IV
HARDWARE OPERATION

The material contained in this section describes the hardware for the Central Processor Unit (CPU).

4.1 MASTER CLOCK

The CPU clock (see Figure 4-1) is a two-phase delay line clock that provides one variable and two fixed clock pulses. The variable clock pulse is called Master Clock (MCLOCK); the fixed clock pulses are called Data Cycle Now Permit (MCDCNP) and Local Register Valid (MLRVLD). All three clock pulses are generated by delay lines which are hereafter referred to as the clock cycle generator.

A Master Clock pulse is composed of an 80-nanosecond positive period and a variable length negative period. The negative intervals range from 80- to 240-nanoseconds in duration and are usually selected by bits 20 and 21 of the control store word (CK Field). Combining the selected negative period with the 80-nanosecond positive period produces one complete Master Clock cycle with an overall clock speed as indicated below.

CLOCK SPEED (nanoseconds)	SECOND HALF CYCLE (nanoseconds)	CK FIELD BITS	
		20	21
160	80	1	1
180	100	1	0
200	120	0	1
320	240	0	0

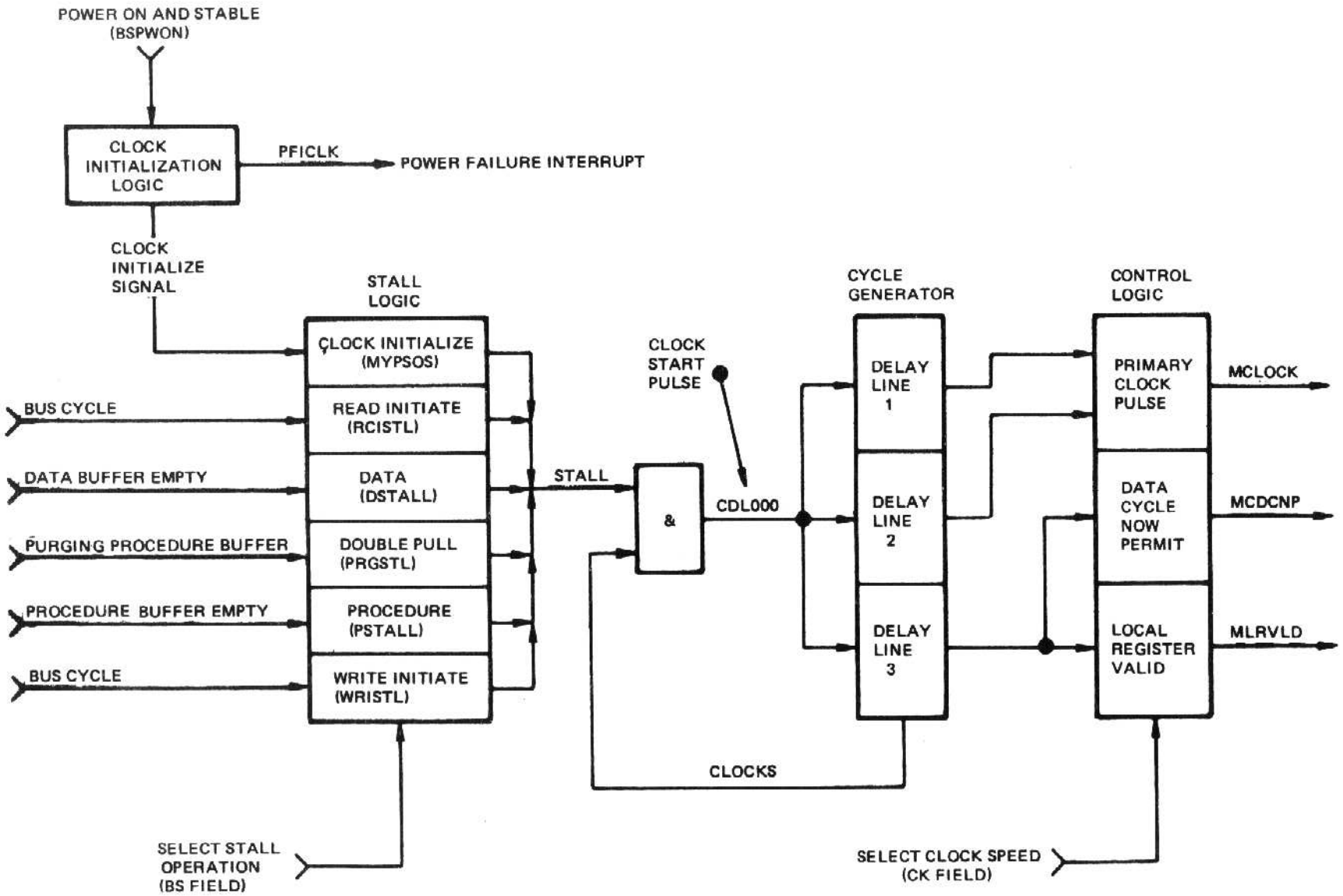


Figure 4-1 Master Clock Logic

A Data Cycle Now Permit pulse indicates that the Megabus address lines are valid; a Local Register Valid pulse indicates that the contents of the control store data register are valid.

The master clock logic can be divided into the following three areas:

- Clock initialization network
- Clock stall network
- Clock cycle generator.

4.1.1 Clock Initialization Network

The clock initialization network monitors the Megabus for a power-up sequence to initialize the CPU and the master clock, and to initiate the necessary CPU housekeeping operations when a power supply malfunction occurs. It performs these functions in conjunction with the clock stall network.

4.1.2 Clock Stall Network

The clock stall network momentarily stalls the clock when additional time is needed to complete one of the following CPU or firmware generated functions:

- A clock initialization that occurs during the CPU power-up sequence.

- A firmware initiated Megabus cycle to which no responding bus activity is expected (e.g., a memory write operation). The clock remains stalled until the selected unit acknowledges or rejects the request.

- A firmware cycle immediately following one in which a Megabus cycle was initiated and to which a response is expected. The clock remains stalled until the selected unit acknowledges or rejects the request.

- A firmware step that calls for data from a bus buffer that has not yet been filled. The clock remains stalled until the applicable buffer is full.

The clock is stalled by inhibiting the clock cycle generator (specifically MCLOCK pulses) when the CPU is initially powered-up, during a read or write cycle initiate, or for single-word or double-word data transfers from an external device (i.e., memory or I/O). The specific type of stall operation is determined by bits 31 through 35 of the control store word (BS Field).

4.1.2.1 Read Cycle Initiate Stalls

A Read Cycle Initiate is defined as a firmware step that requests data from a source external to the CPU (e.g., memory), and is intended to allow firmware to use the data in subsequent

firmware steps. The clock cycle generator is inhibited at the next MCLOCK pulse, preventing any further clock activity until the request is accepted or rejected over the Megabus.

4.1.2.2 Write Initiate Stalls

During a write operation to an external device, the clock is stalled until the device responds with a positive or negative acknowledgment.

4.1.2.3 Data Stalls

A Data Stall operation inhibits MCLOCK pulses to provide any additional time needed to fill the Data (BD) buffer or the Procedure buffer with pertinent data from memory. Two basic signals are used for this purpose. The first, ALOUTD, denotes that the data buffer is empty; the second, ALOUTP, denotes that the procedure buffers are empty. A third signal, PURGEF, inhibits additional double-fetch operations until all previous double-fetch requests have been answered and discarded.

4.1.3 Clock Cycle Generator

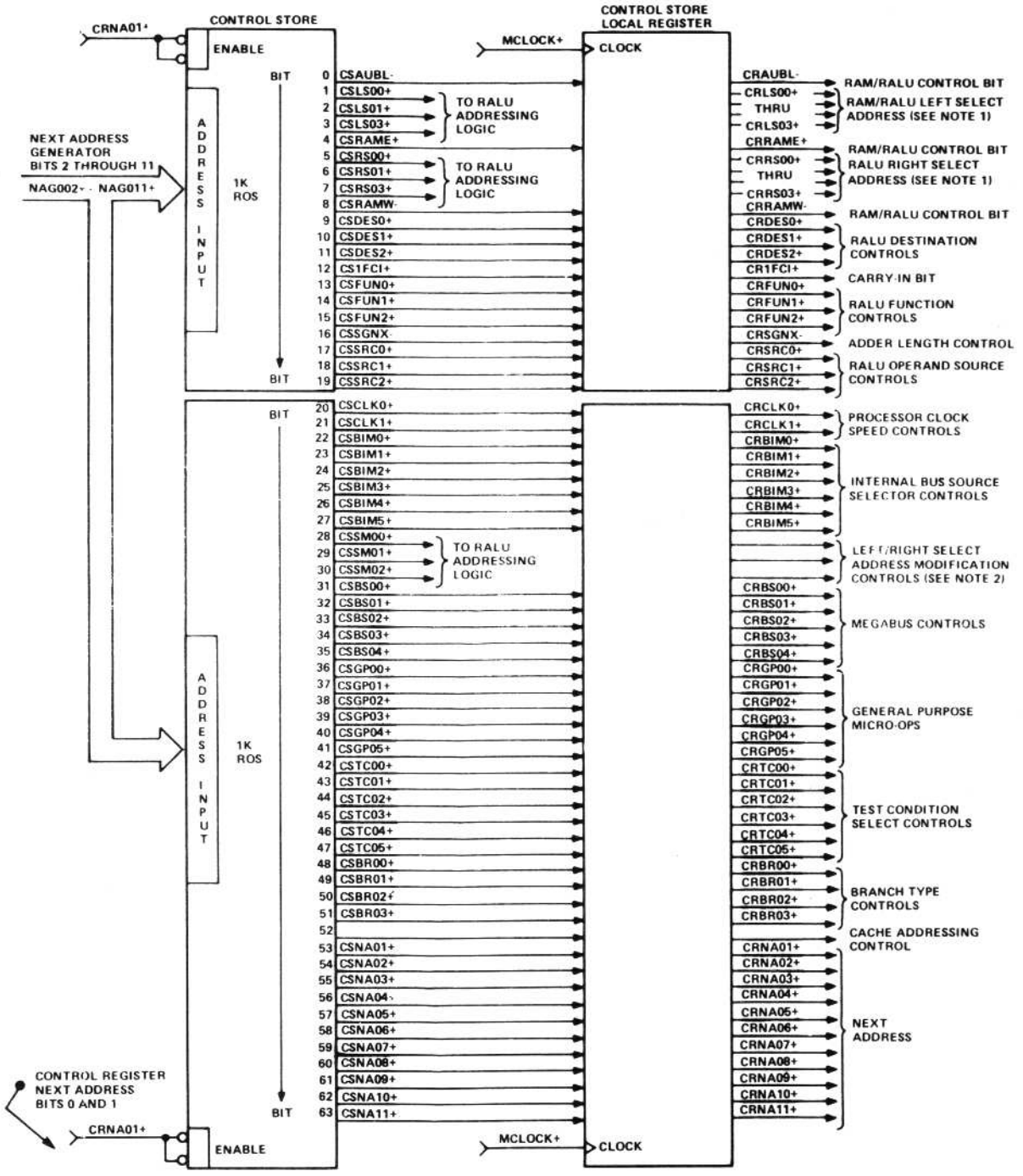
The clock cycle generator provides the selectable clock pulses used throughout the CPU, and consists of three delay lines plus associated control logic. Delay lines 1 and 2 generate the primary Master Clock (MCLOCK) pulse; delay line 3 generates the two previously described secondary clock pulses (MLRVLD and MCDCNP). All delay lines are tapped at selected intervals to generate their respective clock cycles.

4.2 CONTROL STORE

Control within the CPU is provided by the generation of specifically formatted 64-bit control words. Each word is selectively obtained from a 2,048 location Read Only Storage (ROS) memory, which is called the control store. The 64-bit output from the control store forms the input to the Control Store Local Register (CR). Figure 4-2 illustrates the control store logic and includes the control store local register.

4.2.1 Control Store Local Register (CR)

The CR register retains firmware control words that emanate from control store for one complete firmware cycle (i.e., the control store output, or firmware word, is strobed into the local register at the positive transition of the clock input to the register, and it retains this data until new data is available at the positive transition of the next primary clock pulse). Outputs from the CR register are available for distribution throughout the CPU.



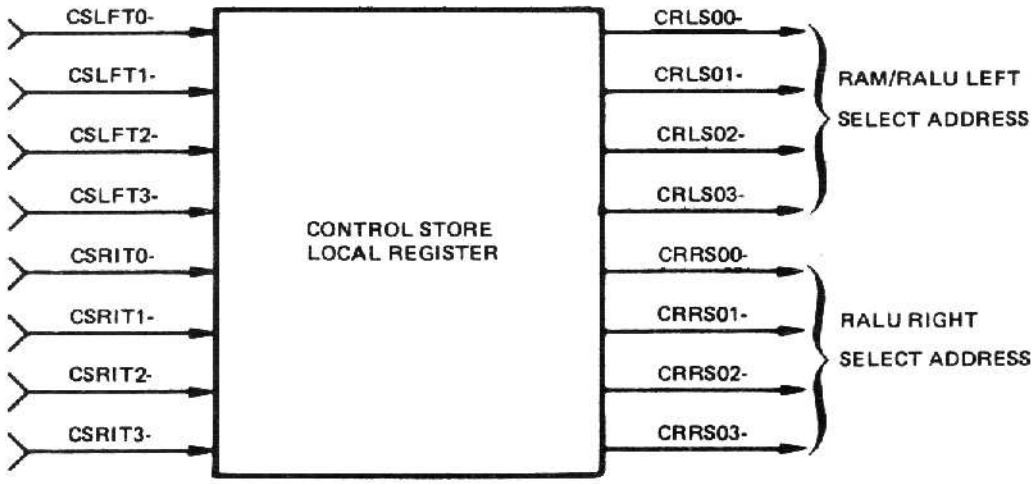
NOTES

1. INPUTS TO THESE SIGNALS ARE OBTAINED FROM THE OUTPUT OF THE RALU ADDRESSING LOGIC (REFER TO SUBSECTION 4.5)

2. THESE SIGNALS ARE NOT LOADED FROM THE OUTPUT OF THE CONTROL STORE. INSTEAD, THE OUTPUTS FROM CONTROL STORE (BITS 28 THROUGH 30) ARE FED TO THE RALU ADDRESSING LOGIC TO PARTICIPATE IN GENERATING RAM/RALU ADDRESSES.

Figure 4-2 Control Store Local Register

The Left and Right Select address bits (i.e., control store bits 1 through 3 and 5 through 7) are not stored directly into the local register. Instead, they are fed to the microprocessor (RALU) addressing logic where they are used in conjunction with the Select Modify bits (i.e., control store bits 28 through 30) to generate the 4-bit left and right select address inputs for the local register. These input signals to the local register and their corresponding output signals are:



4.2.2 Control Store Addressing

Control store addressing is primarily controlled by the next address generation logic (refer to subsection 4.3).

4.3 NEXT ADDRESS GENERATION (NAG) LOGIC

The NAG logic (see Figure 4-3) generates the next firmware address for the control store using one of three methods. All methods use bits 53 through 63 of the firmware word to form a tentative next address. These bits comprise the 11-bit NA field that can directly address any one of the 2,048 control store locations.

Method 1: This method uses as the alternate next address, bits 53 through 61 of the firmware word, in conjunction with logical Ones replacing bits 62 and 63, to form the 11-bit address.

Method 2: This method obtains the alternate next address from the CPU branch logic which generates numerous predefined addresses. The address generated is determined from a decode of the instruction register contents and other control logic.

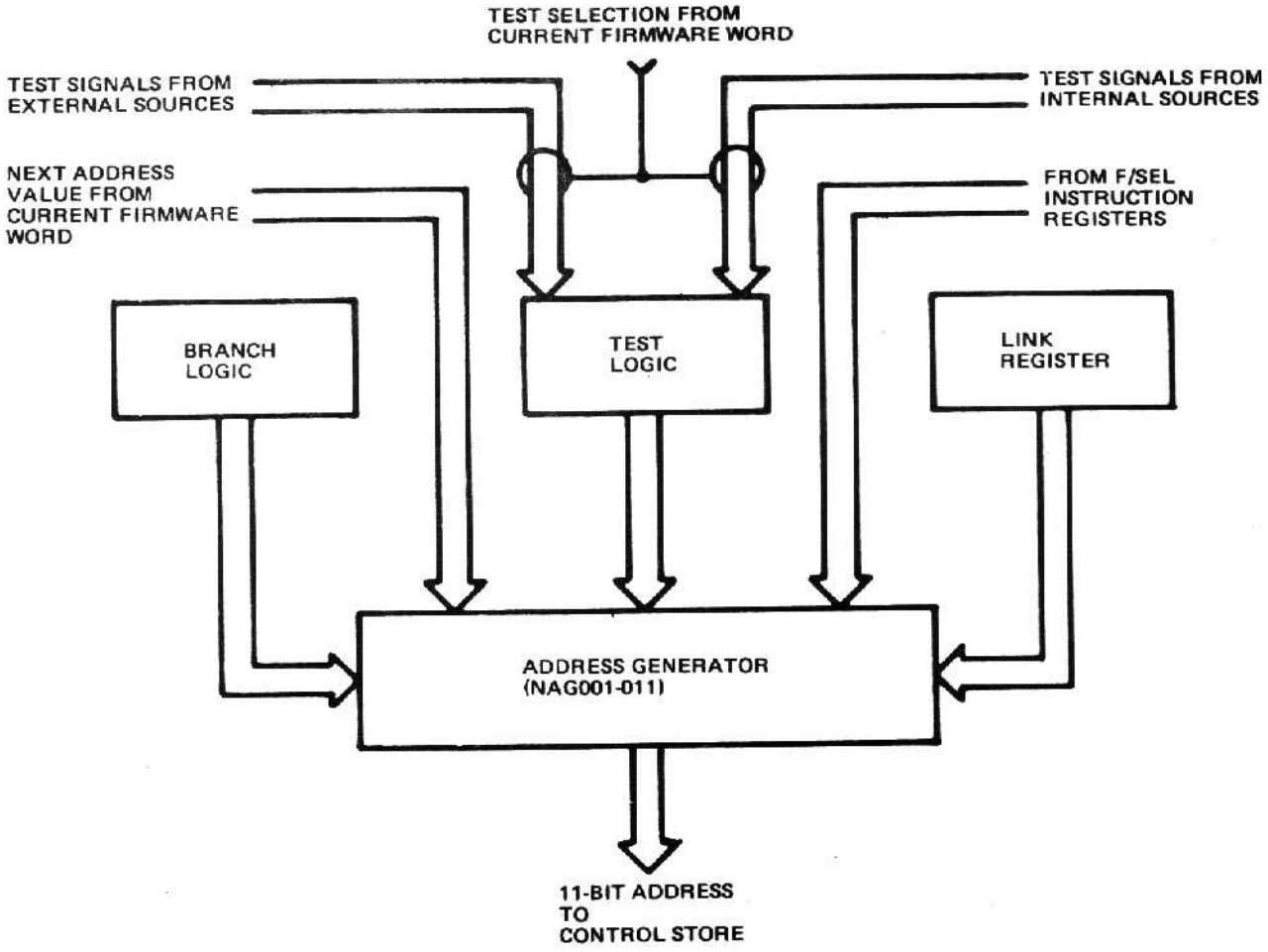


Figure 4-3 Next Address Generation Logic

Method 3: This method uses the 8-bit contents of the LINK register with three constant bits (as shown below) to form the alternate 11-bit address.

The next address generation logic can be divided into four areas: (1) test logic, (2) branch logic, (3) LINK register, and (4) address generator. These logic areas and selected fields of the firmware word make possible the above methods of generating the next firmware address.

4.3.1 Test Logic

The test logic receives inputs from sources both internal and external to the CPU, providing 69 hardware signals that can be used as test conditions. One of the 69 test signals is selected by the TC Field of the current firmware word to participate in generating the next firmware address. These test conditions are too numerous to catalog here, but are defined in Table 3-16.

The output from the test logic (hereafter referred to as the test signal) indicates whether or not the test condition is satisfied, and is fed directly to the address generator. The test signal is used by the address generator to determine whether to substitute the alternate next firmware address for the tentative value (CRNA01-11).

4.3.2 Branch Logic

The branch logic contains several branch PROMs that provide the next firmware address for major branch operations.

4.3.3 LINK Register (XL)

The XL register is an 8-bit firmware address register (not visible to software) that supplies eight bits of the next firmware address when a Link Branch (or subroutine return) type operation is selected; the most significant two bits of the address are forced to 01 and the least significant bit is forced to Zero.

4.3.4 Address Generator

The address generator provides the next firmware address for the control store. The precise manner used to generate the next address is determined by the BR Field of the current firmware word. This field specifies the type of branch operation being performed as a result of a specific test condition. The eight branch types that can be specified include two binary branches (X0 and XL) and six major branches (XA, XB, XR, XE, XW, and XF).

4.3.4.1 X0 Branch

The X0 branch type consists of both unconditional and conditional branches. The alternate next address is derived by performing a logical OR operation between the NA field and a value of 3 (hexadecimal).

4.3.4.2 XL Branch

The XL branch type returns control to the normal firmware sequence after execution of a firmware subroutine. The alternate next address is determined from the contents of the LINK register.

4.3.4.3 XA Branch

The XA branch type is used as the first step in analyzing each instruction. Formation of the alternate next address for XA branches depends on the instruction form.

4.3.4.4 XB Branch

The XB branch type is used to analyze the address syllable portion of the data descriptor for commercial type instructions.

4.3.4.5 XR Branch

The XR branch type is used to fetch an indirect address, perform indexing, read operand(s) from memory, or execute jump type or I/O type instructions.

4.3.4.6 XE Branch

The XE branch type completes the op-code decoding necessary to begin execution of the single- and double-operand instructions included in the CPU instruction repertoire.

4.3.4.7 XW Branch

The XW branch is used to store a result.

4.3.4.8 XF Branch

The XF branch type is used to exit the instruction currently being executed and return to the Instruction Fetch firmware.

4.4 MICROPROCESSOR

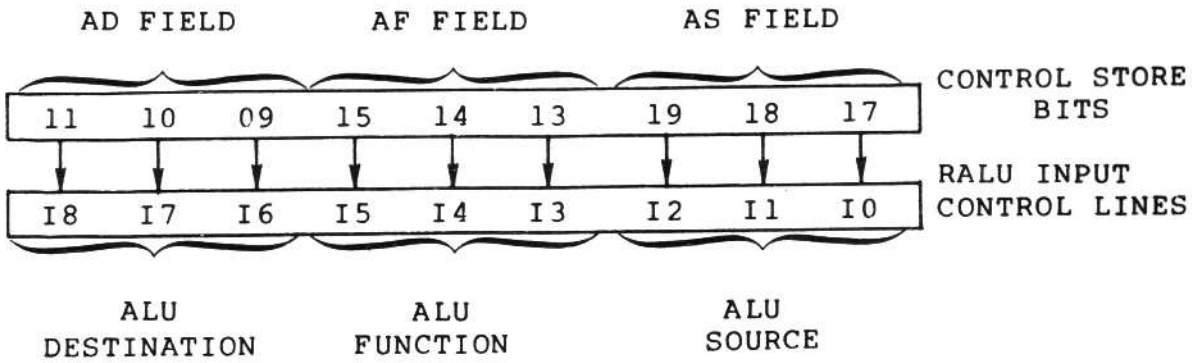
The microprocessor (see Figure 4-4), hereafter referred to as the Register File and Arithmetic Logic Unit (RALU), performs arithmetic, logical, and shift operations as directed by the current firmware word.

The major logic areas of the RALU are:

- Microinstruction decoder
- Register file
- Q register
- Data source selector
- Arithmetic/Logic Unit (ALU)
- Data output selector
- Shift logic.

4.4.1 Microinstruction Decoder

Control store generates a microprocessor instruction by encoding the RALU input control lines (I0 through I8) with the microinstruction code necessary to process data through the RALU. This is accomplished using the AD, AF, and AS fields of the current firmware word as shown below:



4.4.2 Register File

The Register File (RF) is the operand storage facility of the RALU, consisting of 16 registers; two working registers, seven data registers, and seven base registers. Each register is 20 bits wide.

The two working registers (D0 and B0) are not software-visible, and provide a temporary storage facility when manipulating data during firmware operations. These registers are located in RF locations 0 and 8, respectively.

The seven data registers (D1 through D7) are software-visible, representing software registers R1 through R7. These registers are located in RF locations 1 through 7.

The seven base registers (B1 through B7) are software-visible, and are located in RF locations 9 through F.

Of the 16 register file locations, any two can be simultaneously accessed by the firmware, providing dual operands in a single firmware step. The contents of the selected locations (or location, if the addresses are the same) appear as the left and right outputs from the register file. The left output may be routed directly (via the data output selector) to the internal bus source selector, while both the left and right outputs are available as sources for the J and K ports.

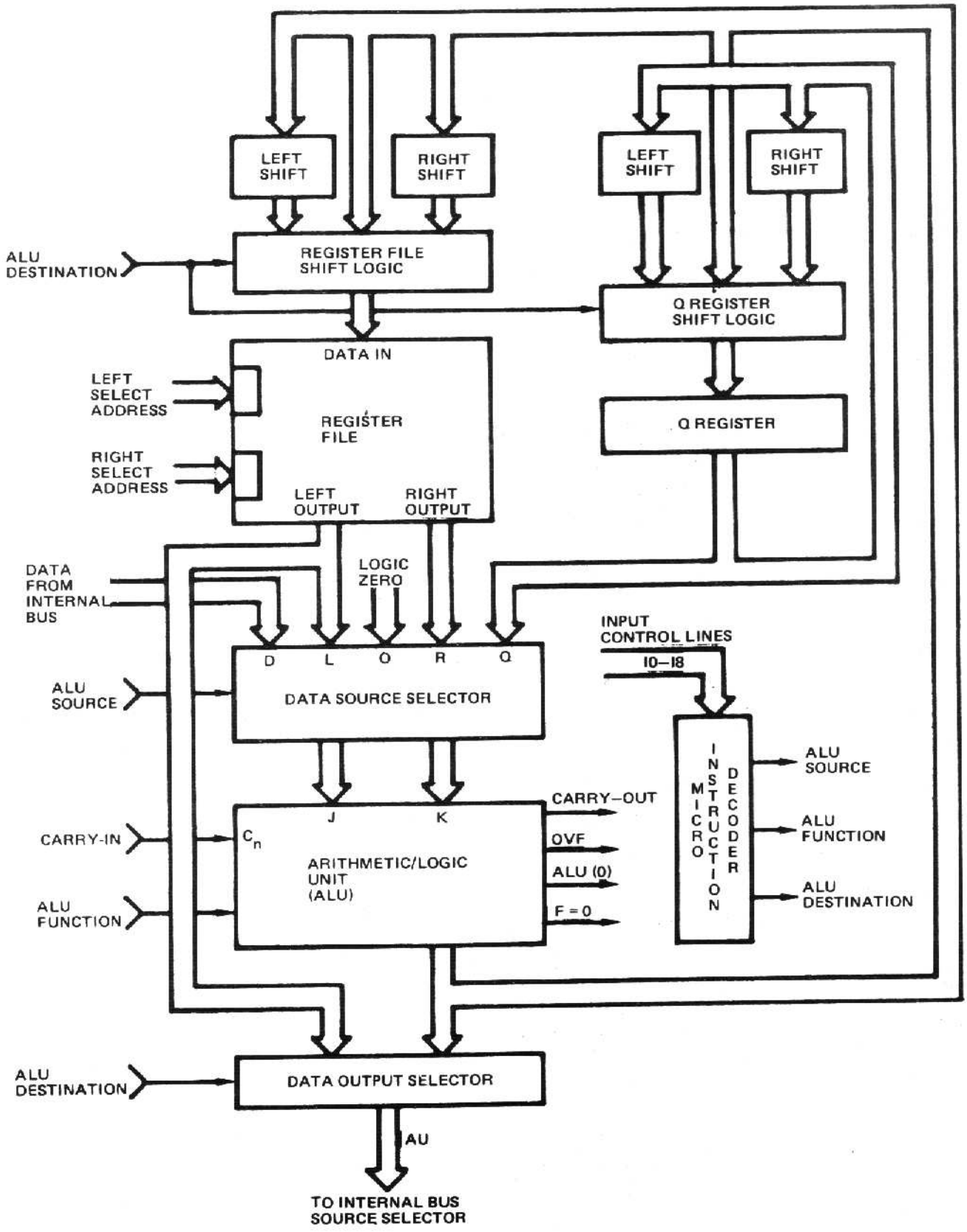


Figure 4-4 Microprocessor Functional Overview

4.4.3 Q Register

The Q register can function as a scratch pad or an extension of any register file register during shift operations and normal transfers of data. This allows, for example, the retention of the least significant half of a double-length product during a multiply operation. Although the Q register is 20 bits wide, only the least significant 16 bits are used during double-precision shift operations.

4.4.4 Data Source Selector

The data source selector is a steering device for data within the RALU, and consists of separate multiplexers for the J and K inputs to the ALU. The sources that serve as inputs to these multiplexers include:

- Register file left and/or right output
- Q register
- Internal bus
- Logical Zero.

The J multiplexer can select any of the above, except the right register file output and the Q register, while the K multiplexer can select any but the internal bus; logical Zero can be selected by either the J or K multiplexer, but not both simultaneously.

4.4.5 Arithmetic/Logic Unit (ALU)

The ALU is the heart of the CPU, performing arithmetic, logical, and compare operations as directed by the firmware. The ALU has two inputs (J and K) that are sourced from the data source selector, and one 20-bit output that may be selected as an input to either the register file or the Q register, and is made available to the internal bus source selector via the data output selector (AU).

The ALU can perform 64 arithmetic and logical operations as directed by RALU input control lines I0 through I5 (see Table 4-1). These operations include the following features:

- Full Carry Look-Ahead
- Overflow Detection
- Result Sign Detection
- All Zeros Detection.

Other ALU outputs that can be tested and/or copied by the firmware include: (1) overflow, (2) carry out, and (3) ALU output equals Zero. These three test signals can relate to an entire 20-bit operation or only the least significant 16 bits.

Table 4-1 RALU Logical and Arithmetic Operations (Sheet 1 of 2)

LOGICAL OPERATIONS							
INSTRUCTION MODIFIER BITS							
I5	I4	I3	I2	I1	I0	GROUP	FUNCTION
0	1	1	0	0	0	OR	L V Q
0	1	1	0	0	1	OR	L V R
0	1	1	0	1	0	Pass	Q
0	1	1	0	1	1	Pass	R
0	1	1	1	0	0	Pass	L
0	1	1	1	0	1	OR	D V L
0	1	1	1	1	0	OR	D V Q
0	1	1	1	1	1	Pass	D
1	0	0	0	0	0	AND	L & Q
1	0	0	0	0	1	AND	L & R
1	0	0	0	1	0	Zero	0
1	0	0	0	1	1	Zero	0
1	0	0	1	0	0	Zero	0
1	0	0	1	0	1	AND	D & L
1	0	0	1	1	0	AND	D & Q
1	0	0	1	1	1	Zero	0
1	0	1	0	0	0	Mask	$\overline{L}$ & Q
1	0	1	0	0	1	Mask	$\overline{L}$ & R
1	0	1	0	1	0	Pass	Q
1	0	1	0	1	1	Pass	R
1	0	1	1	0	0	Pass	L
1	0	1	1	0	1	Mask	$\overline{D}$ & L
1	0	1	1	1	0	Mask	$\overline{D}$ & Q
1	0	1	1	1	1	Pass	0
1	1	0	0	0	0	EX-OR	L ⊻ Q
1	1	0	0	0	1	EX-OR	L ⊻ R
1	1	0	0	1	0	Pass	Q
1	1	0	0	1	1	Pass	R
1	1	0	1	0	0	Pass	L
1	1	0	1	0	1	EX-OR	D ⊻ L
1	1	0	1	1	0	EX-OR	D ⊻ Q
1	1	0	1	1	1	Pass	D
1	1	1	0	0	0	EX-NOR	$\overline{L \veebar Q}$
1	1	1	0	0	1	EX-NOR	$\overline{L \veebar R}$
1	1	1	0	1	0	Invert	$\overline{Q}$
1	1	1	0	1	1	Invert	$\overline{R}$
1	1	1	1	0	0	Invert	$\overline{L}$
1	1	1	1	0	1	EX-NOR	$\overline{D \veebar L}$
1	1	1	1	1	0	EX-NOR	$\overline{D \veebar Q}$
1	1	1	1	1	1	Invert	$\overline{D}$

DEFINITIONS

X = don't care
+ = addition
- = subtraction
& = logical AND
V = logical OR
⊻ = exclusive OR
D = ALU input from internal bus
L = ALU input from L latches
R = ALU input from R latches
Q = ALU input from Q register

Table 4-1 RALU Logical and Arithmetic Operations (Sheet 2 of 2)

FUNCTION			SOURCE			ARITHMETIC OPERATIONS			
INSTRUCTION MODIFIER BITS						C_n = 0 (Low)		C_n = 1 (High)	
I5	I4	I3	I2	I1	I0	GROUP	FUNCTION	GROUP	FUNCTION
0	0	0	0	0	0	Add	L + Q	Add + 1	L + Q + 1
0	0	0	0	0	1	Add	L + R	Add + 1	L + R + 1
0	0	0	0	1	0	Pass	Q	Increment	Q + 1
0	0	0	0	1	1	Pass	R	Increment	R + 1
0	0	0	1	0	0	Pass	L	Increment	L + 1
0	0	0	1	0	1	Add	D + L	Add + 1	D + L + 1
0	0	0	1	1	0	Add	D + Q	Add + 1	D + Q + 1
0	0	0	1	1	1	Pass	D	Increment	D + 1
0	0	1	0	0	0	Subtract	Q - L - 1	Subtract	Q - L
0	0	1	0	0	1	Subtract	R - L - 1	Subtract	R - L
0	0	1	0	1	0	Decrement	Q - 1	Pass	Q
0	0	1	0	1	1	Decrement	R - 1	Pass	R
0	0	1	1	0	0	Decrement	L - 1	Pass	L
0	0	1	1	0	1	Subtract	L - D - 1	Subtract	L - D
0	0	1	1	1	0	Subtract	Q - D - 1	Subtract	Q - D
0	0	1	1	1	1	One's complement	- D - 1	Two's complement	- D
0	1	0	0	0	0	Subtract	L - Q - 1	Subtract	L - Q
0	1	0	0	0	1	Subtract	L - R - 1	Subtract	L - R
0	1	0	0	1	0	One's complement	- Q - 1	Two's complement	- Q
0	1	0	0	1	1	One's complement	- R - 1	Two's complement	- R
0	1	0	1	0	0	One's complement	- L - 1	Two's complement	- L
0	1	0	1	0	1	Subtract	D - L - 1	Subtract	D - L
0	1	0	1	1	0	Subtract	D - Q - 1	Subtract	D - Q
0	1	0	1	1	1	Decrement	D - 1	Pass	D

DEFINITIONS

+ = addition
- = subtraction
& = logical AND
V = logical OR
V = exclusive OR

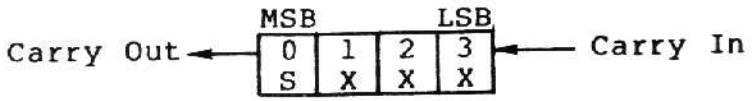
D = ALU input from internal bus
L = ALU input from left output of register file
R = ALU input from right output of register file
Q = ALU input from Q register

4.4.5.1 Look-Ahead Logic

The RALU consists of five stages, with each stage providing a Carry Generate and a Carry Propagate signal. These signals are used in conjunction with an external carry generator (i.e., external to the RALU) to form the look-ahead logic. This logic determines whether or not the Carry Input (CN) to the RALU is propagated through each stage. The determination is based on an interpretation of the input operand rather than awaiting the ripple carry through each stage.

4.4.5.2 Overflow Logic

The overflow logic is internal to the RALU and is used in conjunction with the sign bit during signed arithmetic operations. Overflow reflects the results of an exclusive OR operation between the carry-in and carry-out of the sign bit, and occurs when the result of any addition or subtraction requires more bit positions than the ALU can accommodate. For example, assume that a 4-bit system (three bits plus the sign bit) is used, consisting of one arithmetic unit where the most significant bit is a Zero for positive numbers and a One for negative numbers as shown below:



S = Sign

0: Positive
1: Negative

X = 0/1

Under the preceding conditions, only a maximum value of +7 and a minimum value of -8 is possible. Overflow is detected by comparing the carry-in and carry-out of the sign bit via an exclusive OR operation that is internal to the RALU; if the carries disagree, the overflow signal is not used during logical operations.

4.4.5.3 Zero Detection Logic

The Zero Detect outputs from the least significant four stages of the RALU are effectively tied together, enabling detection of a logical Zero condition for the 20-bit system.

4.4.6 Data Output Selector

The data output selector selects whether the left output from the register file or the ALU output is made available at the output (AU) of the RALU. This output can be used directly in the current firmware step, and is available to the internal bus source selector for distribution to other CPU elements via the internal bus. The output selector also specifies whether the ALU output is to be copied into the register file. If this is done, the register file location modified is the one that supplied the right output to the data input selector.

NOTE

Write operations can be performed only into the right side of the register file.

4.4.7 Shift Logic

The RALU shift logic optionally shifts the output of the ALU (or the ALU and the Q register) before delivering the results to the register file (or the register file and the Q register). The shift logic is capable of shifting single- or double-word operands left or right by one bit position as directed by RALU input control lines I6 through I8 (see Table 4-2).

Table 4-2 RALU Shift Operations

DESTINATION			SHIFT OPERATIONS			
INSTRUCTION MODIFIER BITS			REGISTER FILE FUNCTION		Q REGISTER FUNCTION	
I8	I7	I6	SHIFT	LOAD	SHIFT	LOAD
0	0	0	X	X	None	ALU
0	0	1	X	X	X	X
0	1	0	None	ALU	X	X
0	1	1	None	ALU	X	X
1	0	0	Left	ALU	Left	Q Reg.
1	0	1	Left	ALU	X	X
1	1	0	Right	ALU	Right	Q Reg.
1	1	1	Right	ALU	X	X

X = don't care

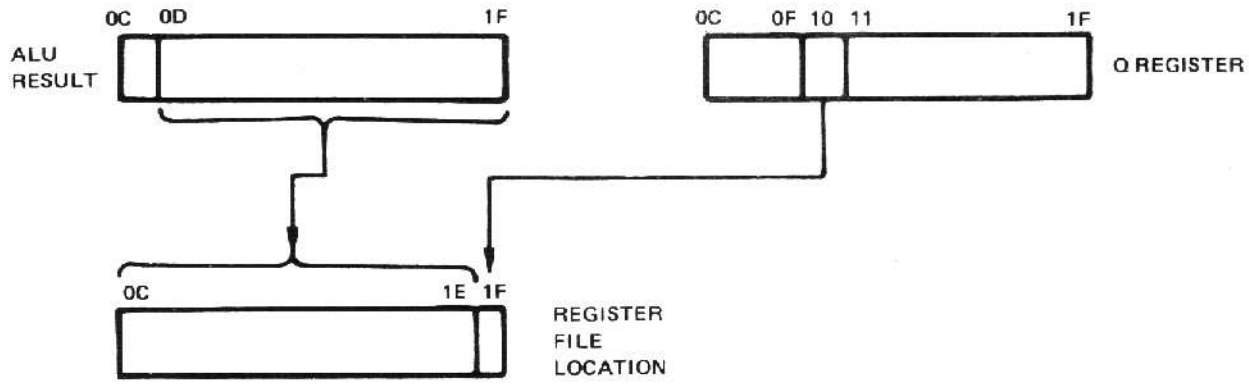
In shift operations, the bit shifted into the vacated bit position is designated as SHIN (shift input), and is controlled by three flip-flops: SHIN1, SHIN2, and MISC (refer to subsection 4.8). The SHIN function is selected as follows:

MISC	SHIN1	SHIN2	SHIN
0	0	0	Internal Bus bit 10
0	0	1	Internal Bus $\overline{\text{bit 10}}$
0	1	0	Zero
0	1	1	Q register bit 1F*
1	0	0	XB register bit 1
1	0	1	Y register bit 7
1	1	0	Zero
1	1	1	Q register bit 1F*

*During shift right operations; otherwise undefined.

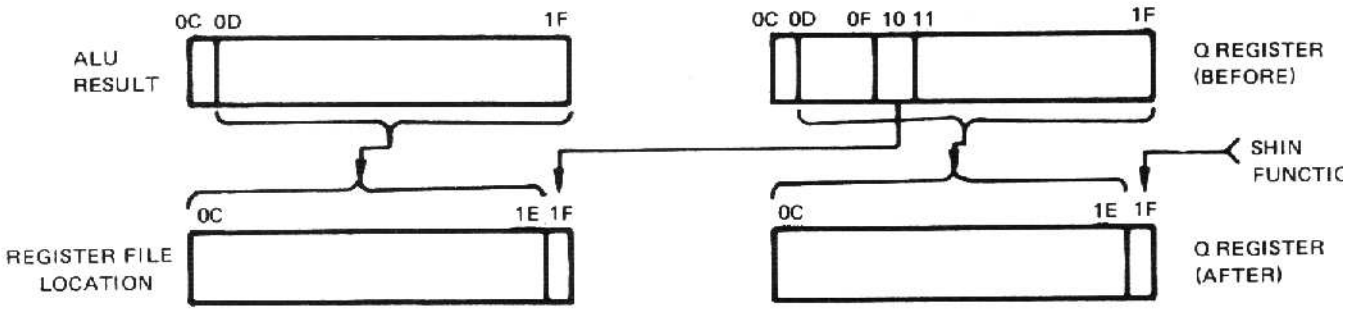
4.4.7.1 Single Left Shift

Bits 0D through 1F of the ALU result are placded in bits 0C through 1E of the selected register file location; bit 1F of the selected register file location receives a copy of Q register bit 10.



4.4.7.2 Double Left Shift

Bits 0D through 1F of the ALU result are placed in bits 0C through 1E of the selected register file location; bit 1F of the selected register file location receives a copy of Q register bit 10. Q register bits 0D through 1F are placed in Q register bits 0C through 1E; Q register bit 1F receives a copy of the SHIN function.

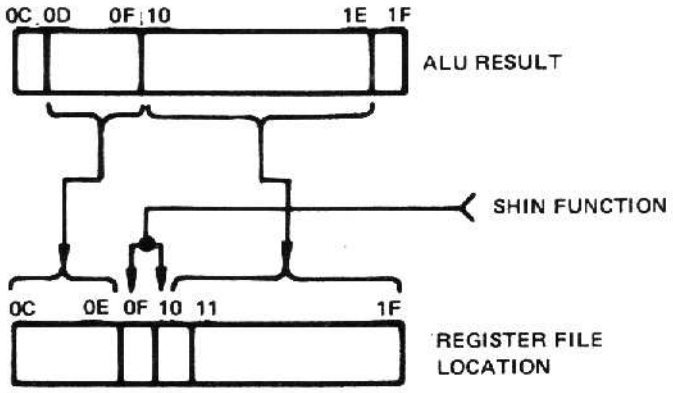


Conceptually, the rightmost 16 bits of the ALU result are concatenated with the rightmost 16 bits of the Q register and shifted left one bit position with the SHIN function shifted in

on the right. The result is placed in the rightmost bit positions of the register file location and the Q register, respectively.

4.4.7.3 Single Right Shift

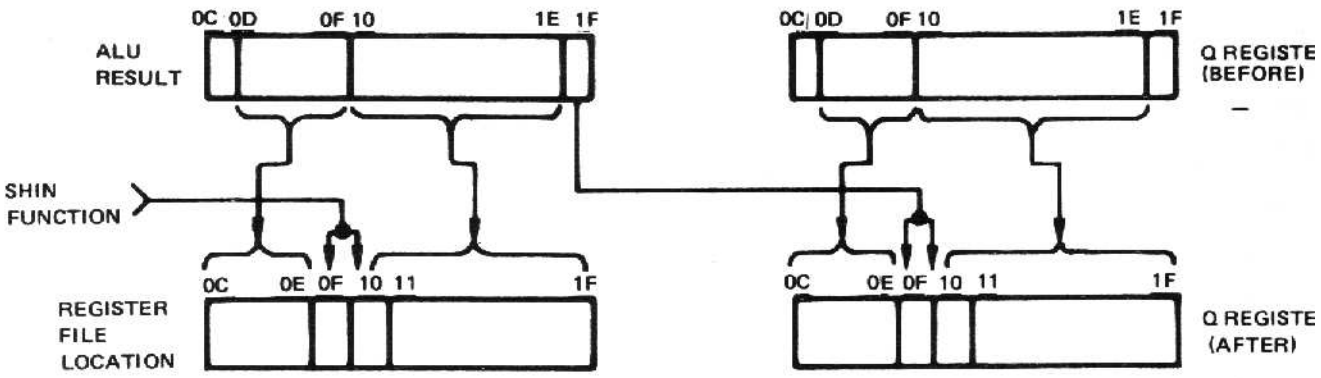
Bits 10 through 1E of the ALU result are placed in bits 11 through 1F of the selected register file location; bit 10 of the selected register file location receives a copy of the SHIN function. Bits 0D through 0F of the ALU result are placed in bits 0C through 0E of the selected register file location; bit 0F of the selected register file location receives a copy of the SHIN function.



Conceptually, the rightmost 16 bits of the ALU result are shifted right one bit position with the SHIN function shifted in on the left. The result is placed in the 16 rightmost bit positions of the register file location.

4.4.7.4 Double Right Shift

Bits 10 through 1E of the ALU result are placed in bits 11 through 1F of the selected register file location; bit 10 of the selected register file location receives a copy of the SHIN function. Q register bits 10 through 1E are placed in Q register bits 11 through 1F; bit 1F of the ALU result is placed in Q register bit 10. Bits 0D through 0F of the ALU result are placed in bits 0C through 0E of the selected register file location; bit 0F of the selected register file location receives a copy of the SHIN function. Q register bits 0D through 0F are placed in Q register bits 0C through 0E; bit 1F of the ALU result is placed in Q register bit 0F.



Conceptually, the least significant 16 bits of the ALU result and the least significant 16 bits of the Q register are concatenated, shifted right one bit position with the SHIN function filling the most significant bit, and the result placed in the least significant 16 bits of the register file location and the Q register, respectively.

4.5 RALU ADDRESSING

The RALU addressing logic can select any one of the 16 registers located within the register file as directed by the LS, RS, and SM fields of the current firmware word. Eight of these registers are directly addressable by the LS and RS fields when the SM field equals zero (see Tables 3-1 and 3-2). Each of the 16 registers can be addressed by using the SM field to modify the LS or RS register address.

The LS, RS, and SM fields are each configured as 3-bit codes that are allocated within the control store as follows:

- LS: bits 1 through 3
- RS: bits 5 through 7
- SM: bits 28 through 30

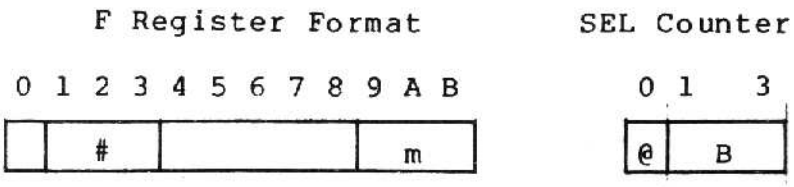
These 3-bit fields generate hexadecimal addresses that deliver the contents of a selected register (or registers) to the left and/or right output of the register file. The register selection performed by the above address fields is defined in Table 3-13.

The LS field also provides addressing for the Random Access Memory (RAM) which is external to the RALU (refer to subsection 4.9.1).

Write operations into the register file can be performed only to a register that is accessed by the right select address lines.

The SM field selects data from one of two sources to modify the LS or RS address:

1. The following F register fields (refer to subsection 1.6.2):



2. One of the following three constants:

   - D: (1,1,0,1)
   - E: (1,1,1,0)
   - F: (1,1,1,1)

These signals are ANDed with the applicable LS or RS field, providing the input signals to the local register for the left and right select addresses.

1.6 INTERNAL BUS

The internal bus (BIXX0C through BIXX1F) provides a 20-bit wide data path that transfers data among elements of the CPU as directed by the BI Field of the current firmware word (see Tables 3-9 through 3-11). The internal bus receives inputs (via the internal bus source selector) from 14 sources, and makes its data available to 12 destinations (see Figure 4-5). These sources and destinations are:

Internal Bus Sources

1. Constant Generators
2. Megabus Data Buffer (BD)
3. Megabus Interrupt Register (RUP)
4. Megabus Procedure 1 Buffer (BP1) or Procedure 2 Buffer (BP2)
5. Control Panel
6. H Register
7. Indicator Register (I)
8. Status/Security Register (S)

9. Program Counter (P), or
   Memory Address Register (Y), or
   MMU Output (Physical Address)
   } Via Address Bus (BA)

10. RAM

11. Hexadecimal Decoder

12. Trap Status (Z-word)

13. Register File (left) Output, or
    ALU Output (AU)

14. Bootload PROM.

Internal Bus Destinations

1. Program Counter (P)
2. Memory Address Register (Y)
   } → Group 1

3. Indicator Register (I) → Group 2

4. RAM → Group 3

5. LINK Register (XL)
6. Instruction Register (F/SEL)
7. Control Panel Display
8. H Register
9. Status/Security Register (S)
10. Megabus Data Lines
11. ALU Data Source Selector
12. Miscellaneous Control Elements
    (Control Flip-Flops, MMU, Test
    Logic, etc.)

   → Group 4

Firmware can select one element (or combinations of several elements) as an internal bus source, and deliver these data to one destination in each of the four groups listed above. The combinations of internal bus elements that are available as internal bus sources are:

1. Two copies of ALU output (bits 0C through 0F) and three interleaving Zeros

2. Eight copies of H register (bit 18) and ALU output (bits 18 through 1F)

3. H register (bits 1C through 1F) and contents of Megabus data buffer.

4. H register (bits 1C through 1F) and contents of Megabus procedure buffer

5. H register (bits 1C through 1F) and control word from control panel

6. H register (bits 10 through 17) right justified and sign extended to 16 bits.

4.7 CPU REGISTERS

The CPU registers, except those contained within the RALU, are described in the following subsections.

4.7.1 Indicator Register (I)

The CPU indicators can be loaded using the firmware controls described in Table 3-11.

4.7.1.1 Arithmetic Indicators

Two indicators can be loaded with the results of arithmetic operations in the CPU; the Overflow (OV) indicator and the Carry (C) indicator. The available inputs to these indicators are:

1. OV Indicator

   - OVFL (ALU overflow signal)
   - Result of exclusive OR operation between internal bus bits 10 and 11

2. C Indicator

   - CRY (ALU carry signal)
   - Internal bus bit 10
   - Internal bus bit 1F
   - Q register bit 1F just prior to right shift in the current firmware step.

4.7.1.2 Bit Test Indicator

Inputs available to the bit test indicator are:

- AUZERO (ALU zero detect signal)
- Internal bus bit 10.

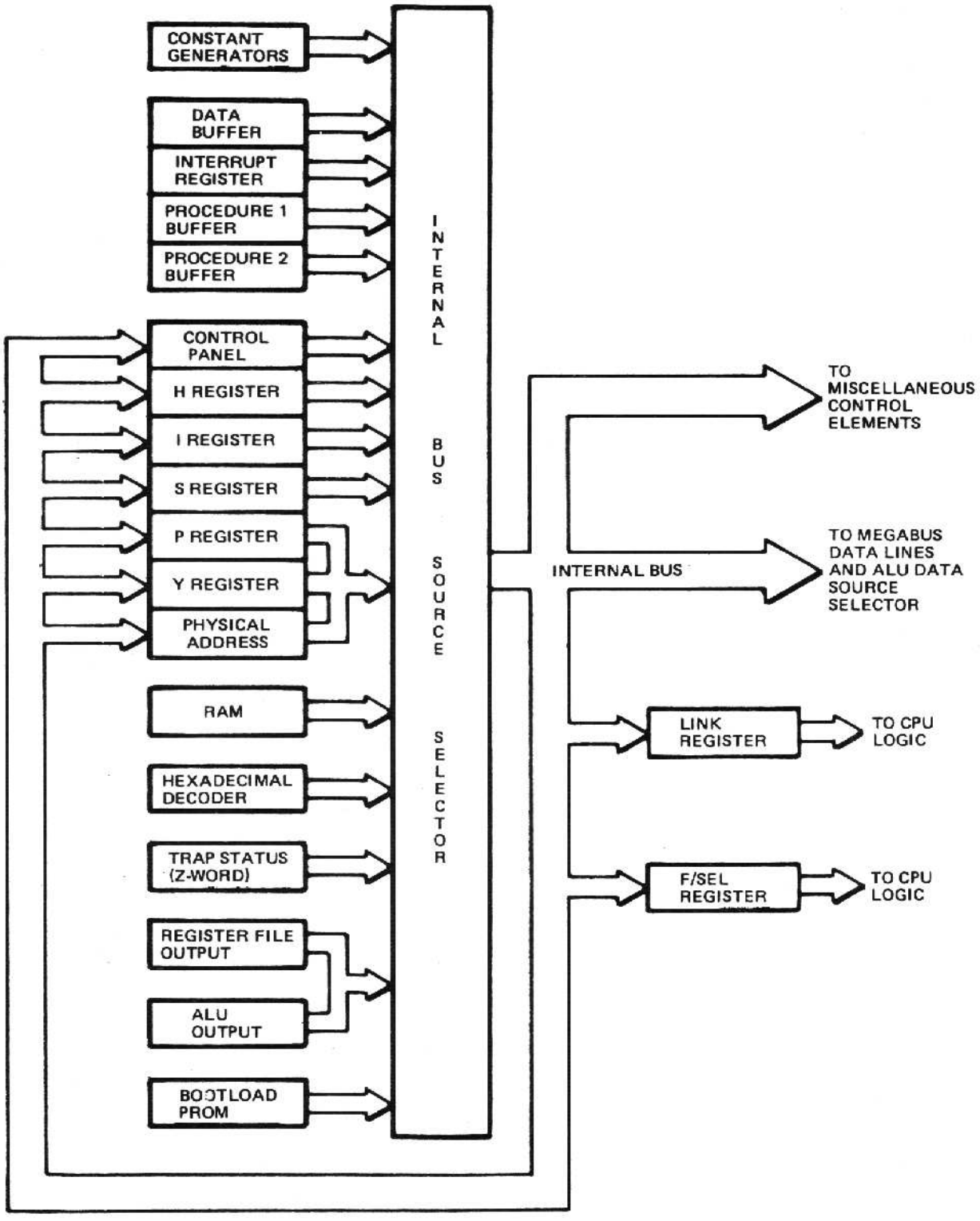


Figure 4-5 Internal Bus Sources and Destinations

4.7.1.3 Input/Output Indicator

This indicator stores the results of the last I/O instruction performed by the CPU. This is accomplished by making the Megabus acknowledge signal available at the input to this indicator. If the I/O instruction is accepted, the indicator is set; otherwise, it is cleared.

4.7.1.4 Comparison Indicators

Three indicators store the results of the last compare operation performed in the CPU; the Greater Than (G), Less Than (L), and Unlike Signs (U) indicators. The inputs available to these indicators are:

1. G Indicator
   - Internal bus bit 10 is zero and the ALU output (16 bits) is not zero
   - Complement of the SIGN flip-flop
2. L Indicator
   - Internal bus bit 10
   - ALU result bit 0C
   - SIGN flip-flop
3. U Indicator
   - Internal bus bit 10.

4.7.2 LINK Register (XL)

Refer to subsection 4.3.3 for a description of the XL register.

4.7.3 Counter Type Registers

Counter type registers are versatile in that they are alternately used as a 4-bit counter or as a 4-bit storage register within their respective circuit configurations.

4.7.3.1 Counter Register (CTR)

The CTR register (RCTR0F through RCTR3F) counts the number of procedure words fetched during instruction execution, and in conjunction with the program counter allows the Trap Handler software to reconstruct the instruction word address in the event an exception condition is detected in the CPU. To accomplish this, firmware initializes the register to a count of 1.

4.7.3.2 Select Register (SEL)

The SEL register (RSEL0F through RSEL3F) generally holds the least significant hexadecimal digit of the current instruction. At times, it is used to count repetitive actions in Shift, Multiply, or Divide operations. Data are received over internal bus bits 1C through 1F. When the SEL register is used as a counter, it is decremented and its contents tested for zero.

4.7.3.3 Byte Indexing Register (XB)

The XB register (RXB0FF through RXB3FF) stores bits shifted out of RALU data registers during half-word, digit, or bit indexing operations. At the start of each instruction, it is cleared to zero. Its output is sent directly to a hexadecimal decoder, and is also available to the internal bus for trap reporting.

4.7.4 Instruction Register (F)

The F register (RF00FF through RF08FF) accepts and stores the most significant three hexadecimal digits of instructions from memory for execution in the CPU.

4.7.5 H Register

The H register (RH10FF through RH1FFF) is configured into two 8-bit segments that accept data directly from the internal bus. Its output is available to the internal bus source selector for delivery to the internal bus, but as its contents are deposited onto the bus, the least and most significant eight bits of the register are swapped.

4.7.6 Status/Security Register (S)

The S register (RS01FF, RS02FF, and RS10FF through RS15FF) retains the system status and security codes for use within the CPU. It is also used for comparing the priority level of an incoming interrupt request with the current CPU operating priority level (refer to subsection 4.11.2). This compare activity is performed to deny acceptance of an incoming request when the request level is equal to or lower than the current CPU priority level. The contents of this register are controlled entirely by firmware, except for the CPU channel number (bits 8 and 9) which are switch controlled.

4.7.7 P Register (Program Counter)

The P register (RP03CF through RP22CF) consists of five 4-bit counters capable of being preset that retain and increment the current instruction address originally obtained from the internal bus.

4.7.8 Memory Address Register (Y)

The Y register (RY03CF through RY22CF) provides operand addresses (via the address bus) for memory or a peripheral device. Its architecture and functionality are basically the same as the P register (refer to subsection 4.7.7). Also included is a logical switching network which, under firmware control, generates a 16-bit address field by isolating the most significant byte (bits 3 through 6) from the 20-bit address. The 16- and 20-bit address fields are defined in Section I as the Short Address Form (SAF) and the Long Address Form (LAF), respectively.

4.7.9 Megabus Registers

The Megabus registers consist of three temporary storage buffers that receive procedure and data words (over the Megabus) from memory. Refer to subsection 4.10.6 for a description of these buffers.

4.7.10 Interrupt Register

Refer to subsection 4.11.3 for a description of the interrupt register.

4.7.11 M Register

The M register accepts pertinent RAM data (see Table 4-3) as the RAM is being updated, and since this data is not immediately available from the RAM, the M register delivers it to the test logic and next address generation logic for instantaneous action. This data is obtained from several sources and loaded into the M register as directed by the GP Field of the current firmware word. These sources include:

- Y register bit 15
- F register bit 0A
- SEL register bits 0 and 2
- H register bits 1A and 1B
- ZERO flip-flop
- AUZERO (ALU Zero Detect signal).

Table 4-3 M Register Format

BIT	SIGNAL	DESCRIPTION
0	RMTRAC	Trace Trap Enable
1	RMSCI1	S1 memory operand length is double-word
2	RMSCI2	S2 memory operand length is double-word
3	RMSCI3	S3 memory operand length is double-word
4	RMSQB6	B6 is in Stack or Queue mode
5	RMSQB7	B7 is in Stack or Queue mode
6	GOTSPU	SIP is part of the processor complex
7	GOTBPU	CIP is part of the processor complex

4.8 CPU CONTROL FLIP-FLOPS

The control flip-flops receive inputs from sources both internal and external to the CPU, permitting modification of firmware actions based on the results of operations performed through the system.

4.8.1 SIGN Flip-Flop

The SIGN flip-flop provides temporary storage of control information during instruction execution, and may be set from:

- Internal bus bit 0
- Internal bus bit 4
- Internal bus bit 19
- One
- Zero.

4.8.2 MISC Flip-Flop

The MISC flip-flop provides temporary storage of control information during instruction execution, and is one of the flip-flops used to select the SHIN function for RALU shift operations (refer to subsection 4.4.6). The MISC flip-flop may be set from:

- Complement of internal bus bit 19
- Internal bus bits 4 through 9 equal to Zero
- CRY - ALU Carry signal
- ACK - Megabus Acknowledge signal
- PROV - MMU Protection Violation signal
- Zero
- One.

4.8.3 SHIN1 Flip-Flop

The SHIN1 flip-flop is primarily used to select the SHIN function for RALU shift operations (refer to subsection 4.4.6), and may be set from:

- I(B) - I register Bit Test indicator
- Zero
- One.

4.8.4 SHIN2 Flip-Flop

The SHIN2 flip-flop is primarily used to select the SHIN function for RALU shift operations (refer to subsection 4.4.6), and may be set from:

- Complement of SIGN flip-flop
- Zero
- One.

4.8.5 ZERO Flip-Flop

The Zero flip-flop is primarily used for temporary storage of an ALU Zero Result condition, and may be set from:

- ALU Zero Detect signal
- QLT active flip-flop from control panel - equals One only if the last CPU Quality Logic Test (QLT) failed
- Zero
- One.

4.8.6 WRAP Flip-Flop

The WRAP flip-flop facilitates the checking of address-arithmetic firmware to detect attempts to exceed the 20-bit capacity of the address registers. If WRAP is On, any access to the Megabus (read request or write operation) will result in the transmission of an illegal address or I/O channel number. This action results in no response via the Megabus, which is interpreted as an "unavailable resource." The WRAP flip-flop may be set from the inequality of the ALU Carry signal (CRY) and the SIGN flip-flop.

4.8.7 NEWXR Flip-Flop

The NEWXR flip-flop distinguishes between reentrant invocations of the XR "splatter" branch. This flop is set when the SEL register is loaded from the internal bus (e.g., during instruction fetch operations); it is cleared when a branch is performed to XR, XE, XW, or XF, and when the WRAP flip-flop is set or cleared.

4.8.8 ACK Flip-Flop

The ACK flip-flop remembers whether the most recent Megabus action was accepted or rejected: if ACK is On, the action was accepted; if ACK is Off, the action was rejected.

4.8.9 YELLOW and PARER Flip-Flops

The YELLOW and PARER flip-flops signal detection of an error in memory or on the Megabus. YELLOW remembers whether at least one data error was corrected by the memory Error Detection And Correction (EDAC) hardware since the last interrogation of this flip-flop; YELLOW is cleared each time it is interrogated by the firmware. PARER remembers whether the most recent Megabus buffer reference (BD or BP) reported either a Megabus parity error or a data error not correctable by memory EDAC hardware. Unless the control panel is in Load, Read, or Write mode, the setting of PARER forces the firmware to control store location 00[illegible] for suitable trap generation.

4.8.10 EXTRAP, INTBSY, and TICK Flip-Flops

The EXTRAP, INTBSY, and TICK service request flip-flops are set by hardware to signal a requirement for a break in firmware flow.

EXTRAP is true when one or more external processors (CIP or SIP) has detected a trap condition. EXTRAP becomes false when all external processors with trap conditions have delivered their trap words.

INTBSY is set when an external interrupt of high enough priority is received and accepted by the hardware. No further interrupt, regardless of priority, can be accepted until firmware services buffer RUP, reloads the level field in the S register, and clears INTBSY.

TICK is set every 8-1/3 milliseconds by a crystal-controlled oscillator, signaling the need for service of the YELLOW logic, the RTC and/or WDT, and the control panel.

4.8.11 LOAD, TRAFFIC, and PANOK Flip-Flops

These flip-flops communicate control information between frimware and the operator.

The LOAD flip-flop can be set and cleared both by the operator and by firmware. During system startup operations, LOAD is normally set by the operator and, when bootload action is completed, cleared by firmware. Thereafter, this flip-flop usually remains OFF, but is sometimes set briefly by firmware as a means of preventing a trap to location 000 when a Megabus cycle is addressed to a possibly unavailable resource.

The TRAFFIC flip-flop is loaded by firmware to control the corresponding indicator on the control panel, but is held OFF by hardware unless the control panel is in Run mode. The TRAFFIC flip-flop may be set from the complement of the ZERO flip-flop, which indicates whether or not the instruction op-code just fetched from memory is an HLT (Halt).

The PANOK flip-flop synchronizes the servicing of operator requests. It is set to Zero whenever the CLEAR or EXECUTE push-button is depressed and when, in register-change mode, a hexadecimal key is depressed. This flip-flop is set to One by the firmware that services the request, and is used to prevent multiple servicing of a single key-stroke.

4.8.12 EFFRING, NONPROC, NOCHECK, SEGERR, and PROV Controls

These signals and flip-flops support normal MMU operations, permit temporary alteration of access rules, and report errors detected by the MMU.

EFFRING is a two-bit register containing the effective ring number, which the MMU uses to determine the degree of privilege appropriate to the current instruction, and against which memory access requests are tested. Firmware loads EFFRING from the S register RING field at the start of each instruction. EFFRING is modified to decrease its privilege level whenever, in the course of formulating an address, it uses data that might have been generated by a less privileged program.

NONPROC establishes a temporary change in the rules of access. Memory references which use the P register as the address source normally require "Execute" permission; when NONPROC has been set, they require only "Read" permission.

NOCHEK establishes a temporary suspension of the rules of access (it does not affect the mapping of segmented virtual addresses to physical, nor the detection of illegal, non-existent addresses). The intent of this functionality is to remove restrictions on memory access by system firmware (interrupt and trap handlers, RTC/WDT service, panel routines, etc.).

SEGERR signals that the MMU has detected an error in a virtual address; the referenced segment is not valid, or its size has been exceeded, or a protection violation has been detected. If SEGERR occurs during a memory reference, it causes the transmission of an illegal physical address. This action results in no response via the Megabus, which is interpreted as an unavailable resource. If no memory reference or access-rights test is requested, signal SEGERR is ignored.

PROV signals that the MMU has detected a protection violation (failure of access-rights check) on an otherwise legal address (i.e., an address in a valid segment and within the segment size). If a protection violation occurs during a memory reference, the PROV flip-flop is locked in the set state until cleared by the firmware function NOCHEK (this function is normally issued by the trap-generation firmware). If a protection violation occurs during a firmware step that explicitly requests an access-rights test, the next firmware step may copy PROV to the MISC control flip-flop.

4.9 MISCELLANEOUS CPU HARDWARE

The following subsections describe those CPU elements not previously defined under one of the major CPU logic areas.

4.9.1 Random Access Memory (RAM)

The RAM consists of sixteen 20-bit auxiliary storage registers that the CPU uses as work areas, and to maintain selected system status conditions. Addressing the RAM for a read or write operation is performed by the Left Select portion of the RALU addressing logic (refer to subsection 4.5). However, the actual RAM read or write operation is performed as directed by the DI Field of the current firmware word.

The organization of the RAM is illustrated in Figure 4-6.

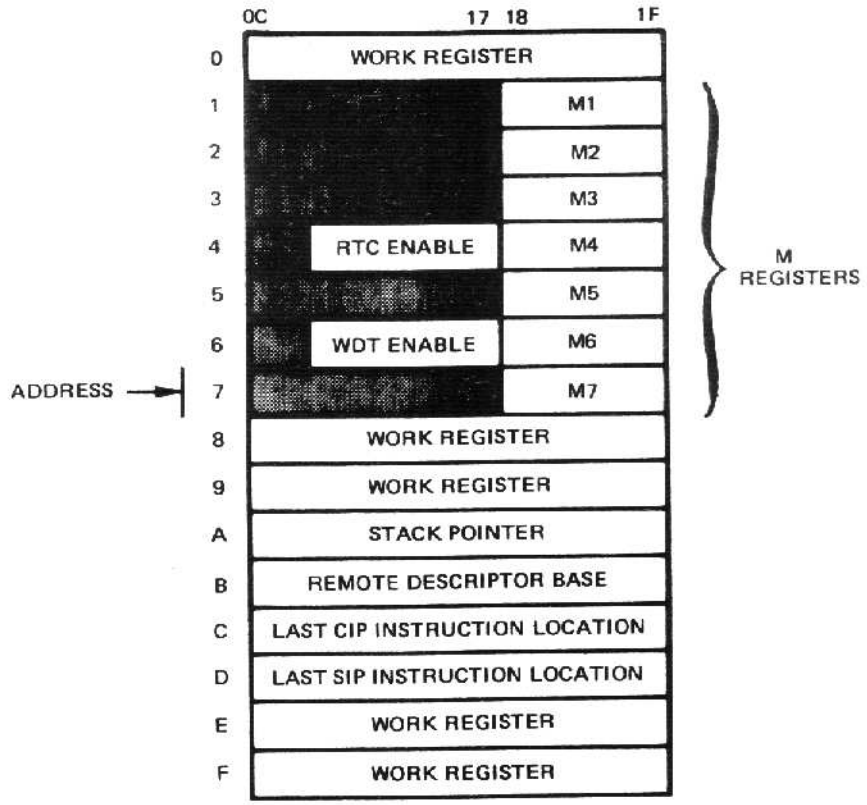


Figure 4-6 RAM Format

4.9.1.1 RAM Location 0

This RAM location contains the instruction word to be reported when a trap occurs.

CONFIDENTIAL & PROPRIETARY

4.9.1.2 RAM Locations 1 through 3

Bits 0C through 17 of these RAM locations are unused; bits 18 through 1F contain software mode registers M1 through M3.

4.9.1.3 RAM Location 4

Bits 10 through 17 of this RAM location contain the mode information for enabling the Real Time Clock (RTC); bits 18 through 1F contain software mode register M4.

4.9.1.4 RAM Location 5

Bits 0C through 17 of this RAM location are unused; bits 18 through 1F contain software mode register M5.

4.9.1.5 RAM Location 6

Bits 10 through 17 of this RAM location contain the mode information for enabling the Watch Dog Timer (WDT); bits 18 through 1F contain software mode register M6.

4.9.1.6 RAM Location 7

Bits 0C through 17 of this RAM location are unused; bits 18 through 1F contain software mode register M7.

4.9.1.7 RAM Location 8

This RAM location contains a pointer to the next word of procedure.

4.9.1.8 RAM Location 9

This RAM location is unused.

4.9.1.9 RAM Location A

This RAM location contains a stack pointer.

4.9.1.10 RAM Location B

This RAM location contains the Remote Descriptor Base (RDB) register.

4.9.1.11 RAM Location C

This RAM location contains a pointer to the most recently accepted Commercial Instruction Processor (CIP) instruction.

4.9.1.12 RAM Location D

This RAM location contains a pointer to the most recently accepted Scientific Instruction Processor (SIP) instruction.

4.9.1.13 RAM Location E

This RAM location is unused.

4.9.1.14 RAM Location F

This RAM location contains a pointer to the next word of procedure.

4.9.2 Bootload PROM

The bootload PROM, which consists of four 2K PROMs, is a standard feature of the CPU. Addressing for the boot PROM is provided by bits 14 through 22 of the address bus. The 20-bit output from the boot PROM is delivered to the internal bus either as directed by the BS Field of the current firmware word or when the Load pushbutton on the control panel is depressed.

The bootload operation has the following modes of operation, depending on the contents of the program counter (P):

P CONTENTS	OPERATION
0000	Executes Basic Logic Test (BLT). Reads one physical record from channel 010 (hexadecimal) into memory, starting at location 0100 (hexadecimal); branches to 0100 (hexadecimal). The preceding is the default Bootload procedure.*
0002	Does not execute BLT. Reads one physical record into memory, starting at location 0100 (hexadecimal), using the channel number previously entered into register D1; branches to 0100 (hexadecimal).
0004	Does not execute BLT. Reads one physical record into memory, starting at the address entered into register B1, using the channel number entered into register D1; branches to the address entered into register B1.

*This procedure is used when the control panel is locked. If the control panel is unlocked, procedure will halt after the BLT unless Run (R) pushbutton is depressed.

The devices supported and the Boot record file formats are as follows:

DEVICES	RECORD FORMAT
Diskette	Data portion (128 bytes) of track 0, sector 0 (first sector)
Card	The contents of the first card; 80 bytes punches in Bootload format
Paper Tape (ASR)	One record of 256 bytes or less, starting with the character that follows the first non-NUL character and continuing to the first X-OFF or T-OFF character with no escaped data
Cartridge	Data portion (256 bytes) of track 0, sector 0
Magnetic Tape	One record of 256 bytes or less; the record must be the first after BOT

4.9.3 Address Bus

The address bus (MYAD03 through MYAD22) accepts addresses from the P and Y registers (refer to subsections 4.7.7 and 4.7.8, respectively) as directed by the BS Field of the current firmware word. The 20-bit output from the address bus is delivered to the Megabus for I/O and memory read or write operations, or to the internal bus for distribution in the CPU.

4.9.4 Memory Management Unit (MMU)

The MMU checks all memory addresses before permitting them to take part in a memory reference (either over the Megabus to main memory or to the cache memory). These checks ensure that addresses are legitimate and do not violate any software imposed restrictions. If a memory address is rejected by the MMU, a protection violation results. Addresses before being processed by the MMU are called Virtaul addresses, while addresses after being processed by the MMU are called Physical addresses.

4.9.5 Cache Memory

The cache memory contains copies of selected (recently referenced) main memory locations. It has a Megabus interface which allows it to make main memory read references on behalf of the CPU and to monitor the Megabus, copying main memory write data if it currently contains a copy of the main memory location addressed. The cache also has a private interface allowing it to communicate to the CPU to which it is dedicated. It receives main memory read requests across this interface, thereby becoming committed to locate the data for the CPU in its local cache array or in the actual main memory. In either case, the requested data is returned to the CPU with an appropriate handshake.

4.9.6 Hexadecimal Decoder

The hexadecimal decoder (RXBD10 through RXBD1F) generates 16-bit masks for bit test operations (i.e., a single bit within a 16-bit word can be selected). The bit being tested is selected by the 4-bit value from the XB register (refer to subsection 4.7.3.3). This value is used to define which of the 16 bits is zero; the other 15 bits are made all ones.

4.9.7 Subcommand Generator

The 6-bit GP field of the Control Store Word (CSWD) is essentially viewed by firmware as three independent hexadecimal subfields: 10, 20, and 30, which control various hardware operations. Actually, these subfields provide a 6-bit hexadecimal address (0 through 3F) to two PROM devices and a binary-to-decimal decoder. These devices comprise the subcommand generator (see Figure 4-7) which provides control signals (subcommands) for the CPU hardware.

Decoder selection occurs when CSWD 36 through 38 equal 6 (110) and MCLOCK+ is inactive. This disables the two PROMs and enables the least significant three bits of the low-order address field (CSWD 39 through CSWD 41) into decoder 30 for selection of the applicable control signal. Also, when CSWD 36 and 37 equal 3 (11), they enable the control panel for service by the CPU.

Selection between the two PROMs is performed by CSWD 36 and 37 as indicated below:

CSWD		PROM
36	37	
0	0	10/20
0	1	10
1	0	20

This enables the respective hexadecimal address field into the selected PROM for selection of the applicable control signal(s).

NOTE

Some overlap between the two PROMs can occur to provide adequate control signal selection.

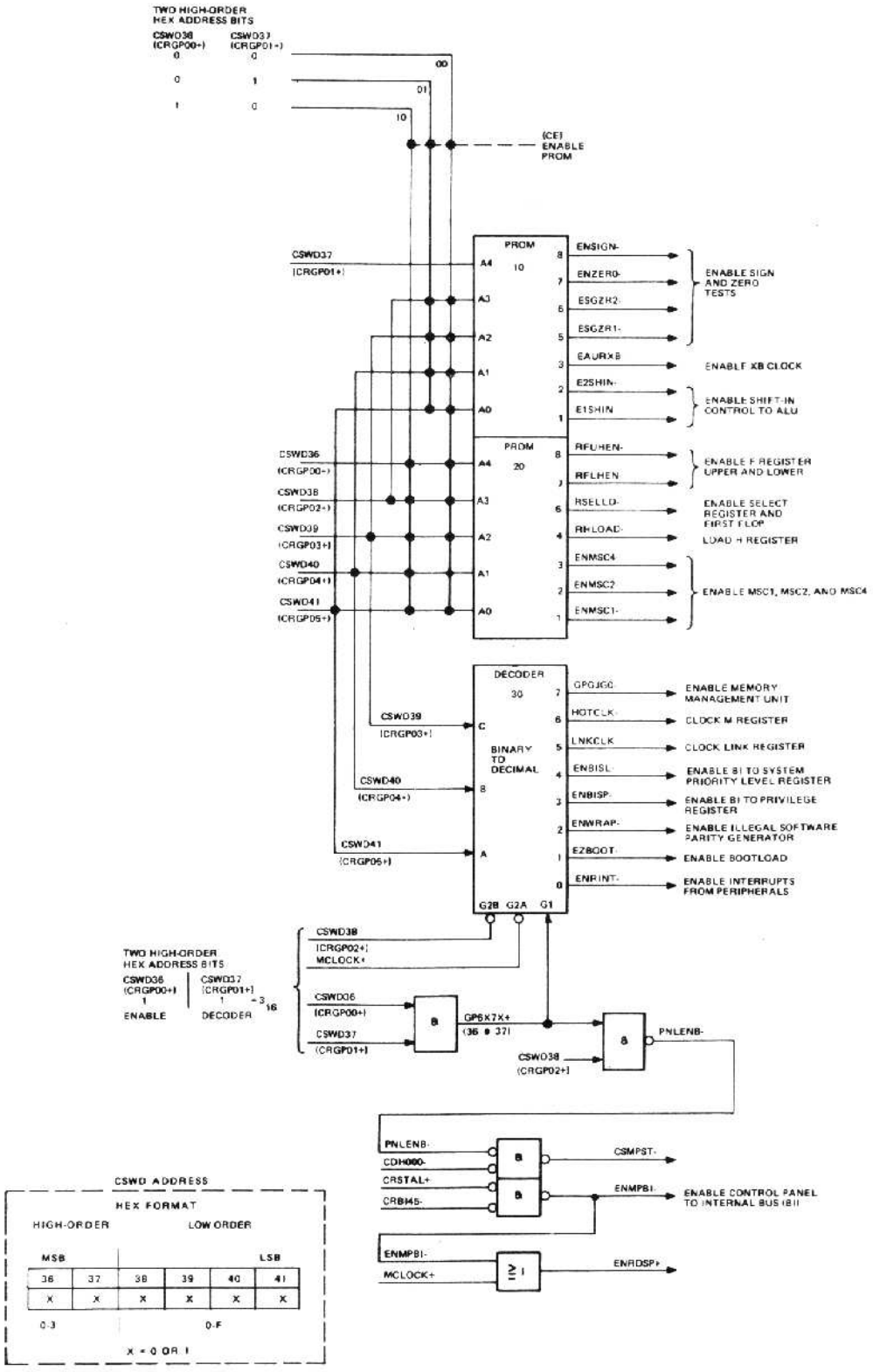


Figure 4-7 Subcommand Generator Logic

4.9.8 Constant Generators

The CPU constant generators provide two types of constants, which are available as internal bus sources as directed by the BI Field of the current firmware word. These constants are:

1. Numeric constants, and

2. Constants used to generate control words for communication with external processors.

4.9.9 Control Panel

Refer to subsection 4.12 for a description of the full control panel.

4.10 MEGABUS NETWORK

The Megabus provides a common communication path among available system units. The design of the Megabus is asynchronous to make communications possible between units of varying speeds.

4.10.1 Interface Logic

The Megabus interfaces with the CPU via a group of transceivers that provide the equivalent electrical characteristics required of all bus connections to allow data, address, and control signals to be routed to and from the CPU. Table 4-4 provides a complete list of the interface signals, while subsections 4.10.1.1 through 4.10.1.6 provide a brief description of each Megabus/CPU interface signal.

4.10.1.1 Timing Signal Lines

The following signals provide the handshake function required by a unit to either initiate, accept, or deny a request for a Megabus cycle from another unit.

Bus Request (BSREQT-)

When true, this signal indicates that one or more of the units connected to the Megabus have requested a bus cycle. When the signal is false, no requests are pending.

Bus Data Cycle Now (BSDCNN-)

When true, this signal indicates that a specific master unit has been granted a requested Megabus cycle and has placed information on the bus for use by a specific slave unit. When this signal is false, the bus is not busy (i.e., between bus cycles).

Table 4-4 Megabus Interface Signals (Sheet 1 of 2)

TYPE	FUNCTION	CPU SIGNAL			
		DRIVER ENABLE	DRIVER INPUT	RECEIVER OUTPUT	BUS SIGNAL
Bus Handshake	Bus Request	GROUND	MYREQT+	BSREQT+	BSREQT-
	Data Cycle Now	GROUND	MYDCNN+	BSDCNN+	BSDCNN-
	Acknowledge	GROUND	MYACKR+	BSACKR+	BSACKR-
	Negative Acknowledge	GROUND	MYNAKR+	BSNAKR+	BSNAKR-
	Wait	GROUND	GROUND	BSWAIT+	BSWAIT-
Data	Data Bit A (unused)	-	-	-	BSDT0A-
	Data Bit 0	MYDCNW-	BIXX10+	BSDT00+	BSDT00-
	Data Bit 1	MYDCNW-	BIXX11+	BSDT01+	BSDT01-
	Data Bit 2	MYDCNW-	BIXX12+	BSDT02+	BSDT02-
	Data Bit 3	MYDCNW-	BIXX13+	BSDT03+	BSDT03-
	Data Bit 4	MYDCNW-	BIXX14+	BSDT04+	BSDT04-
	Data Bit 5	MYDCNW-	BIXX15+	BSDT05+	BSDT05-
	Data Bit 6	MYDCNW-	BIXX16+	BSDT06+	BSDT06-
	Data Bit 7	MYDCNN-	MYDT17+	BSDT07+	BSDT07-
	Data Bit B (unused)	-	-	-	BSDT0B-
	Data Bit 8	MYDCNN-	MYDT18+	BSDT08+	BSDT08-
	Data Bit 9	MYDCNN-	MYDT19+	BSDT09+	BSDT09-
	Data Bit 10	MYDCNN-	MYDT1A+	BSDT10+	BSDT10-
	Data Bit 11	MYDCNW-	BIXX1B+	BSDT11+	BSDT11-
	Data Bit 12	MYDCNW-	BIXX1C+	BSDT12+	BSDT12-
	Data Bit 13	MYDCNW-	BIXX1D+	BSDT13+	BSDT13-
	Data Bit 14	MYDCNW-	BIXX1E+	BSDT14+	BSDT14-
	Data Bit 15	MYDCNW-	BIXX1F+	BSDT15+	BSDT15-
Main Memory Address	Address Bit 0	MYDCNN-	GJAD00+	BSAD00+	BSAD00-
	Address Bit 1	MYDCNN-	ZGNDAA	BSAD01+	BSAD01-
	Address Bit 2	MYDCNN-	ZGNDAA	BSAD02+	BSAD02-
	Address Bit 3	MYDCNN-	GJAD03+	BSAD03+	BSAD03-
	Address Bit 4	MYDCNN-	GJAD04+	BSAD04+	BSAD04-
	Address Bit 5	MYDCNN-	GJAD05+	BSAD05+	BSAD05-
	Address Bit 6	MYDCNN-	GJAD06+	BSAD06+	BSAD06-
	Address Bit 7	MYDCNN-	GJAD07+	BSAD07+	BSAD07-
	Address Bit 8	MYDCNN-	GJAD08+	BSAD08+	BSAD08-
	Address Bit 9	MYDCNN-	GJAD09+	BSAD09+	BSAD09-
	Address Bit 10	MYDCNN-	GJAC10+	BSAD10+	BSAD10-
	Address Bit 11	MYDCNN-	GJAD11+	BSAD11+	BSAD11-
	Address Bit 12	MYDCNN-	GJAD12+	BSAD12+	BSAD12-
	Address Bit 13	MYDCNN-	GJAD13+	BSAD13+	BSAD13-
	Address Bit 14	MYDCNN-	GJAD14+	BSAD14+	BSAD14-
	Address Bit 15	MYDCNN-	MYAD15+	BSAD15+	BSAD15-
	Address Bit 16	MYDCNN-	MYAD16+	BSAD16+	BSAD16-
	Address Bit 17	MYDCNN-	MYAD17+	BSAD17+	BSAD17-
	Address Bit 18	MYDCNN-	MYAD18+	BSAD18+	BSAD18-
	Address Bit 19	MYDCNN-	MYAD19+	BSAD19+	BSAD19-
	Address Bit 20	MYDCNN-	MYAD20+	BSAD20+	BSAD20-
	Address Bit 21	MYDCNN-	MYAD21+	BSAD21+	BSAD21-
	Address Bit 22	MYDCNN-	MYAD22+	BSAD22+	BSAD22-
	Address Bit 23	MYDCNN-	RXBOFF+	BSAD23+	BSAD23
Transfer	Memory Reference	MYDCNN-	CIMREF+	BSMREF+	BSMREF-
	Byte	MYDCNN-	CRCLK0+	BSBYTE+	BSBYTE-
	Bus Write	MYDCNN-	CIREAD	BSWRIT+	BSWRIT-
	Second Half Read	MYDCNN-	MYSHBC+	BSSHBC+	BSSHBC-
	Lock	MYDCNN-	MYRMWC+	BSLOCK+	BSLOCK-
	Memory Error (Red)	MYDCNN-	GROUND	BSREDD+	BSREDD-
	Memory Error (Yellow)	MYDCNN-	GROUND	BSYELO+	BSYELO-
	Data Parity Left	MYDCNN-	MYDP00+	BSDP00+	BSDP00-
	Data Parity Right	MYDCNN-	MYDP08+	BSDP08+	BSDP08-
	Address Parity (bits 0-7)	MYDCNN-	MYAP00+	BSAP00+	BSAP00-
	Double Pull	MYDCNN-	XTDBPL+	BSDBPL+	BSDBPL-

Table 4-4 Megabus Interface Signals (Sheet 2 of 2)

TYPE	FUNCTION	CPU SIGNAL			
		DRIVER ENABLE	DRIVER INPUT	RECEIVER OUTPUT	BUS SIGNAL
Verify Bus Continuity	Logic Test Out	-	-	-	BSQLTO-
	Logic Test In	-	-	-	BSQLTI-
	Logic Test Active	-	-	-	BSQLTA+
Establish Positional Priority	Tie-Breaking Network	-	-	-	BSAUOK+
	Tie-Breaking Network	-	-	-	BSBUOK+
	Tie-Breaking Network	-	-	-	BSCUOK+
	Tie-Breaking Network	-	-	-	BSDUOK+
	Tie-Breaking Network	-	-	-	BSEUOK+
	Tie-Breaking Network	-	-	-	BSFUOK+
	Tie-Breaking Network	-	-	-	BSGUOK+
	Tie-Breaking Network	-	-	-	BSHUOK+
	Tie-Breaking Network	-	-	-	BSIUOK+
	Tie-Breaking Network	-	-	-	BSMYOK+
Miscellaneous	Master Clear	GROUND	MYMCLR+	BSMCLR+	BSMCLR-
	Power On	-	-	-	BSPWON+
	Resume Interrupt	GROUND	ENRINT+	BSRINT+	BSRINT-
	Spare Line (unused)	-	-	-	BSSPR1-
	Spare Line (unused)	-	-	-	BSSPR3-
	Spare Line (unused)	-	-		BSSPR4-
	External Connection (unused)	-	-	-	BSEXTC+

Bus Acknowledge Response (BSACKR-)

When true, this signal indicates to the master unit that the slave unit has received and accepted a specific transfer from the master unit.

Bus Negative Acknowledge Response (BSNAKR-)

When true, this signal indicates to the master unit that a slave unit is refusing a specific transfer (i.e., the slave unit cannot accept the transfer and the master unit should not attempt a retry). For example, a busy memory addressed for a data transfer.

If a situation arises when no unit on the bus recognizes the transfer because of improper addressing or a malfunction, and no response (ACK, NAK, or WAIT) is generated within 5 microseconds, the CPU (which monitors all bus transfers) will issue a Bus Negative Acknowledge Response (BSNAKR-) signal on behalf of the entire system. This is referred to as the Dead Man Time-Out operation.

Bus Wait (BSWAIT-)

When true, this signal indicates to the master unit that the slave unit cannot accept a specific transfer at this time (i.e, the slave unit is temporarily busy and the master unit must initiate successive retries until the transfer is acknowledged).

4.10.1.2 Information Signal Lines

The signals described in this subsection effect the transfer of information during a bus cycle as data or information signals.

Bus Data Lines (BSDT00- through BSDT15-)

The bus data bits can be formatted for a single data word (16 bits), channel number coding (CPU), or the low-order address bits (8 through 23), depending on the operation being performed. Thus, data, address, control, register, or status information can be reflected by the 16 data lines.

Bus Address Lines (BSAD00- thorugh BSAD23-)

The 24 address lines can be formatted for a single 23-bit main memory address to select one of 8M bytes.

The address lines can also be formatted for a channel number code (CPU), and I/O function code on lines 18 through 23, or a combination of all three for an IOLD operation.

4.10.1.3 Information Control Signal Lines

The following signals serve as data, address, and information control signals that effect the transfer and control of such information during a bus cycle.

Bus Memory Reference (BSMREF-)

When true, this signal indicates that bus address lines 0 through 23 contain a complete main memory address from the master unit. When false, the Bus Memory Reference signal indicates that the bus address lines contain a channel number on lines 8 through 18 (with or without a function code on lines 18 through 23), or a main memory module address code on lines 0 through 7; the exact configuration and direction fo flow depends on the operation being performed.

Bus Byte (BSBYTE-)

When true, this signal indicates that the current transfer is a byte rather than a word transfer (used during memory write operations only).

Bus Write (BSWRIT-)

When true, this signal indicates that the master unit is transmitting data to the slave unit. When this signal is false, the initial bus cycle signals a read request ($\overline{\text{BSWRIT}}$), while the data lines contain the requesting channel number; the slave unit, if it accepts the request, replies with a read response via a subsequent bus cycle, which is defined as a Second Half Bus Cycle (BSSHBC).

Bus Second Half Bus Cycle (BSSHBC-)

When true, this signal indicates to the master unit that the current information generated by the slave unit is the information previously requested during the initiation cycle.

Bus Lock (BSLOCK-)

When true, this signal indicates that the master has requested a change in the status of the memory unit lock flop. The bus lock signal also enables a three-cycle, Read-Modify-Write operation, which allows the following three cycles to be executed for the requesting unit without interruption (i.e., other units attempting a Read-Modify-Write operation will not be acknowledged):

1. First Cycle (read): The address bus contains the memory address, and the data bus contains the channel number of the requestor.

2. Second Cycle (response): The address bus contains the channel number of the requestor, and the data bus contains data read from main memory.

3. Third Cycle (write): The address bus contains the memory address, and the data bus contains data to be written into memory.

Bus Double Pull (BSDBPL-)

When true, this signal indicates that the master unit is requesting a double-word operand from the slave unit. During the first Second Half Bus Cycle, BSDBPL- is redelivered to the requesting unit, indicating that another word follows.

NOTE

If a single fetch memory is installed on the system, BSDBPL- is not redelivered during the Second Half Bus Cycle, notifying the requesting unit that only single-word operations will be performed.

4.10.1.4 Status/Error Signal Lines

The following signal lines provide main memory error reporting signals for the available units, and two-way bus parity lines for odd parity signals used with the address and/or information bits placed on the Megabus. Two lines provide for a bus continuity check, combined with a check on the integrity of the resident logic test in each unit. A single line is used to indicate the status of system power.

Bus Red (BSREDD-)

The Bus Red error signal can only be generated by a main memory unit that contains EDAC logic. When true, the signal indicates that the memory has detected an error during a read (Second Half Bus Cycle) operation.

Bus Yellow (BSYELO-)

The Bus Yellow error signal can only be generated by a main memory unit that contains EDAC logic. When true, the signal indicates that memory has detected and corrected an error during a read (Second Half Bus Cycle) operation.

Bus Address Parity (BSAP00-)

The level of the Bus Address Parity signal provides odd parity for address bits 0 through 7 (module address bits).

Bus Data Parity - Left Byte (BSDP00-)

The level of the Bus Data Parity - Left Byte signal provides odd parity for the left data byte (bits 0 through 7).

Bus Data Parity - Right Byte (BSDP08-)

The level of the Bus Data Parity - Right Byte signal provides odd parity for the right data byte (bits 8 through 15).

Bus Quality Logic Test Out (BSQLTO-) and In (BSQLTI-)

The Bus Quality Logic Test Out and In signals are static integrity signals which, if continuously true, indicate that each available unit performed its resident Quality Logic Test (QLT) successfully. The signal is relayed from unit to unit from one end of the bus to the other and back. This signal effectively provides a continuity check for all available units.

Bus Quality Logic Test Active (BSQLTA+)

When true, this signal indicates that at least one unit on the bus has not successfully completed its logic test or that there is a break in continuity on the bus (i.e., a unit is not installed).

Bus Power On (BSPWON+)

When the Bus Power On signal is true, it indicates that all system power supplies are functioning correctly. This signal goes true when power has stabilized and goes false several milliseconds before power fails.

4.10.1.5 Tie-Breaking Control Signals

There are nine tie-breaking signals (BSAUOK+ through BSIUOK+), all of which must be true to provide an enable for any unit requesting a bus cycle. If more than one unit simultaneously requests a bus cycle, the cycle is granted to only one unit on a positional priority basis. The priority extremes are the ends of the bus; memory has the highest positional priority and the CPU has the lowest. Thus, under simultaneous request conditions, the highest positioned requesting unit receives true enables from all nine tie-breaking signals, while the remaining requesting units receive eight or less, depending on the relative position of their decreasing priority.

Bus My OK (BSMYOK+)

When true, this signal indicates to the next lower priority unit that the generating unit, and certain other units of a higher positional priority, have not requested a bus cycle within the last 20 nanoseconds; therefore, a cycle can be granted (if requested) to a lower priority unit.

4.10.1.6 Operational Control Signals

The following control signals are asynchronous in relation to the functions they perform and the normal initiation and control of bus cycles.

Bus Resume Interrupt (BSRINT-)

When true, this signal allows the CPU or memory to reissue an interrupt that has previously been refused.

Bus Master Clear (BSMCLR-)

When true, this signal indicates that the Master Clear (CLR) pushbutton, located on the CPU control panel, has been depressed.

4.10.2 Megabus Network Operation

The information contained in this subsection describes the bus cycle timing and controls in relation to the handshaking techniques used to establish communications between any two units in the system. Bus dialogue is completely asynchronous, and each bus cycle can be considered as an independent handshaking sequence between a master unit and a slave unit. To implement this type of operation, all major logic boards within the units that comprise the system have similar Megabus cycle control logic.

Figure 4-8 depicts the general timing relationships incurred when handshaking techniques are applied between a master unit and a slave unit on request of the master unit. The Bus Cycle Request (BSREQT) signal is common to all available units, and when true, indicates that one or more have initiated a request for a bus cycle.

If a bus cycle request is actually the result of simultaneous requests, the tie-breaking logic (contained in each unit) resolves the priority dilemma by granting the bus to a specific unit on the basis of position. At this point, it is enough to understand that memory resides at the high priority end of the bus and the CPU resides at the low priority end; the remaining units occupy intermediate priority positions.

When a bus cycle is granted, the Bus Data Cycle Now (BSDCNN) signal goes true, indicating that the tie-breaking operation is complete and a specific unit has been granted master status (gained bus control). At this time, the data and address information designated for the slave unit is placed on the bus. Assuming a multi-unit system, the Bus Data Cycle Now signal allows each potential slave unit to internally generate a Data Cycle Now Delayed (BSDCND) signal. The delay (60 nanoseconds) provides bus skew correction time for each unit, allowing the currently active bus address information to be properly examined by the decoding networks available in each unit. Under these conditions, the Data Cycle Now Delayed signal, in conjunction with signal MYCHAN that is derived from the appropriate unit's decoding of the predefined address, allows an internal strobe to be generated in the designated slave unit. The slave unit then returns one of three response signals (BSACKR, BSNAKR, or BSWAIT) to the master unit to complete the handshake and to indicate that the slave unit acknowledges the communication as being accepted, denied, or postponed, respectively.

A bus handshake is an asynchronous operation in which the change of a signal level can only occur in response to the level change of a handshake signal from a defined unit. Since various

units can differ in the length of time required between strobe development and acknowledgment, signal transitions depend on the type of internal functionality. In the case where no acknowledgment is received by a Master Unit (no response signal generated), the CPU Dead Man Time-Out operation allows a delay of 5-microseconds before clearing the bus on behalf of the entire system.

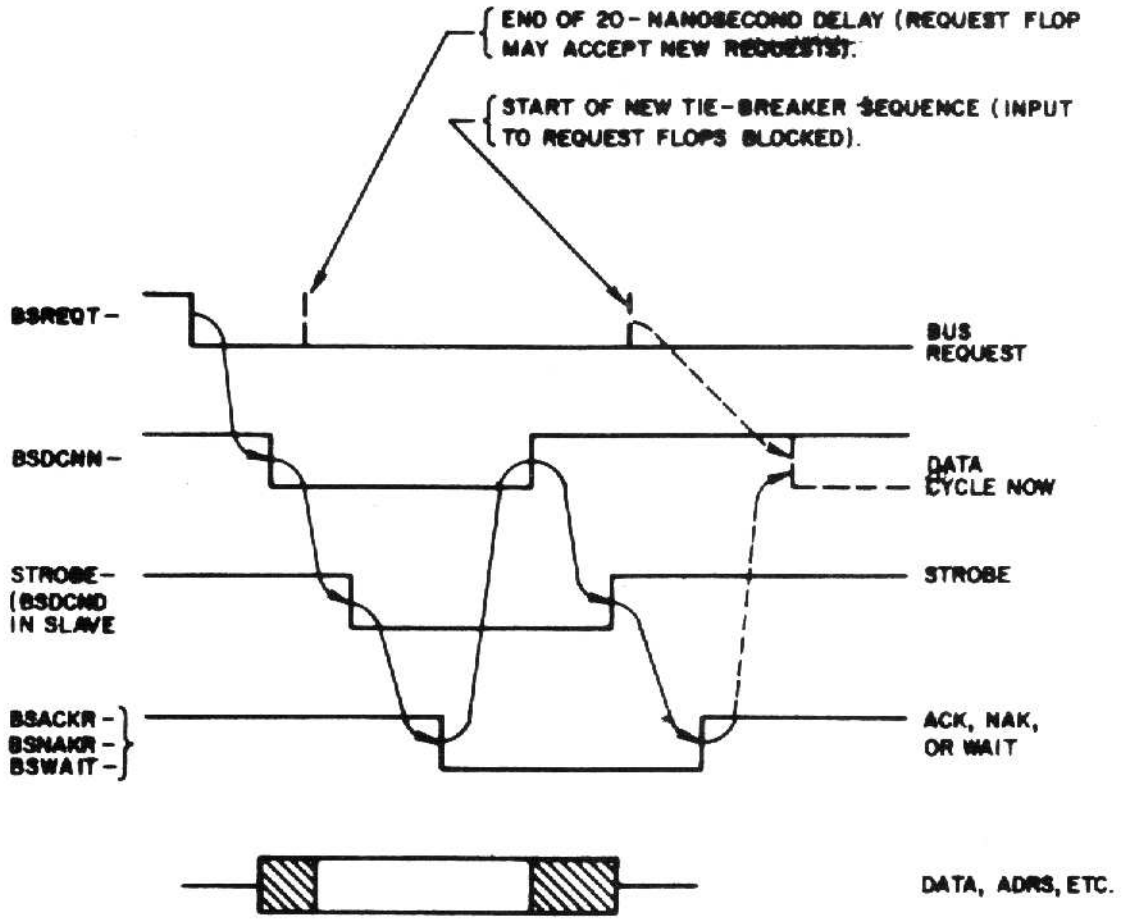


Figure 4-8 Bus Timing Diagram

4.10.3 Megabus Tie-Breaking Function

The tie-breaking function resolves simultaneous Megabus requests and grants bus cycles on a positional priority basis. In any given system, memory has the highest priority and the CPU has the lowest priority; they reside at opposite ends of the bus. Other units occupy intermediate positions and have a priority that increases relative to their proximity to the memory end of the bus.

Each unit on the Megabus contains the logic necessary for granting a Megabus cycle. When the Data Cycle Now signal is valid, the requesting unit gains access to the Megabus.

Referring to Figure 4-9, note that each unit's grant logic monitors the preceding unit's tie-breaking lines (BSAUOK through (BSIUOK). Therefore, only one unit has its grant logic satisfied at any one time. The one primary line that is necessary to satisfy the input structure of the grant logic is BSAUOK. In reality, BSAUOK is the preceding higher priority signal BSMYOK.

NOTE

Main memory always takes priority on Megabus accesses and does not need to monitor the remaining unit's tie-breaking lines.

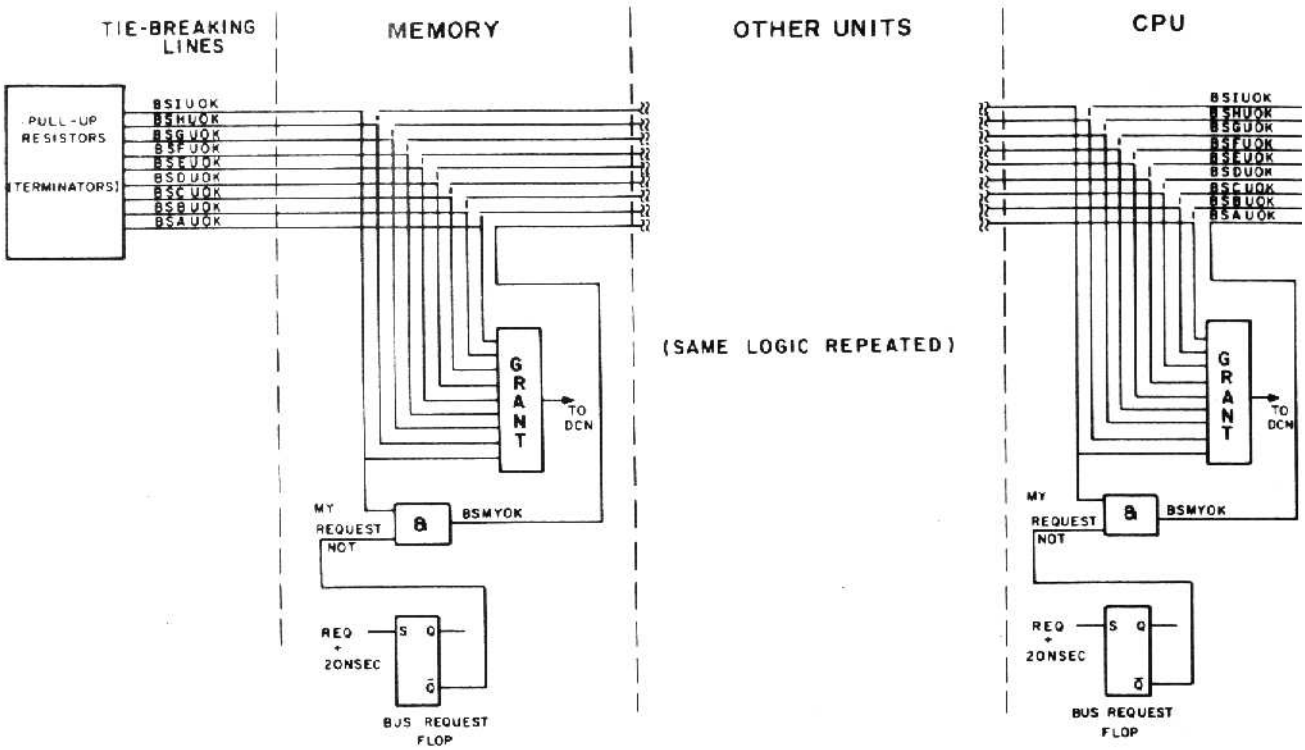


Figure 4-9 Megabus Tie-Breaking Logic

If the CPU requires access to the Megabus (as indicated by control store bit 31), the User flip-flop sets, initiating a CPU start request. When there is no Megabus activity, the Request flip-flop sets, producing a CPU bus request and disabling other units from gaining access to the Megabus during the processor response (see Figure 4-10).

With the Grant flip-flop set, the Data Cycle Now signal generates one of the following three responses: Acknowledge (ACK), Negative Acknowledge (NAK), or Wait (WAIT). When the response

signal is received, the Grant flip-flop clears and the Megabus is available for use by another unit.

4.10.4 Memory Read Request

The CPU can: (1) with a single memory read request, signal that it wants the delivery of two sequential words from memory, and (2) have both a double-word and a single-word request outstanding to two different memory modules simultaneously.

Double-word data is always stored in the P1 and P2 Procedure Buffers whereas single-word data is stored in the Data Buffer (BD). Since both single- and double-word requests may be present at the same time, the processor tags the request in the function code field at the time of the request. Single-word requests are tagged 00_{16}, while double-word requests are tagged 20_{16}. During the request, bus data lines 10 through 15 constitute the tag. During the Second Half Bus Cycle (BSSHBC), address bits 18 through 23 constitute the tag echoed by the memory.

4.10.4.1 Double-Word Fetch

A double-word fetch operation is executed only if: (1) the Procedure Buffers are both empty and (2) memory is not currently processing a double-word pull. Firmware then performs a Procedure Stall (setting the Procedure Buffer Stall flip-flop) until memory transfers either a procedure or the first word of a double-word fetch (see Figure 4-10).

During the double word fetch operation, two flip-flops (Word A Request and Word B Request) sample the Second Half Bus Cycle signal. These two flip-flops set when a memory request is present and test the Second Half Bus Cycle signal when the CPU acknowledges with the My Second Half Response signal. When a double word is received from memory, the Word A Request and Word B Request flip-flops are respectively cleared. During a single word fetch, only one Second Half Bus Cycle signal is received, clearing the Word B Request flip-flop. Two additional flip-flops, Take Word C and Take Word D, track the firmware's usage of the data received from memory. The Take Word C flip-flop clears when firmware uses the first word and the Take Word D flip-flop clears when firmware uses the second word from memory.

4.10.4.2 Single Word Fetch

Single word fetches require at least two firmware steps. The first step generates a Megabus Read Cycle causing the Data Buffer Stall flip-flop to be set when memory (or the I/O) accepts the request cycle. The second step attempts to input the data onto the internal bus. These two firmware steps need not be contiguous. If the Second Half Bus Cycle signal is delayed, the Data Buffer Stall flip-flop remains set until the Second Half Read Data signal is received.

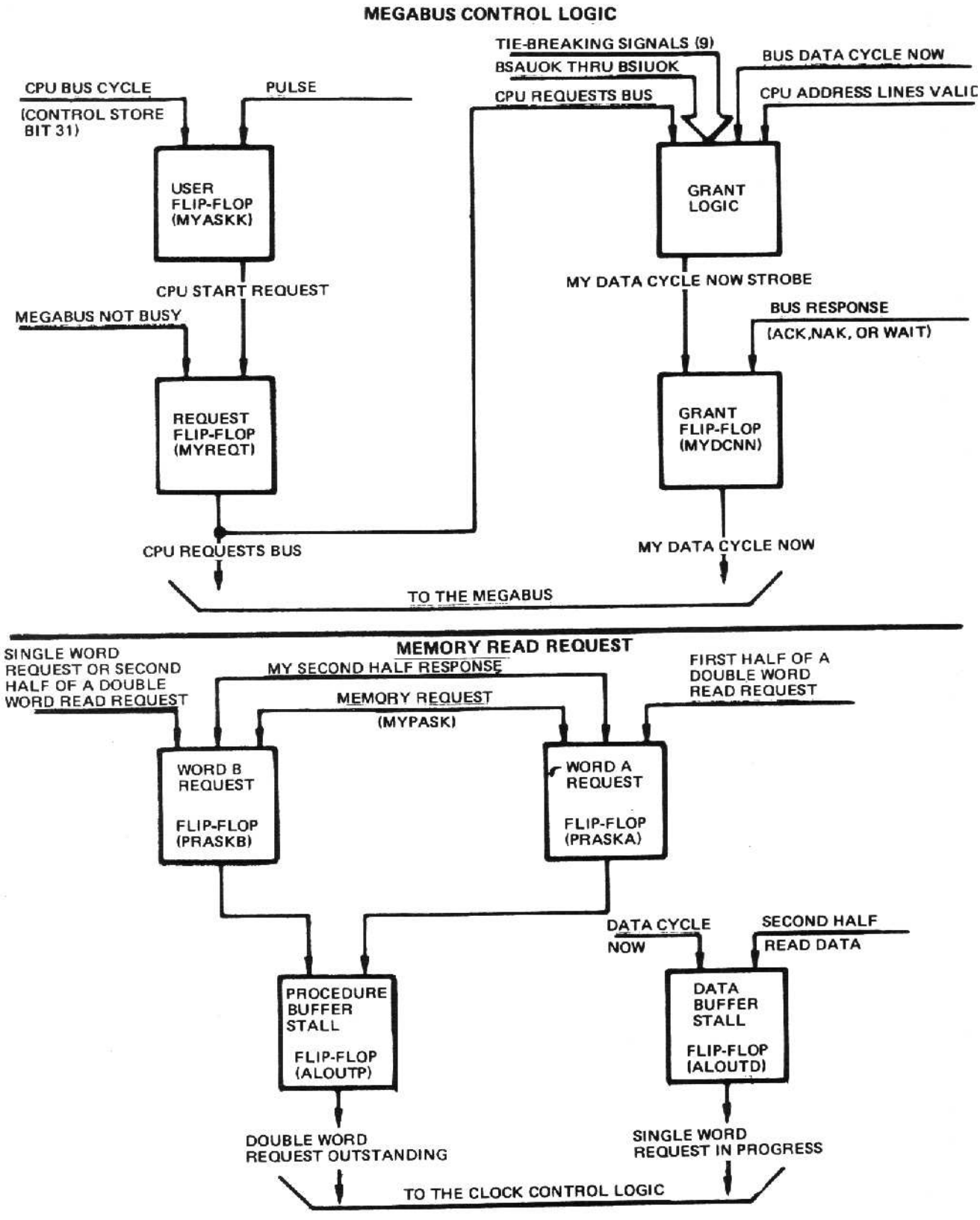


Figure 4-10 Megabus Control and Memory Read Request Logic

4.10.5 Response Summation

The functionality of the CPU logic shown in Figure 4-10 is similar to that of all units operating as a slave unit. However, the specific logic used for the generation of the three response conditions (ACK, NAK, and WAIT) depends on the design requirements of each individual unit. The following information summarizes slave unit response conditions.

Acknowledge Response (BSACKR)

An Acknowledge response (ACK) is generated if the slave unit is currently capable of accepting a bus transfer from the master unit.

Wait Response (BSWAIT)

A Wait response (WAIT) is generated if the slave unit is temporarily busy and cannot accept a transfer, but will be able to accept the transfer after a brief delay. On receipt of the WAIT signal, the master unit generates continual retries until the transfer is completed.

Negative Acknowledge Response (BSNAKR)

A Negative Acknowledge response (NAK) is generated if the slave unit cannot accept a transfer. On receipt of the NAK signal, the master unit does not attempt a retry.

No Response (ACK, NAK, or WAIT)

If no unit on the bus recognizes a transfer, no response occurs. Under these conditions, the CPU (which monitors all bus transfer) awaits a legitimate signal for approximately 5 microseconds, after which time it clears the bus with a NAK signal; the NAK signal is generated by the setting of the Bus Time-Out flip-flop in the CPU logic.

4.10.6 Megabus Registers

The Megabus registers consist of two procedure buffers (P1 and P2) and one data buffer (BD).

4.10.6.1 P1 and P2 Buffers

The P1 and P2 buffers (BIP110 through BIP11F and BIP210 through BIP21F) receive procedure words from the Megabus as a result of double-word requests. Sixteen bits are received from Megabus data lines 0 through 15, while the remaining four bits are loaded from two integrity bits (BSREDD and BSYELO) and two parity bits (BSDP00 and BSDP08).

The P1 and P2 buffers are loaded by the signals My Second Half P1 and P2 (MYSHP1 and MYSHP2, respectively). MYSHP1 is

generated when the first BSSHBC signal is received by the CPU, and MYSHP2 is generated when the second BSSHBC signal is received.

The P1 and P2 buffers are unloaded onto the internal bus by signals Enable P1 to BI or Enable P2 to BI (ENP1BI and ENP2BI, respectively).

The signal PRTAKC controls which register is unloaded onto the bus during a Procedure Stall operation. If PRTAKC is true, ENP1BI is forced true and the P1 buffer is unloaded; if PRTAKC is false, ENP2BI is forced true and the P2 buffer is unloaded.

The 20-bit output of the P1 buffer is labeled Bus Internal P110 through 1F (BIP110 through BIP11F), integrity signals Red and Yellow (BIP1RD through BIP1YL) and Parity signals 8 and 0 (BIP1P8 and BIP1P0). The 20-bit output of the P2 buffer is labeled BIP210 through BIP21F, BIP2RD and BIP2YL, and BIP2P8 and BIP2P0.

4.10.6.2 BD Buffer

The BD buffer, like the P1 and P2 buffers, receives as input from the Megabus BSDT00 through BSDT15, BSREDD and BSYEL0, and BSDP08 and BSDP00. The operand data from the Megabus is distributed throughout the CPU via the internal bus. The BD buffer is loaded by My Second Half Read (MYSHRD) when the BSSHBC signal is received by the CPU during a single fetch operation. It is unloaded onto the internal bus by Enable Data Buffer to BI (ENDTBI), which comes true when the Data Stall (CRDSTL) signal is true.

The 20-bit output from the BD buffer is labeled BIDT10 through BIDT1F, BIDTRD, BIDTYL, BIDT08, and BIDT00.

4.11 INTERRUPT CONTROL LOGIC

The interrupt control logic is divided into three areas:

- Address compare logic
- Level check logic
- Interrupt register.

4.11.1 Address Compare Logic (see Figure 4-11)

An interrupt message on the Megabus is accepted for examination by the CPU if the currently active data cycle (enabled by the interrupt requesting unit) has not been initiated by the CPU itself and does not require a memory reference. If these conditions are satisfied, the Bus Address Compare (BSACMP) signal enables a comparison of the CPU internal channel identification number (supplied by the hexadecimal rotary switch) and Megabus address bits 14 through 17 (least significant four bits of slave unit channel number) to determine if the interrupt message is

intended for this CPU. If the compare operation is successful (A=B) and Megabus address bits 8 through 13 equal Zero (indicating a CPU type channel number), a CPU address compare signal is generated and fed to the strobe select circuits to participate in determining whether the Megabus format reflects a second half bus cycle or an external interrupt. A second half bus cycle generates the Second Half Cycle Strobe signal (MYSHCS) which, in turn, sets the Second Half Read Cycle Flop (MYSHRC). An external interrupt generates the Interrupt Strobe signal (MYINTS) which, in turn, sets the Interrupt Received flop (MYINTR), providing a clock signal for both the negative acknowledge response and interrupt flip-flops (refer to subsection 4.11.2).

At this point, a decision is made to either accept the interrupt for service by allowing the interrupt flip-flop to set, or to delay acceptance by allowing the negative acknowledge response flip-flop to set. The decision is based on a comparison of the interrupt request's priority level to the current operating level of the CPU, and the state of the interrupt busy flip-flop (refer to subsection 4.11.2).

4.11.2 Level Check Logic

The level check logic (see Figure 4-12) compares Megabus data bits 10 through 15 (level of incoming request), S register bits 10 through 15 (current CPU operating level), and the state of the Interrupt Busy flip-flop (MYINTB). If the priority of the incoming request is equal to or lower than the current operating level of the CPU and/or the interrupt busy flip-flop is set (indicating that the CPU is currently servicing a previous interrupt), the output from the comparator sets the No Acknowledge Response flip-flop (MYNAKR), delaying acceptance of the interrupt. However, if the incoming priority level is higher than the current CPU operating level and the interrupt busy flip-flop is not set, the comparator output sets the Interrupt flop-flip (MYINTF). At this point, the interrupt is accepted and the interrupt busy flip-flop is set.

The interrupt busy flip-flop is also set when a power failure occurs and the CPU is not operating at the hghest (zero) priority level.

4.11.3 Interrupt Register

The interrupt register (see Figure 4-13) receives the level of the incoming interrupt request over Megabus data lines 10 through 15 when the interrupt request is accepted by the CPU (i.e., when the interrupt flip-flop is set). This action allows firmware to process the request subsequent to an acknowledgment. Also, if the Megabus format reflects a second half bus cycle rather than an external interrupt request, Megabus data lines 10 through 15 are fed to the Megabus data buffer when the second half read cycle flip-flop is set.

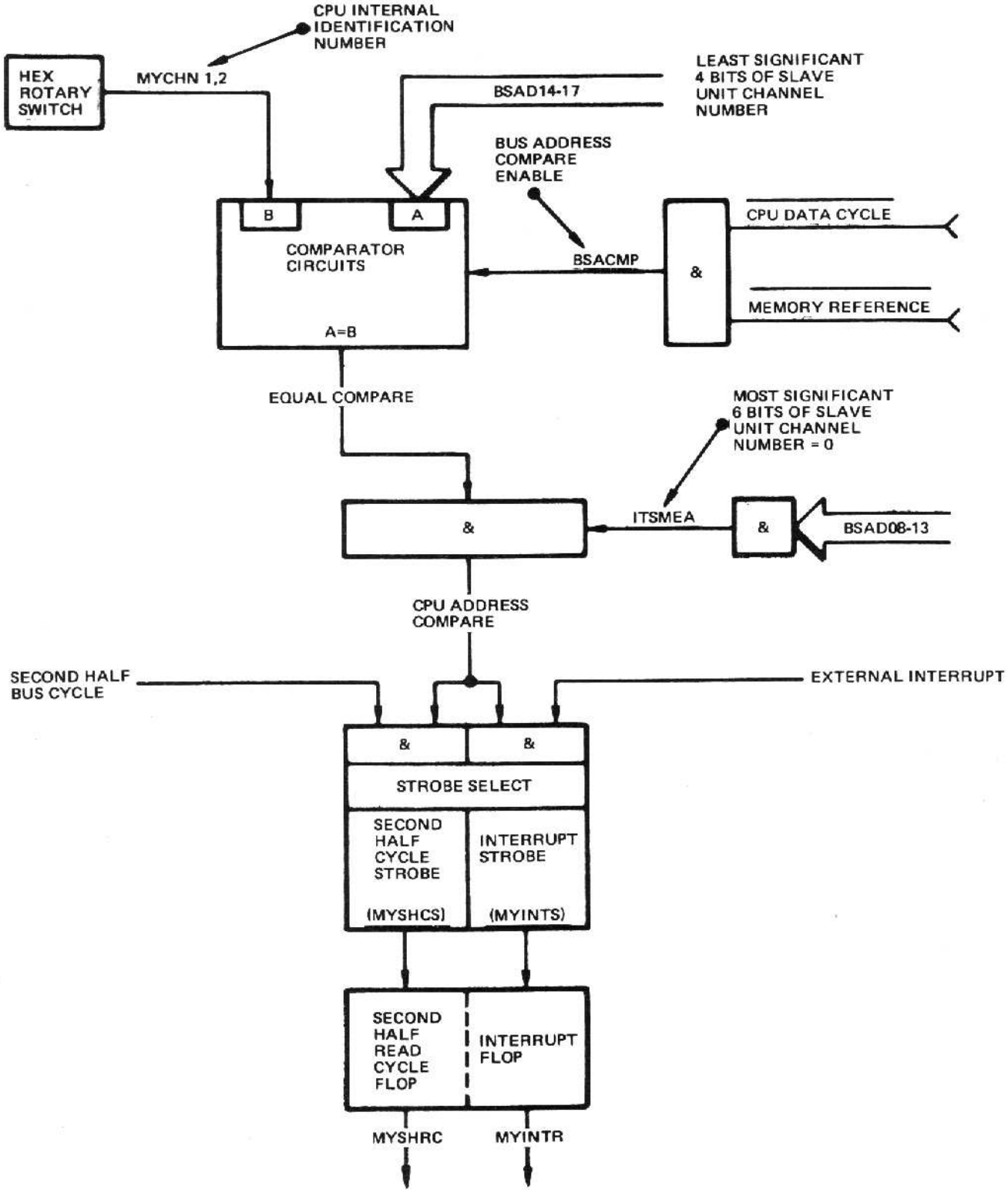


Figure 4-11 Address Compare Logic

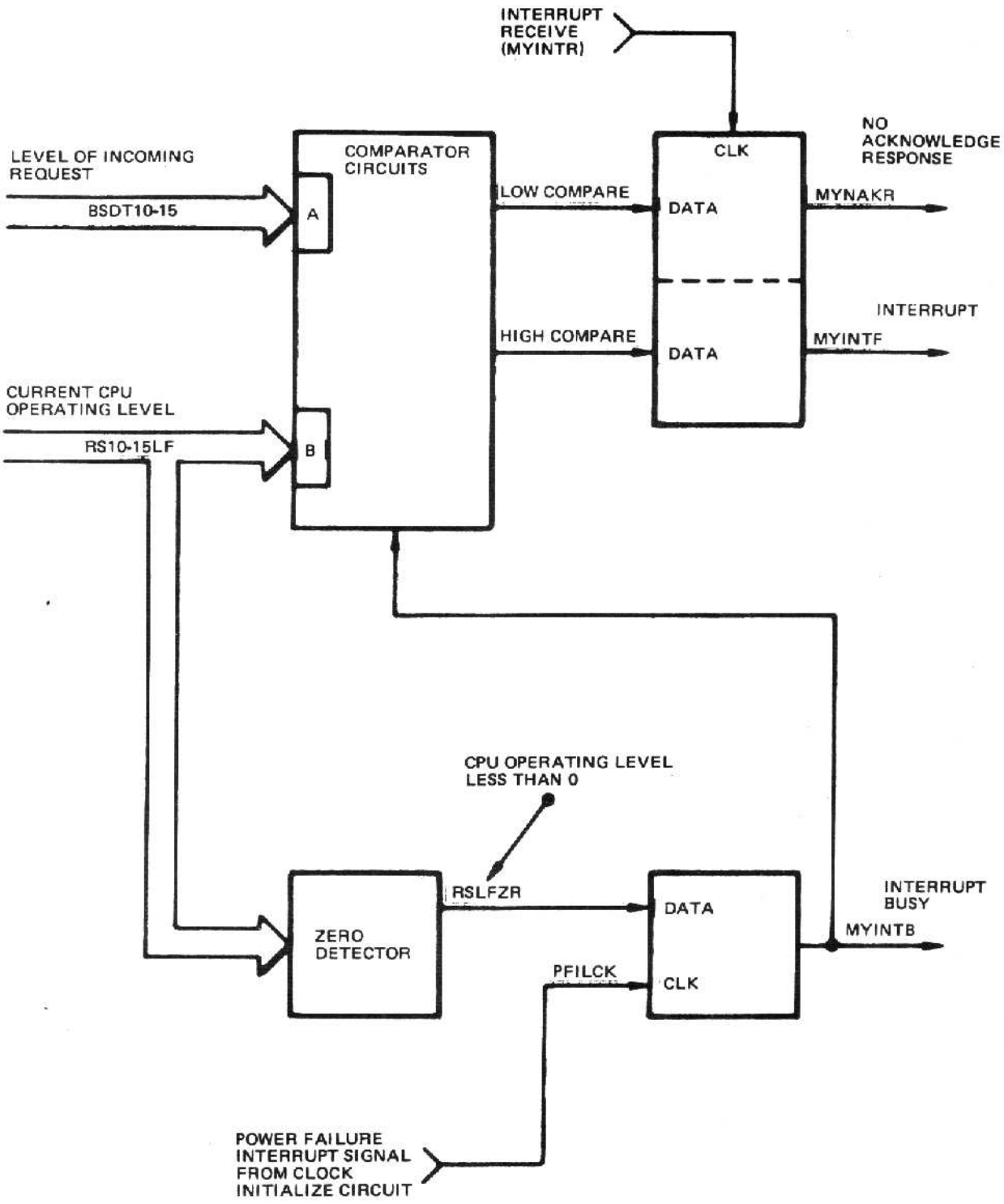


Figure 4-12 Level Check Logic

The output from the interrupt register (BINN10 through BINN1F) is delivered to the internal bus by the Enable Interrupt to BI (ENINBI) signal, indicating that the CPU is ready to service the interrupt.

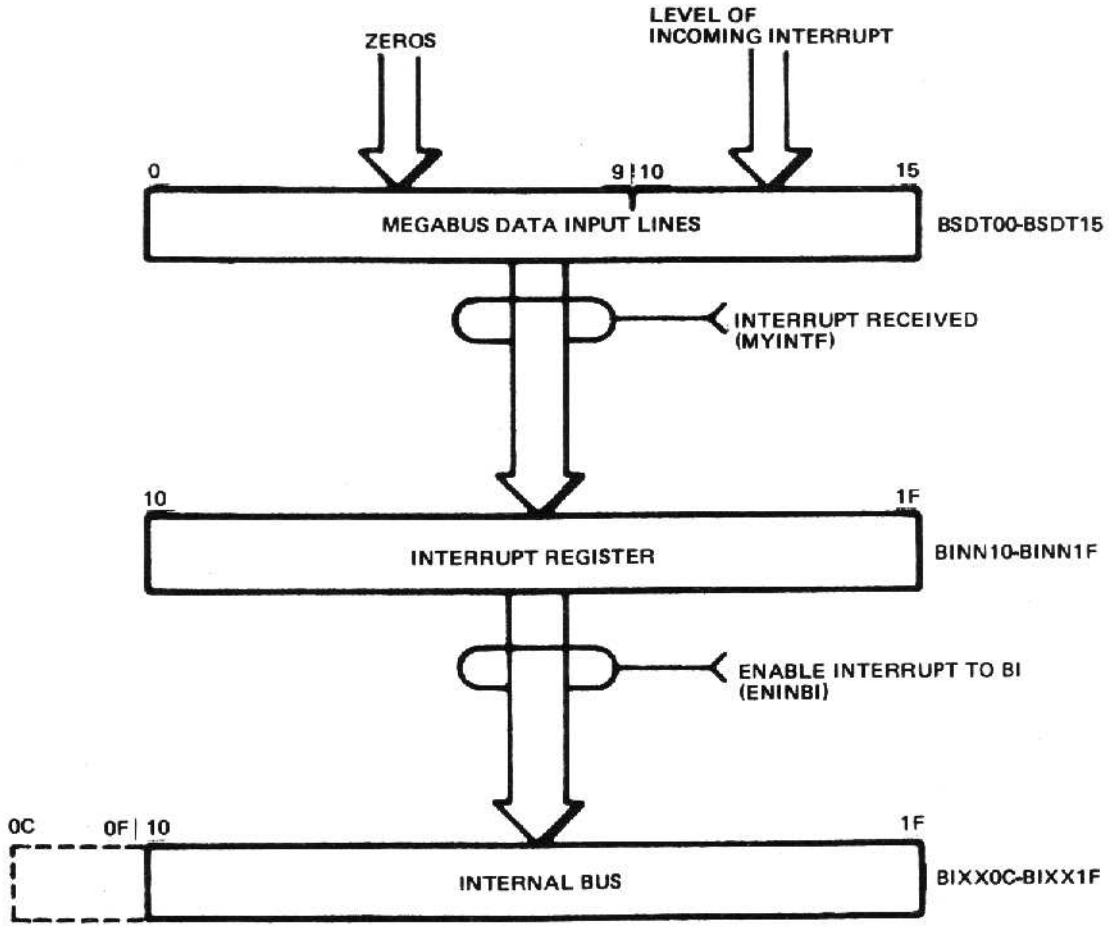


Figure 4-13 Interrupt Register

4.12 FULL CONTROL PANEL

This subsection provides a description of the full control panel physical characteristics and principles of operation. A description of the basic control panel is not included because its limited operating controls and indicators are similar to the corresponding full panel controls and indicators (refer to subsection 2.1.9.9.1). To obtain a comprehensive understanding of each basic panel control and indicator function, the reader should reference the applicable diagrams for the basic control panel and the corresponding text for the full control panel.

NOTE

The material contained herein assumes that a full control panel is connected to an operational CPU, the CPU is powered up, and the panel is active (unlocked).

4.12.1 Physical Characteristics

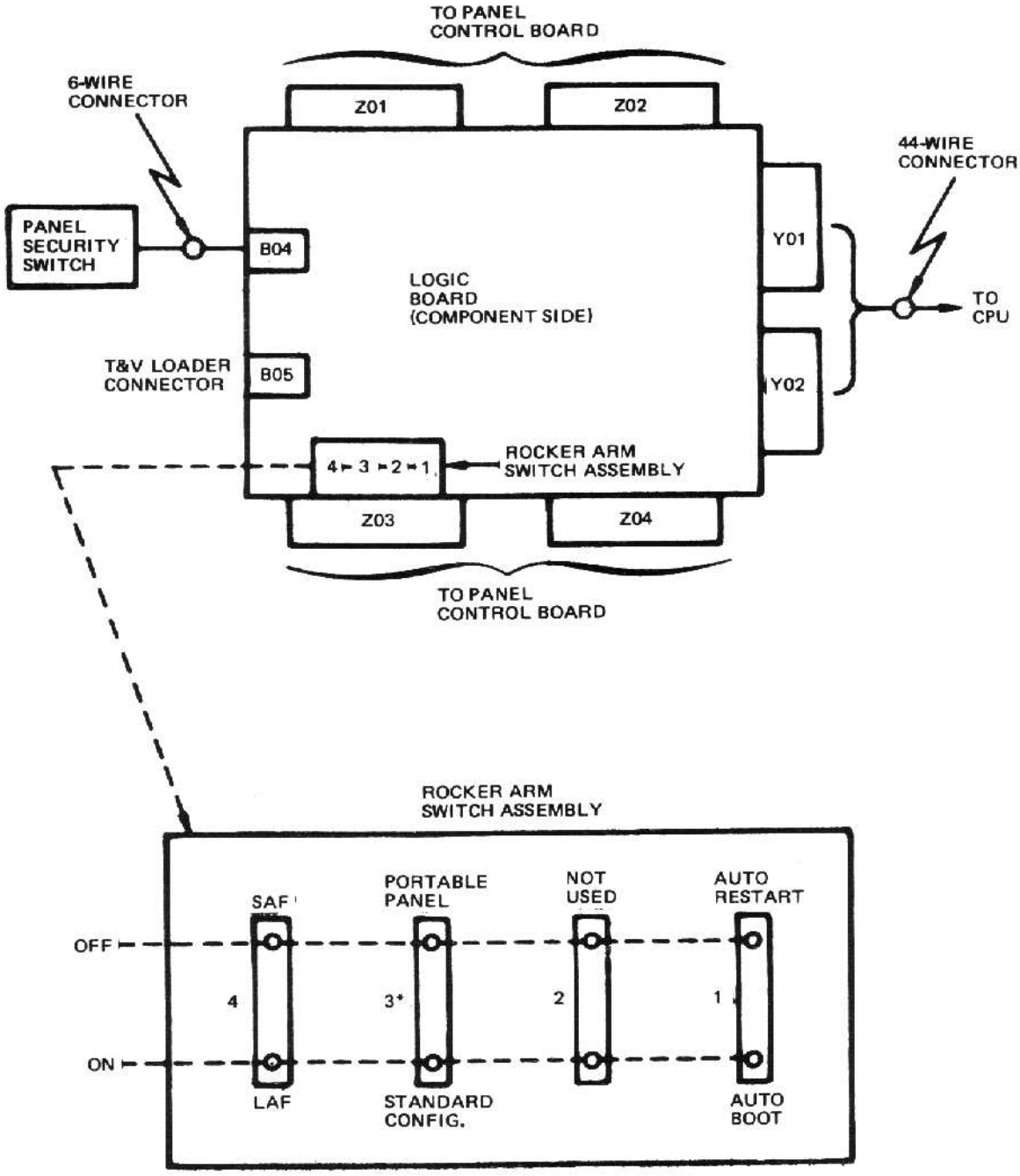
The control panel is a self-contained unit, consisting of one control and one logic board. The control switches, system status, and register display indicators are mounted on the control board; their associated circuit components and configurations are mounted on the logic board. The control switches, status indicators, and display indicators are as follows (see Figure 2-2 for the physical location of each component):

- POWER on/off switch
- PANEL SECURITY lock/unlock
- LOCATION display (register address)
- CONTENTS display (register data)
- CPU controls and indicators
- Key-pad array.

Four connectors provide the direct signal interface connection between both boards (see Figure 4-14). The rocker arm switch assembly is mounted on the logic board (refer to Table 2-1 for correct switch settings). In addition, the logic board connects to the CPU with a 44-wire connector and to the PANEL SECURITY switch with a 6-wire connector. The 44-wire pin-to-pin cable connections are shown in Figure 4-15. The PANEL SECURITY and POWER switches are connected to the panel frame on the left side of the control board.

4.12.2 Principles of Operation

Figure 4-16 illustrates the data flow, control signal development, and signal interface between the CPU and control panel, and should be referred to throughout the text as needed. The principal elements of the panel are the key-pad array and the LOCATION/CONTENTS display. All other elements support each register and key-pad, and produce the control panel internal and CPU interface control signals.



*SWITCH 3 IS NOT IMPLEMENTED ON THE BASIC PANEL.

Figure 4-14 Interconnections and Rocker Arm Switch Assembly

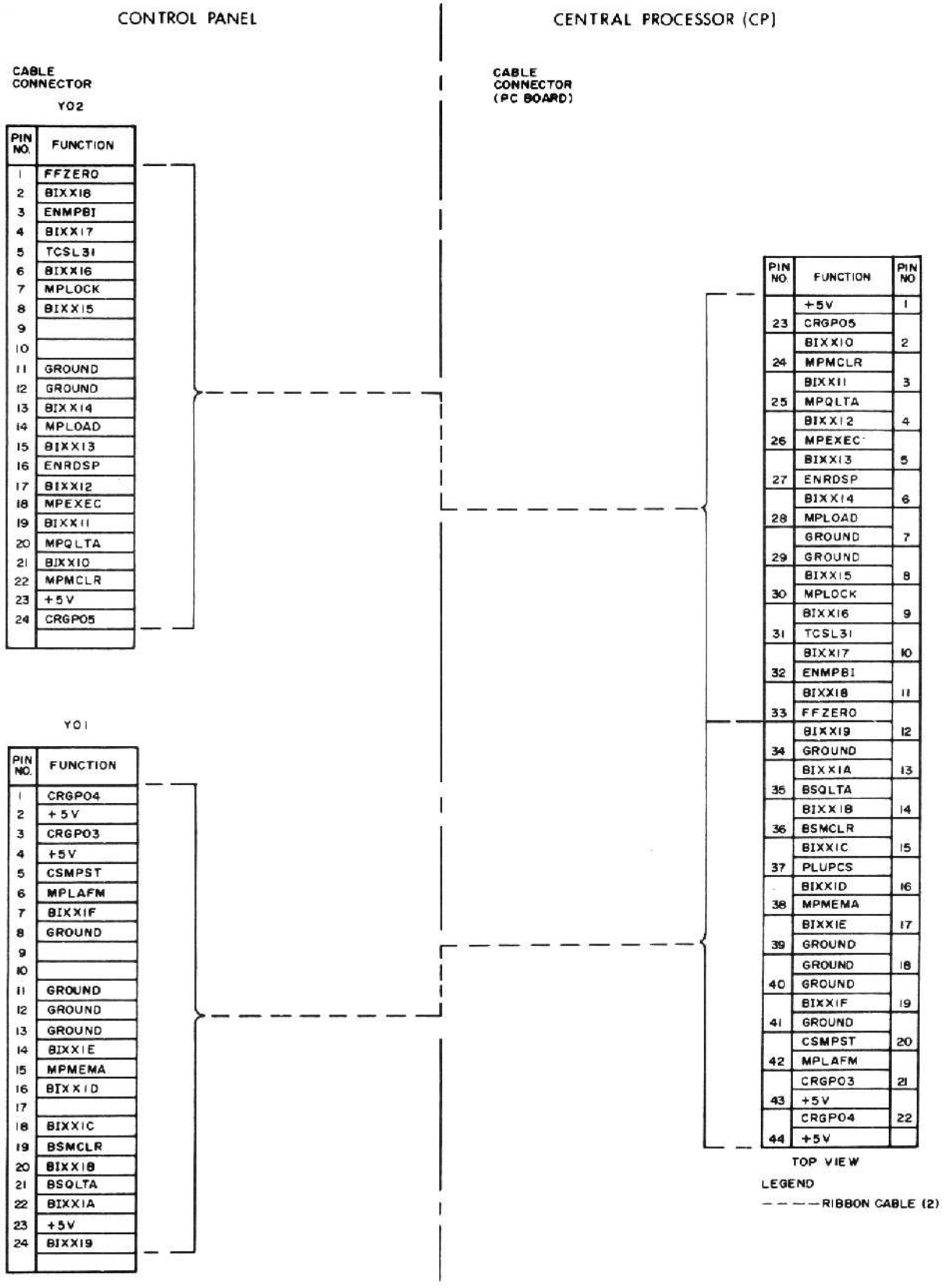


Figure 4-15 Control Panel to CPU Signal Cable Connections

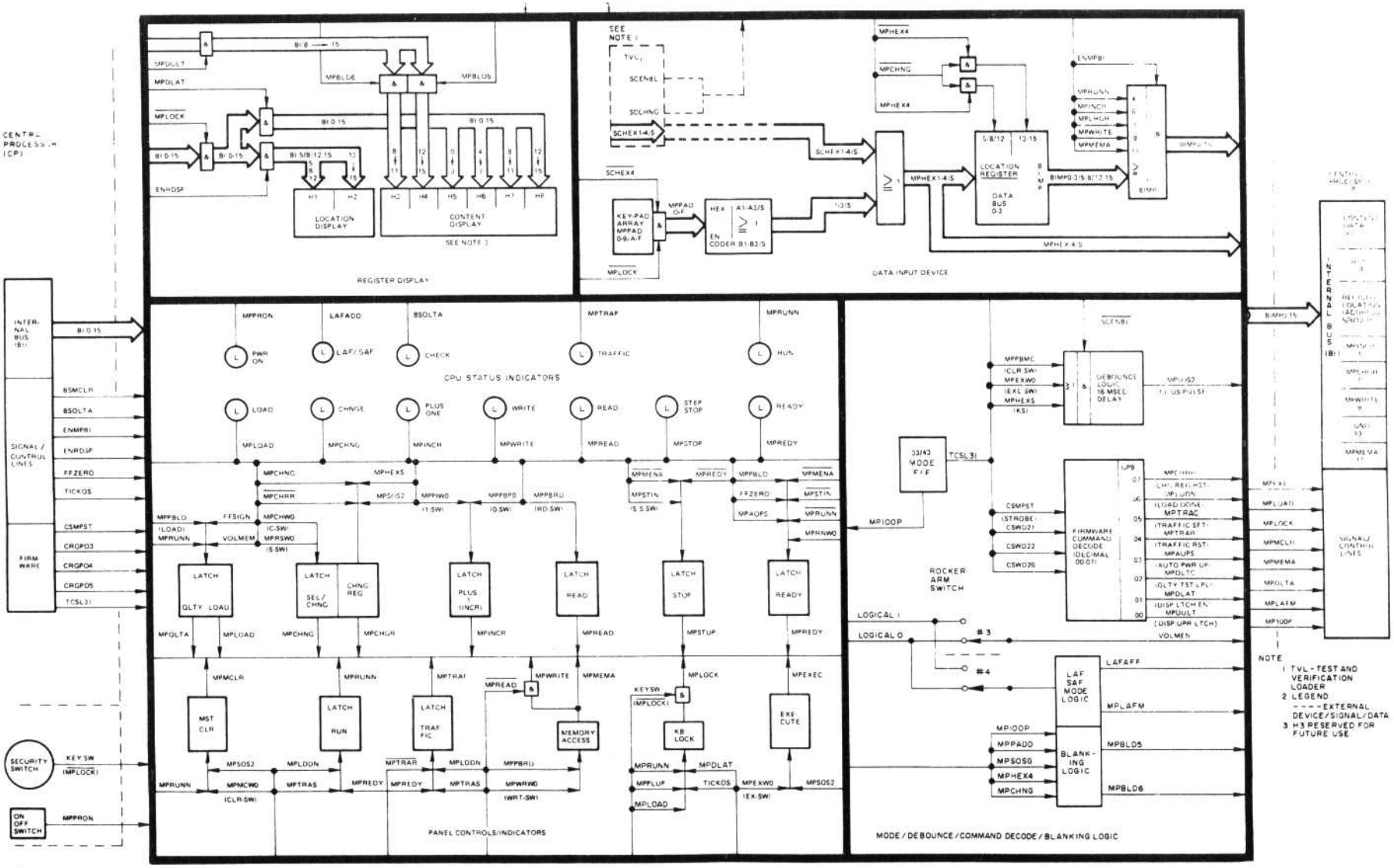


Figure 4-16 CPU Panel Control Logic

4.12.2.1 Control Panel Functionality

All control panel activity is governed by CPU firmware to allow manual operations from the panel when the CPU is in its Stop state, and to automatically update the respective control panel displays and indicators at 8-millisecond intervals when the CPU is in its Run state. Also, with the CPU in its Run state, the firmware acknowledges the following two manual functions from the panel to permit selection of an addressable CPU register for display in the panel displays, or to stop the CPU for manual intervention:

- Select mode: Manual selection via the Select (S) pushbutton
- Stop mode: Manual selection via the STOP (S) pushbutton.

All data and signal communication between the CPU and control panel is through 16 Internal Bus (BI) and 28 unique control panel transmission lines. The CPU gates (ENMPBI) panel data (see Table 4-5) onto the internal bus, and simultaneously gates (ENRDSP) address data from the internal bus into the control panel LOCATION display. All data character transfers between the panel and internal bus are controlled by specific control panel selections (e.g., Read or Write Memory) and the resident CPU control panel firmware.

Table 4-5 Control Panel to CPU Internal Bus Bit Assignments

BIT(S)	FUNCTIONS	REMARKS
0-3	CPU register data	From panel data bus
4	Run mode (true); Stop mode (false)	-
5	Panel location register data	Refer to bits 8 and 12 through 15
6	Increment (true); Nonincrement (false)	-
7	Change mode (true); Select mode (false)	-
8	Panel location register data	Refer to bits 5 and 12 through 15
9	Write mode (true); Read mode (false)	-
10	N/A	Ground
12-15	Panel location register data	Refer to bits 5 and 8

4.12.2.2 Data Input Devices

A key-pad array and a Test and Verification Loader (TVL) provide a user with two optional data input devices to the CPU. The TVL is primarily intended to load diagnostic programs from a tape cassette into main memory. For a complete explanation of the TVL, refer to the Type LDU9101 Test and Verification Loader Manual (Order No. FL97).

Sixteen single-action pushbutton switches (keys) comprise the key-pad array. Each key is assigned a unique hexadecimal character (i.e., 0 through 9/A through F) that is inscribed on its keycap; the keys are used to input data to a hexadecimal encoder. Activating a key (keystroke) generates its corresponding hexadecimal character code (MPHEX1 through MPHEX4) and a unique keystroke (MPHEXS) signal at the encoder output to form a register address or a hexadecimal data character to modify a register. For further details on the key-pad array, refer to the discussion entitled Select/Change Modes in subsecton 4.12.2.4.3.

4.12.2.3 Register Displays

The location and contents indicator fields from the control panel display network enable the user to interrogate or modify any of the selected CPU registers (see Table 2-2).

4.12.2.3.1 LOCATION Display

The LOCATION display consists of two hexadecimal indicator assemblies that are designed to display hexadecimal characters H1 and H2. Both indicators monitor the internal bus to display (ENRDSP) the selected CPU register address that resides in the control panel location register (see Figure 4-16).

4.12.2.3.2 CONTENTS Display

The CONTENTS display consists of five hexadecimal indicators (H4 through H8) as shown in Figure 4-16, and is designed to display the hexadecimal character contents of a CPU addressable 16-bit or 20-bit register (see Table 2-2). Data is stored into the display from internal bus bits 0 through 15 when the Display Latch Enable (MPDLAT) and/or the Display Upper Latch (MPDULT) command is received and decoded in the panel.

4.12.2.4 Panel Controls/Indicators

To implement a panel initiate CPU operation, the CPU at predetermined intervals gates, via ENMPBI, the key-pad output and current panel status (see Table 4-5) onto the internal bus for firmware interrogation and implementation of one of the following CPU operational modes:

- Stop/Ready/Run
- Read/Write Memory
- Select/Change
- Plus 1 (Increment) Plus 0 (Non-Increment).

Concurrent with the preceding operations, the CPU has a fifth operational mode called LAF/SAF. In this mode, the CPU monitors the panel transmission lines to condition itself for or to implement one of the following panel initiated CPU operations:

- Load/Restart
- Execute.

The manner in which each panel status condition, its corresponding operation, and the associated indicators are activated is described in subsequent paragraphs. A description of two specific CPU status indicators (CHECK and DC ON) and the operation of the Clear (CLR) pushbutton is provided in Table 2-1.

4.12.2.4.1 Stop/Ready/Run Modes

Run mode enables the CPU to process instructions. To enter the Run mode, the CPU must first be initialized to its Stop (MPSTOP) state and the panel Ready (MPREDY) signal must be activated. Stop mode allows manual intervention and single instruction execution from the panel, and is activated by depressing the Stop (S) pushbutton (MPPBST). This causes the CPU to complete its current instruction and to perform the following:

- Halt all memory resident program processing activity
- Enter a continuous panel service loop
- Clear the RUN, READY, and TRAFFIC indicators
- Update the Instruction (D0) register with the next instruction for execution
- Equate the program counter to one greater than the next instruction memory location address
- Illuminate the STOP/STEP indicator.

To enter the Ready state, depress the control panel Ready (R) pushbutton (MPRUNN); this illuminates the Ready indicator and enables both the RUN and TRAFFIC indicators. Depressing the EXECUTE (E) pushbutton causes the CPU to process instructions, beginning with the instruction contained in the CPU instruction (F) register. Then the CPU control panel firmware generates a Traffic Set (MPTRAC) command, which is decoded in the panel to illuminate the RUN indicator, and to illuminate the TRAFFIC indicator. The Ready/Run states are also activated when a panel or CPU initiated Load operation is implemented.

4.12.2.4.2 Read/Write Mode

The following two control panel prerequisites are assumed completed by the user:

- The Increment (Plus 1)/Nonincrement (Plus 0) mode is enabled (refer to the following discussion).
- The Memory Address (A0) and Bus Data (B0) registers are initialized (refer to the TVL manual*) for the proper display in the LOCATION and CONTENTS displays, respectively.

Activating a control panel Read (R) or Write (W) operation: (1) clears Increment ($\overline{\text{MPINCR}}$) and CPU Stop mode ($\overline{\text{MPSTOP}}$), (2) conditions, via MPMEMA, the CPU for a Read/Write memory cycle through the internal bus and (3) illuminates the selected READ or WRITE indicator. Also, activating Write (MPWRW0) clears the control panel Load/Restart state. Depressing the Execute (E) pushbutton initiates a CPU memory cycle to perform a preselected control panel Read or Write operation. A Read memory ($\overline{\text{MPWRITE}}$) cycle fetches data from a preselected memory location and stores it in the bus data register (CONTENTS display). A Write memory (MPWRITE) cycle takes data from the bus data register and stores it in a preselected memory location. Each successive depression of the Execute (E) pushbutton repeats a Read or Write memory cycle at the same or next sequential memory location as determined by the Plus 1 (Increment)/Plus 0 (Nonincrement) pushbutton selection.

4.12.2.4.3 Select/Change Modes

The Select/Change modes allow a user to select, display, and modify (from the key-pad array) selected CPU registers as described in the following text. Visual display of a register address and its contents is provided in the LOCATION and CONTENTS displays, respectively.

Select Mode

Activating Select (S) mode: (1) clears change (C) mode (MPCHGR), (2) extinguishes the CHANGE indicator, and (3) notifies the CPU control panel firmware through the CPU internal bus. It also enables, via $\overline{\text{MPCHNG}}$, the location register to accept key-pad data to form one of 49 register addresses (see Table 2-2 and Figure 4-17).

To form a legitimate CPU register address, the location register is configured to accept alpha hexadecimal characters 8 through F in its leftmost character position (H1), and numeric hexadecimal characters 0 through 7 in its rightmost character

*The TVL manual is entitled: Type LDU9101 Test and Verification Loader Manual (Order No. FL97).

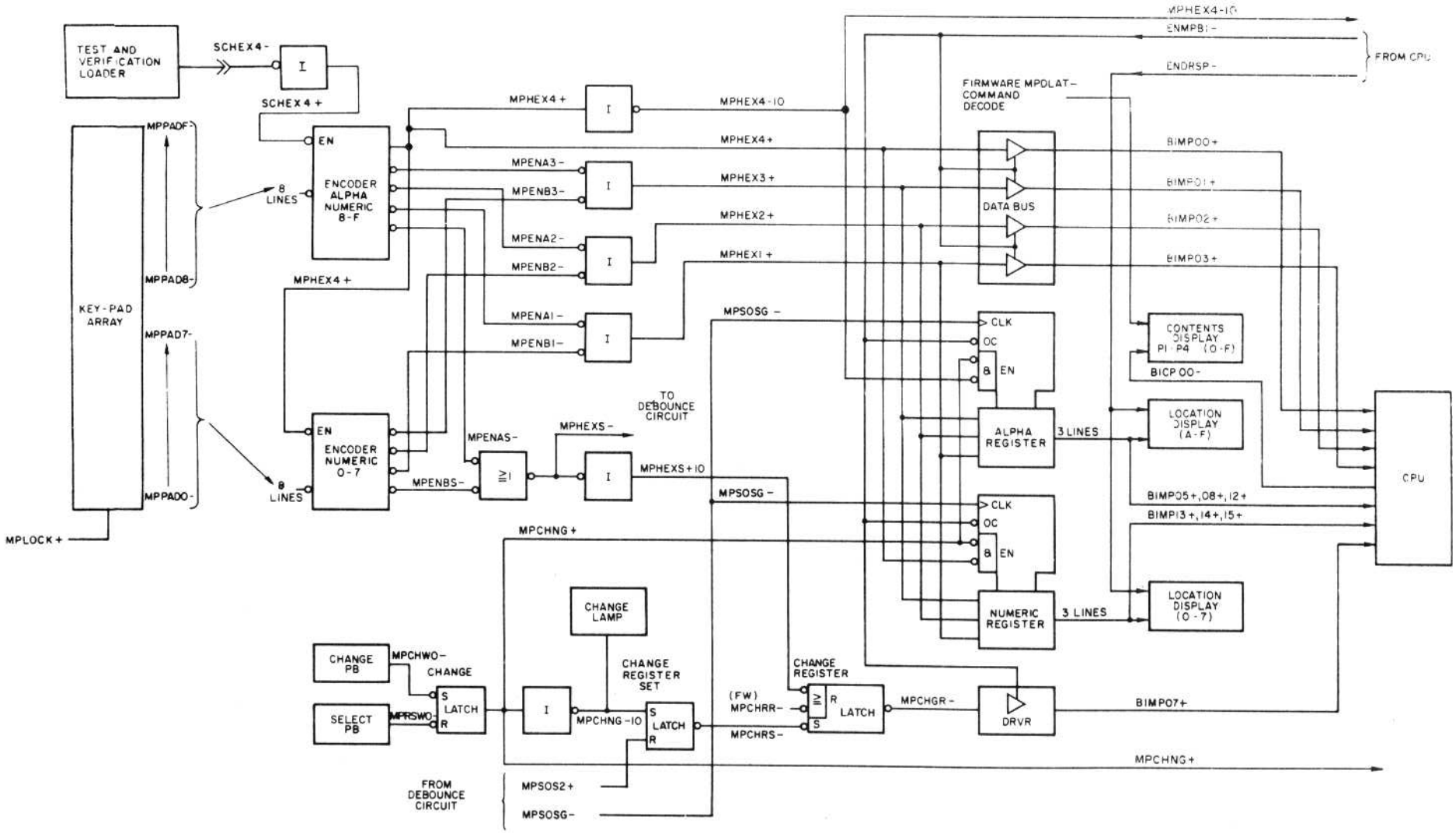


Figure 4-17 Key-Pad Array Logic

position (H2). Each additional depression of an alphanumeric character overlays a character in its respective location register position. This operation is allowed in both the CPU Stop and Run modes.

Change Mode

Activating Change (C) mode (MPCHNG): (1) clears Select mode, (2) illuminates the CHANGE indicator, and (3) notifies firmware through the internal bus. It also allows a user to change the contents of the following addressable CPU registers: A0, B0 through B7, D0 through D7, and E0. All remaining addressable CPU registers (e.g., C0 through C7) are read only type registers. Only 4 characters are needed to fill the data registers (D0 through D7). Base register (B0) requires just 4 characters when used as the source data in a memory panel write operation. The remaining changeable registers (A0, B1 through B7, and E0) are designated as address registers and 5 characters are entered when filling the selected register from the control panel.

Key-pad input activity to a selected CPU register occurs in the following manner and can be observed in the CONTENTS display. Depressing characters 0 through F activates, via MPHEXS, a 6-millisecond debounce network, which generates a 1.6-microsecond (MPS0S2) pulse. This is to allow sufficient time to stabilize any keystroke line transients and to activate a CPU flag (MPCHGR) that is required by the firmware to process a data character. With each depression of a key: (1) the CPU shifts all four/five data characters in the selected CPU register left by one character position, (2) truncates the leftmost character (H4), and (3) stores the corresponding character in the rightmost character position (H8). This process is repeated for each subsequent depression of a key.

NOTE

Refer to Table 2-2 in determining the number of characters required to fill the selected register. To avoid entering the wrong data, exercise care when entering the desired data pattern.

4.12.2.4.4 Increment/Non-Increment Functions

The Increment (MPPOW0+, Plus 1) and Nonincrement (MPPBP0+, Plus 0) panel functions enable firmware to initiate a Nonincrement or Increment Read/Write memory operation with each depression of the EXECUTE (E) pushbutton. Hence, a Read/Write memory cycle is performed at each new sequential memory address in an Increment operation (i.e., Plus 1 is on), or at the same memory address in a Nonincrement operation (i.e., Plus 0 is on).

4.12.2.4.5 Long/Short Address Form (LAF/SAF)

The central processor monitors the address mode (LAF/SAF) which is controlled by switch 4 of the rocker arm switch assembly. Switch 4 set to the LAF position, followed by a master clear (e.g., powering up or depressing CLR pushbutton), will cause the CPU address development to be 20 bits in length. Switch 4 set to the SAF position, followed by a master clear, causes the address fields to be 16 bits in length.

NOTE

With the portable panel and the basic panel installed in the same system, address mode control takes place in the basic panel. When LAF mode is selected, it will be displayed on both the portable panel and the basic panel.

4.12.2.4.6 Load/Restart Operation

Two fundamental CPU start-up operations, called Load and Restart, are included in the system to manually or automatically load a program (Bootstrap), or to restart the CPU from a predetermined memory location. A manual program Load (MPLOAD) is enabled with Stop mode active ($\overline{\text{MPRUNN}}$) and when the control panel Load (L) pushbutton (MPPBLD) is depressed. Actual execution commences with the depression of the Execute (E) pushbutton. This causes the CPU to load a program from the program designated input device (Default operation) or from a user-selected input device. In a Default operation, a system Quality Logic Test (MPQLTA) is included and is implemented with the first depression of the Execute (E) pushbutton. This illuminates, via BSQLTA, the CHECK indicator, which is extinguished when the test is successfully completed (see Table 2-1). Depressing the Execute (E) pushbutton a second time implements the loading of a program from the default device. A user may select an alternate input device after the Quality Logic Test (QLT) has been completed. First, depress the Stop (S) pushbutton and alter the contents of Data register 1 (D1) to the desired BOOT channel number. Then, in sequence, depress the Ready (R) and Execute (E) pushbuttons. When the preceding load activity is successfully concluded, both the RUN (MPRUNN) and the TRAFFIC (MPTRAF) indicators are illuminated.

When the PANEL SECURITY switch is on (MPLOCK) or a basic panel is installed, an automatic program load or CPU restart sequence is available. The sequence is designed to automatically restart an unattended CPU when ac source power is restored to the CPU after a power failure. In a CPU with a non-volatile memory (e.g., core memory or memory save power supply), program restart commences at memory location 0. When the CPU contains a volatile memory (VOLMEM-), the Default Load operation previously described is performed. When auto boot is performed, the Memory QLT (lower 8K) will be executed. When auto restart is performed, the Memory QLT will be bypassed.

NOTE

If both the portable panel and the basic panel are installed in the same system, auto boot/ auto restart control takes place in the basic panel.

4.12.2.4.7 Execute Operation

The Execute (E) pushbutton is enabled when the Stop mode and the panel are active to initiate one of the following panel selected operations in the CPU:

- Read/Write Memory
- Execute Single Instruction
- Program Load/Restart.

Depressing the Execute (E) pushbutton sends a 1.6-microsecond strobe pulse, 6-milliseconds after its initial depression, to the CPU to allow firmware to implement the selected function.

4.12.2.5 Panel Lock Request

The control panel, when active and requiring service (e.g., display update, register change, mode change), transmits its locked state to the CPU. The 8-millisecond update cycle is controlled internally by the CPU when in the Run mode.