	0000 0122	

STRING INSTRUCTIONS

SLL	0000 01DA			
SLLS	0000 0022			
SR	0000 0016			
SRA	0000 01DC			
SRHA	0000 019C			
SRHL	0000 0198			
SRHLS	0000 0120			
SRL	0000 01D8			
SRLS	0000 0020			
SS	0000 01BA			
SSR	0000 013A			
ST	0000 00A0			
ST1	0000 009E	296		
START	0000 0001			
STB	0000 01A4			
STB1	0000 01B7	695	729	
STBP	0000 00DC			
STBP.001	0000 07C9	3006		
STBP.002	0000 07D2	3020	3023	
STBP.003	0000 07D5			
STBP.004	0000 07D8	3034		
STBP.N1	0000 07C5	3004		
STBP.NEG	0000 07C3			
STBP.POS	0000 07C6	3002		
STBP.SGN	0000 07F5			
STBP.Z1	0000 012D	513	515	
STBP.ZIP	0000 012C	3009		
STBP1	0000 07BB	388		
STBPL2.1	0000 07ED	3066		
STBPLP2	0000 07DF	3077		
STBPLP2B	0000 07E1	3070		
STBPLP2C	0000 07E7			
STBPS.1	0000 07F9	3085		
STBPSTOR	0000 07F2	514	3074	
STBR	0000 0124			
STBR1	0000 013E	498		
STD	0000 00E0			
STD1	0000 0614	394		
STDE	0000 0104			
STDE1	0000 065E	451		
STDE2	0000 066C	2472		
STE	0000 00C0			
STE1	0000 00FF	345		
STFAIL	0000 020D	200		
STFAIL2	0000 0060	895		
STH	0000 0080			
STH1	0000 009C	247		
STM	0000 01A0			
STM1	0000 04A1	689		
STM71	0000 0609	1673	1709	2134
STM72	0000 060A	2355		
STM@	0000 04A7	1670	2128	
STMD	0000 00FC			
STMD1	0000 0619	436		

STRING INSTRUCTIONS

```
STMD2      0000 061B          2376
STMD@      0000 061F          1680  2138
STMD@1     0000 0620          2382
STME       0000 00E2
STME1      0000 01E0           398
STME2      0000 01E3           791
STME@      0000 01F8          1679  2137
STME@1     0000 01F9           824
STMTAB     0000 04A8          1883
STORBYTE   0000 071E          2699  2722  2787  2836  2850
STPS       0000 03F2          1635
SVC        0000 01C2
SVC1       0000 0169           743
SYSQINT    0000 0251
TBIT       0000 0002          1262  1298
TBT        0000 00E8
TEL        0000 041F          1639
TEMP       0000 0010          1140  1146  1243  1244  1244  1246  1247  1247  1248  1267  1289  1302

TENS       0000 0079          2161  2164
TENUP8     0000 06AF          3018  3024  3029  3035  3041
TENXOVF    0000 07B9          2960  2974
TENXPLO    0000 07B4          2958  2967  2969
TENXPL1    0000 07B6          2971
TENXPL2    0000 07B8          2955  2973
TENXPLA    0000 07A5          2954
TENXPLB    0000 07AF          2964
TENXPLUS   0000 079F          2914  2918
TERMCHAR   0000 06D8          2590
THI        0000 0186
TI         0000 01E6
TLATE      0000 01CE
TLATE1     0000 04DB           761
TLATE2     0000 04E2          1953
TLSU       0000 05CE          2197  2230
TRANSL     0000 02EA
TRAP00     0000 0000
TRAPFF     0000 01FE
TRYMOD     0000 035B          1547  1556  1564  1578  1592
TS         0000 01C0
TSTDFU     0000 03D6          1560  1569
TWAIT      0000 0240           626   628   630   829   933  1330
TWAIT1     0000 0241           860  1011  1017  1031  1061  1093  1094  1239
UFAULT     0000 0264          1051
UMV        0000 0753          2551
UMVA       0000 0753
UMVL.1     0000 0762          2842
UMVL.2     0000 0767          2846
UMVLP1     0000 0760          2859
UNDERFLO   0000 064E          2426
UNLOAD.1   0000 056A          2132
UNLOAD.2   0000 056F
UNLOAD.3   0000 0574          2136
WAIT       0000 0243           995
```

STRING INSTRUCTIONS

WARM.1	0000 058E	2187		
WARM.2	0000 0597	2191		
WARMSTRT	0000 0587			
WD	0000 01B4			
WDCS	0000 05F4	2297		
WDCS1	0000 05F5	2318		
WDCS2	0000 05FA	2312		
WDR	0000 0134			
WH	0000 01B0			
WHR	0000 0130			
WINDOW	0000 023F	968	977	3076
WTRANSL	0000 02EA	1263		
X	0000 00AE			
XH	0000 008E			
XHI	0000 018E			
XI	0000 01EE			
XR	0000 000E			