TEXAS INSTRUMENTS

Improving Man's Effectiveness Through Electronics

Model 980 Computer Maintenance Manual

Arithmetic Unit And Control Console

MANUAL NO . 960699 - 9702
ORIGINAL ISSUE 15 JULY 1972
REVISED AND REISSUED 1 SEPTEMBER 1975

Digital Systems Division

LIST OF EFFECTIVE PAGES

INSERT LATEST CHANGED PAGES DESTROY SUPERSEDED PAGES

Note: The portion of the text affected by the changes is indicated by a vertical bar in the outer margins of the page.

Model 980 Computer Maintenance Manual,
Arithmetic Unit and Control Console (960699-9702)

Original Issue . . . . . 15 July 1972
Revised and Reissued . . . . . 1 September 1975

Total number of pages in this publication is 128 consisting of the following:

PAGE NO.	CHANGE NO.	PAGE NO.	CHANGE NO.	PAGE NO.	CHANGE NO.
Cover	0				
Eff. Pages	0				
iii - iv	0				
1-1 - 1-10	0				
2-1 - 2-2	0				
3-1 - 3-6	0				
4-1 - 4-86	0				
5-1 - 5-4	0				
Alphabetical Index Div	0				
Index-1 - Index-8	0				
User's Response	0				
Bus. Reply	0				
Cover Blank	0				
Cover	0				

PREFACE

This manual contains maintenance instructions for the Arithmetic Unit and the Control Console of either the Texas Instruments Model 980A or Model 980B Computer. Maintenance information is distributed within the remaining four sections of this manual as follows:

- Section I describes the general functions and characteristics of the Arithmetic Unit and Control Console.
- Section II describes the installation of the Arithmetic Unit and Control Console.
- Section III describes the operation and programming for the Arithmetic Unit.
- Section IV provides theory of operation for the Airthmetic Unit and the Control Console.
- Section V describes maintenance and trouble analysis procedures for the units.

Before using this manual for maintenance, review the System Description manual for the computer to gain a better understanding of the system interrelations. Maintenance documentation for the circuit boards is contained in the Parts List and Assembly Drawings manual, the Electrical Drawings manual, the Load, Pin and Wire List manual, and the Logic Documentation manual for the computer system. These manuals are described in the System Description manual listed below that applies to the particular computer:

- *Model 980A Computer Maintenance Manual, System Description*, TI Part Number 960699-9701
- *Model 980A Computer Maintenance Manual Electrical Drawings*, TI Part Number 960699-9707
- *Model 980A Computer Maintenance Manual Memory, Memory Controller and Direct Memory Access Channel*, TI Part Number 960699-9703
- *Model 980A Computer Maintenance Manual Load and Pin List*, TI Part Number 960699-9708
- *Model 960A/980A Computer Maintenance Manual Power Supply*, TI Part Number 226750-9705
- *Model 980B Computer Maintenance Manual, System Description*, TI Part Number 943012-9701
- *Model 980B Computer Maintenance Manual Electrical Drawings*, TI Part Number 943012-9708
- *Model 980B Computer Maintenance Manual Memory, Memory Controller and Direct Memory Access Channel*, TI Part Number 943012-9702
- *Model 980B Computer Maintenance Manual Load and Pin List*, TI Part Number 943012-9705

- *Model 960B/980B Computer Maintenance Manual Power Supply*, TI Part Number 942773-9703

- Program Descriptions, listings and operating procedures for the Performance Assurance Tests TI Part Number 961961-9770.

SECTION I

GENERAL DESCRIPTION

1.1 GENERAL

Figure 1-1 shows both the 980A and 980B control consoles and figure 1-2 shows the Arithmetic Unit printed circuit boards.

1.2 ARITHMETIC UNIT FUNCTIONS

The Arithmetic Unit (AU) contains logic for four major system functions:

- Instruction Execution
- System Interrupt Control
- Input/Output Bus Control
- Console Function Control.

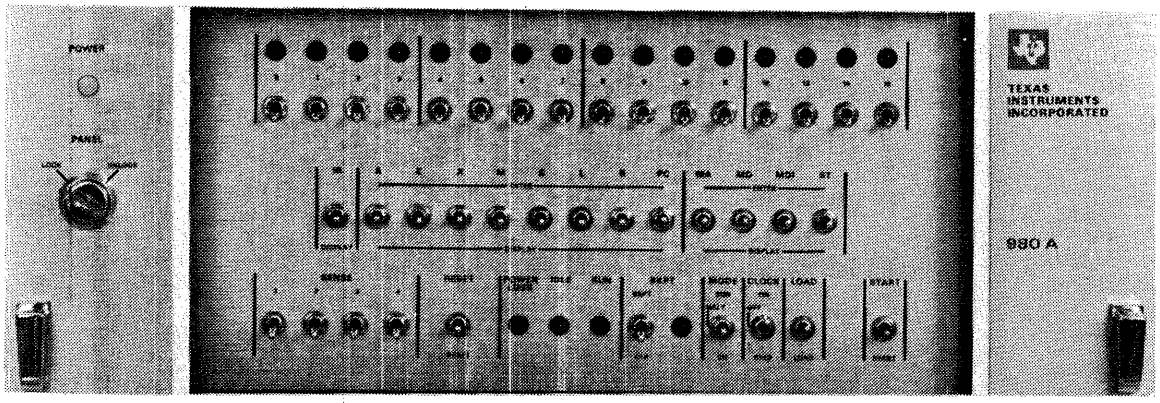
1.2.1 INSTRUCTION EXECUTION. Execution of the basic instruction set is the primary function of the AU. Machine instructions can be 16, 32, or 48 bits in length. The basic memory word contains 16 bits of binary data; therefore, 32 and 48 bit instructions must be accessed from 2 and 3 successive memory locations, respectively. The instruction formats are described in the *Model 980 Computer Assembly Language Programmer's Reference Manual* (TI Part Number 943013-9701).

After the AU has accessed the first instruction word, a legal instruction operation test is performed. All operand and address calculations are performed and the operation is executed. For the AU to perform this function, a 16-bit random access memory is required to store the program data and instructions.

1.2.2 SYSTEM INTERRUPT CONTROL. The interrupt feature provides a method of switching from one routine to another due to an internal or external condition. During program execution, the AU monitors the system for three standard types of interrupt and one optional interrupt type. The standard interrupt types are: internal, Direct Memory Access Channel (DMAC), and Input/Output bus interrupts. The optional interrupt feature adds up to 32 priority vectored interrupt lines. When any interrupt occurs, the AU detects the condition and, assuming interrupts are enabled, initiates appropriate action on a priority basis.

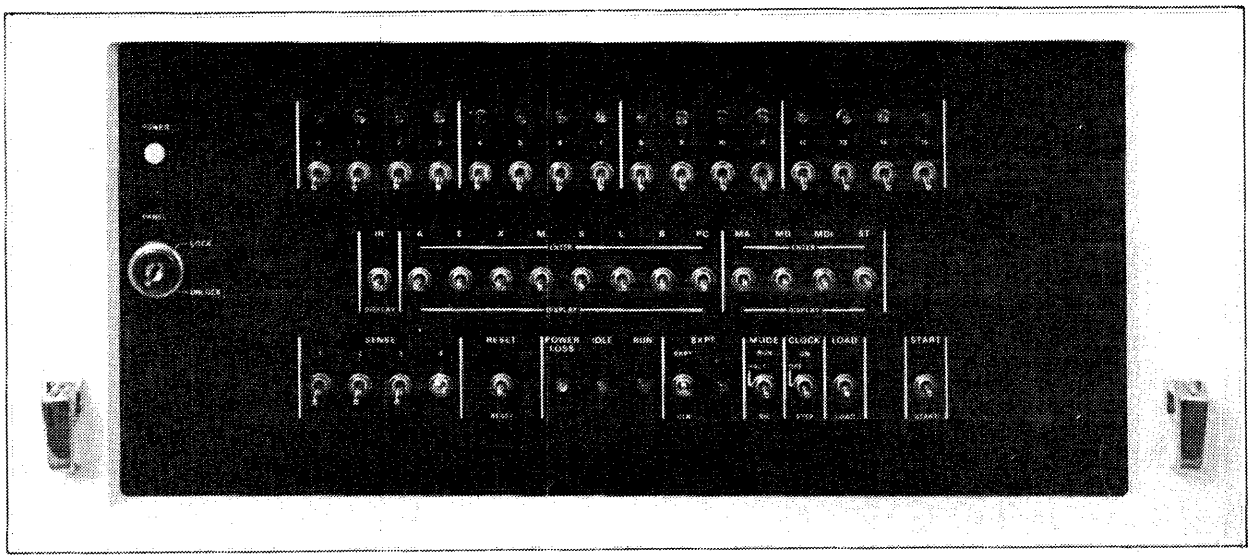
1.2.3 INPUT/OUTPUT BUS CONTROL. The 16-bit Input/Output (I/O) bus is used for single-word transfers between memory and a slow-speed I/O device. Each transfer requires the execution of an AU instruction. All timing and control for direct I/O data transfers is provided by the AU.

1.2.4 CONSOLE FUNCTION CONTROL. The AU also functions as controller for the control console. All console functions are sequenced by logic contained within the AU.



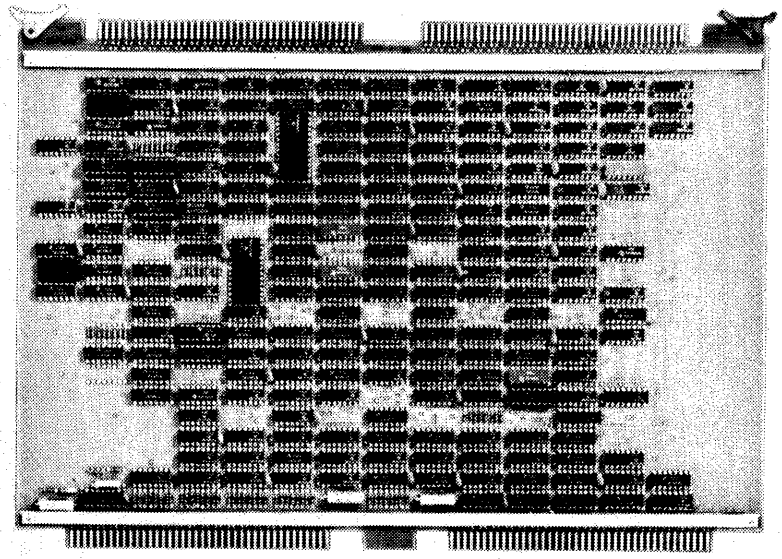
980A CONTROL CONSOLE

131705 (980-773-18-2)



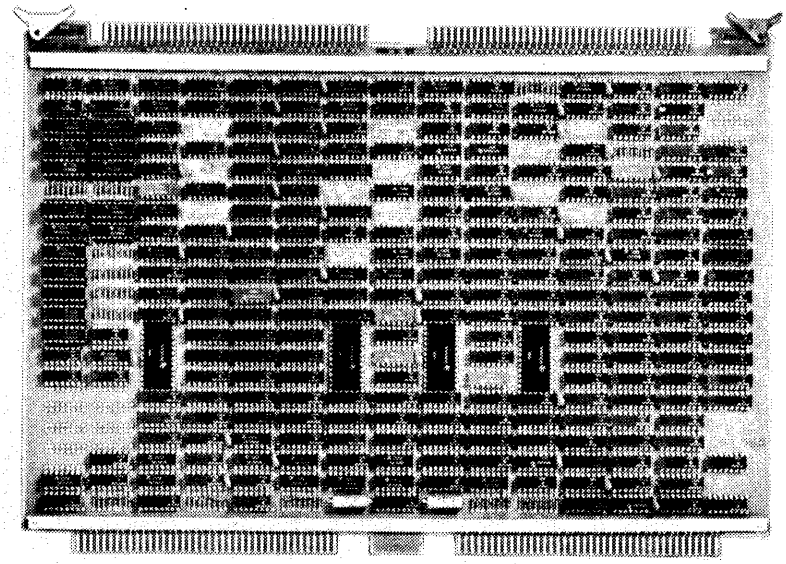
129829 (980-374-13-7)

Figure 1-1. 980A and 980B Control Consoles



AU1

131706



AU2

131707

Figure 1-2. Arithmetic Unit Printed Circuit Boards No. 1 and No. 2

1.3 ARITHMETIC UNIT CHARACTERISTICS

All circuitry in the Arithmetic Unit consists of small and medium scale integrated circuits of transistor-transistor logic (TTL). The plastic dual-in-line packages are mounted on two multilayer printed circuit boards (TI Part Number 960754(AU1) and 960751(AU2)). All AU logic is designed to function synchronously with a single 4.0-MHz clock signal which is obtained from the memory controller board.

1.3.1 ARITHMETIC UNIT LOCATION AND CONNECTIONS. The two AU boards (AU1 and AU2) have dual 80-pin connectors on the bottom-edge that plug into the computer connector plate slots as shown in the mechanical configuration that is illustrated in Section II—Installation.

The +5 Volts and ground signals on the connector plate printed circuit board are bussed on etched paths to the AU board connectors.

All bottom edge connector signals between the AU connector pins and other system units are carried through wire-wrapped connections. Partition signals, required between the two AU boards, are brought to the top-edge connector pins and are carried through the one-to-one printed circuit connector boards (TI Part Number 960961).

A basic functional diagram of the AU and related units that interconnect to the AU through the connector plate wiring is illustrated in figure 1-3.

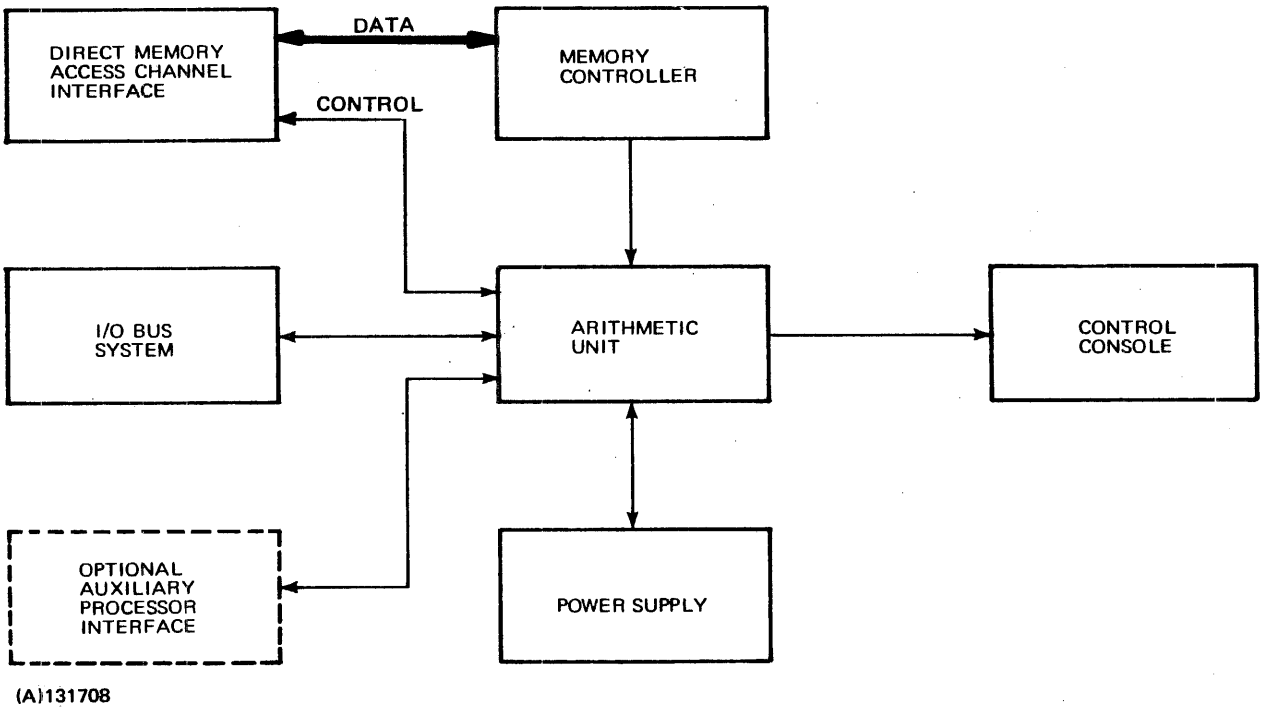


Figure 1-3. Arithmetic Unit System Interface, Functional Diagram

1.3.2 **ARITHMETIC UNIT CONTROL.** Logic control functions in the AU are performed with a TTL Read-Only Memory (ROM). The control ROM consists of 256 words with 72 bits-per-word. A logic state in the AU is identified by the present address of the ROM controller. Major control signals that are required for state functions are taken from the contents of the 72-bit ROM word being addressed as a given state. Micro sequencing is performed by changing the ROM address as determined by the state transfer logic. State transfers or ROM address changes occur on the negative edge of the True system clock pulse to maintain synchronization.

A functional diagram of the AU control configuration is illustrated in figure 1-4. In a given logic state, the present ROM address is required by the AU test logic to enable appropriate decisions resulting from AU test conditions. With this information, the test logic generates an enable signal for the next ROM address selection. The ROM address logic uses the next address enable signal to select one of eight possible next ROM addresses, or a default address. Once selected, the next ROM address is loaded into the ROM address register on the negative-edge of the true system clock pulse and a state transfer occurs.

Of the nine possible next addresses that are generated by the ROM address logic, four are obtained from ROM outputs of the present state. The remaining five are generated with logic.

1.3.3 **ARITHMETIC UNIT MEMORY ACCESS.** The 16-bit data structure of the Arithmetic Unit has the capability of addressing 65, 536 words of random access memory. The AU requests memory cycles from the semiconductor memory system; however, the memory controller performs all memory cycle priority and control functions. Interface signals between the AU and memory controller and the generalized timing information are shown in figure 1-5.

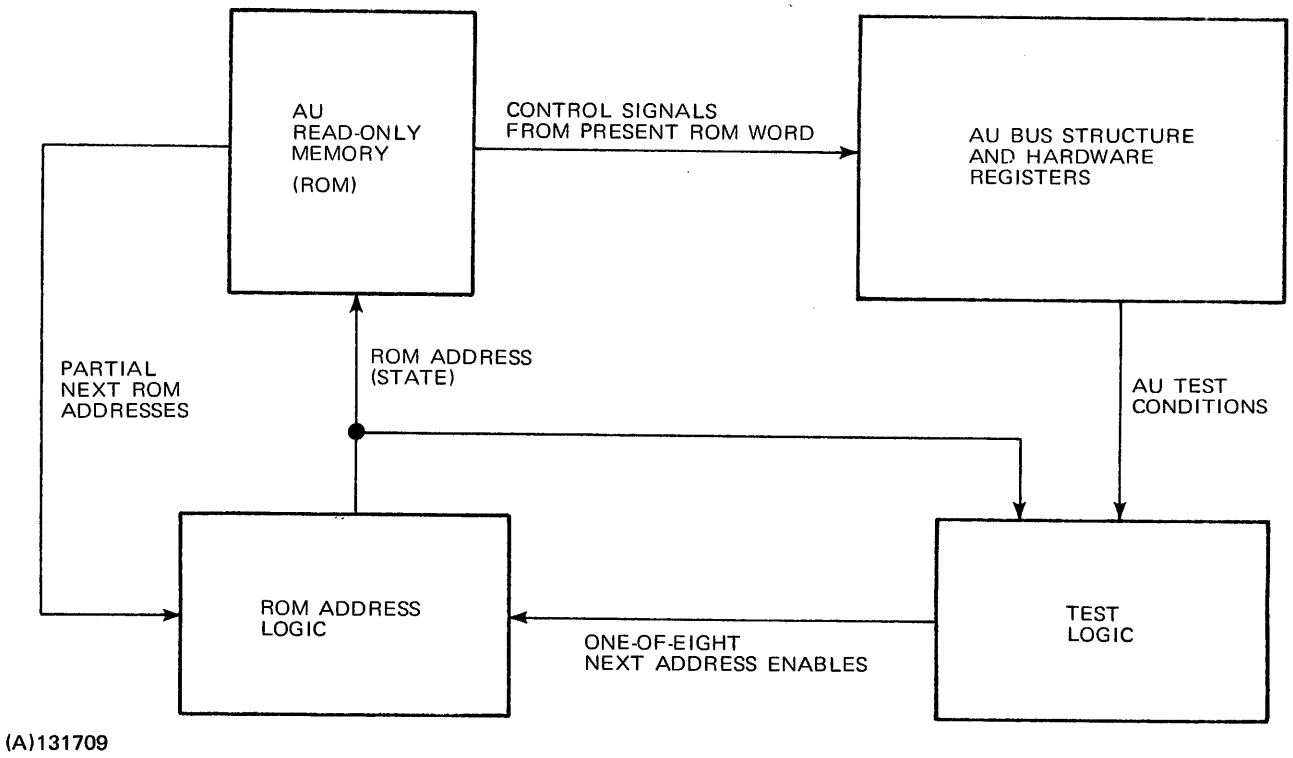


Figure 1-4. Arithmetic Unit Control Configuration, Functional Diagram

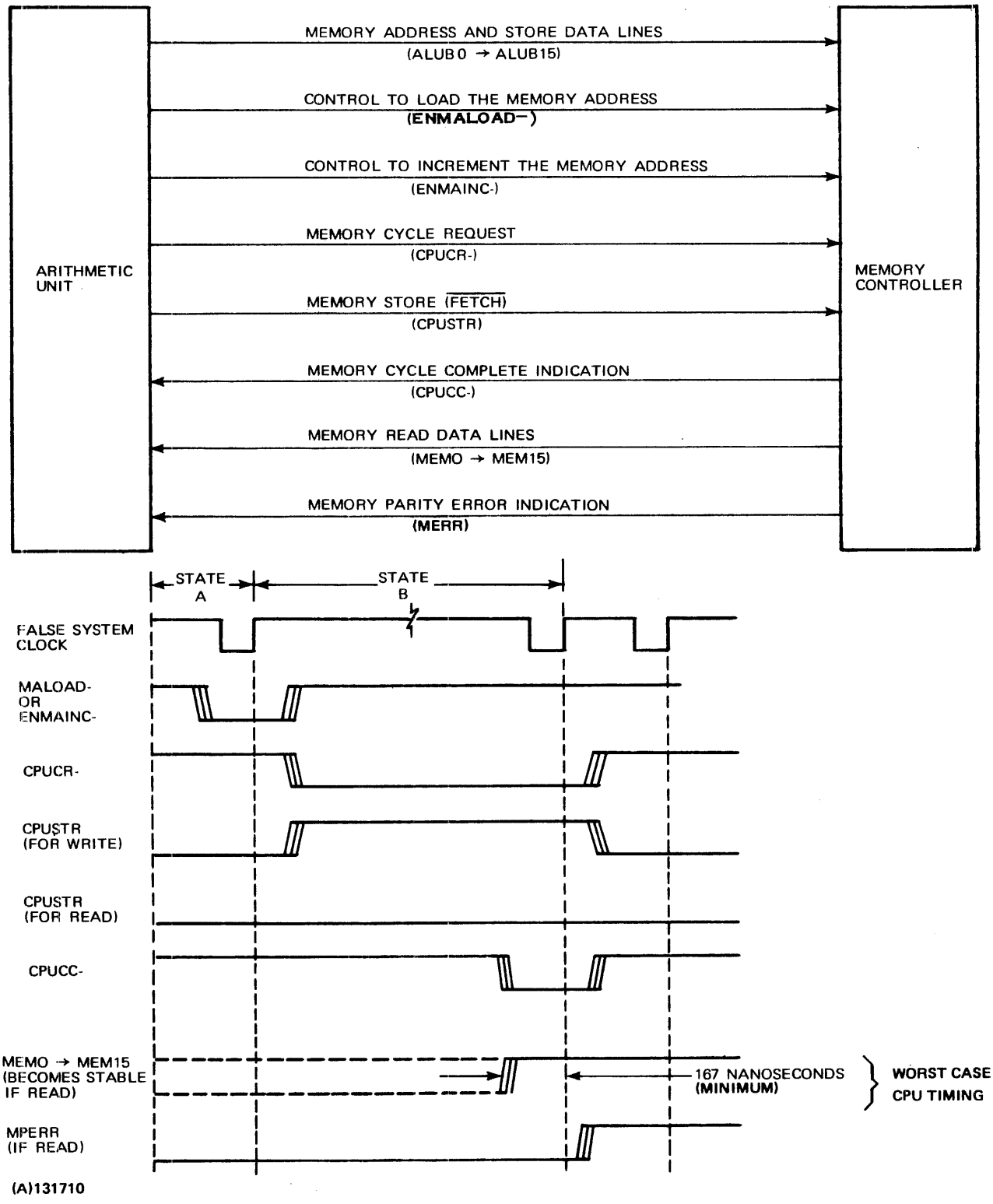


Figure 1-5. Memory Controller Interface Connections, Signals, and Timing

A 16-bit memory address register is duplicated on both the arithmetic unit and memory controller. This allows the same sixteen lines (ALUBO→ALUB15) to carry both memory addresses and write data to the memory controller board. Both registers may be parallel loaded or incremented by one in the logic state that precedes the memory cycle. Any time the AU memory address register is loaded or incremented, the register on the memory controller receives the same controls; however, the converse is not true.

Arithmetic memory cycles require two logic states. These states are illustrated in figure 1-5 as state A and state B. In state A, the memory address register is conditioned to be loaded. The ENMALOAD– signal can load the memory address register with the address which appears on the ALUB lines, or the ENMAINC– signal can increment the present address by one. Once the memory address is conditioned to be loaded, an AU memory cycle may be requested.

A typical memory cycle state occurs during state B. During state B, the AU generates CPUCR– as a cycle request to the memory controller. If a store cycle is requested, CPUSTR is generated along with CPUCR–. A fetch cycle is requested if only CPUCR– is generated. As shown in figure 1-5, an AU cycle request disables the AU system clock. This condition remains until the memory controller finishes the cycle and returns a cycle complete signal (CPUCC–) to the AU. Then the clock starts and state transfer occurs. Since the clock stops during memory cycles, AU control signals which require clocks do not effect the AU until the end of the memory cycle state.

The minimum time duration of a cycle state is 750 nanoseconds. Two system factors can extend this time. The Direct Memory Access Channel (DMAC) can be in operation, making it impossible for the AU to gain immediate access to memory. Also, the memory controller must refresh the MOS memory. This requires a 750-nanosecond cycle every 31 microseconds (every 62 microseconds on the 980A) during normal operating conditions. If either of these conditions occur, the AU remains in its cycle state until the memory controller provides access to the AU and issues a CPUCC– signal.

During a store cycle, data stored in memory is taken from the ALUB lines. If data is fetched from memory, the AU takes data from the MEM lines. Read data must be stable for 167 nanoseconds before the edge of the clock pulse occurs to meet worst case propagation delays through the AU data path. If a memory parity error is detected by the memory controller, the MERR signal indicates this condition in the state immediately following the memory cycle.

1.3.4 **DIRECT INPUT/OUTPUT.** The AU provides control and timing for the direct I/O bus by the execution of two I/O bus instructions. The basic interface signals between the AU and I/O bus system are shown in figure 1-6. Data transferred out to the I/O bus system is transferred on the IOUT lines. Input data to the AU is transferred on the DP lines.

During the execution of both the Write Direct Signal (WDS) or Read Direct Single (RDS) instructions the first word of the instruction is sent to the I/O bus as the control word along with the strobe signal GOIO–. This control word contains the coded external I/O register number which is unique to each I/O controller used. This word also contains a code bit to identify the difference between WDS and RDS instructions.

After the control word is sent, the WDS instruction sends an operand word out on the IOUT lines. It also strobes the data with TERMIO–. An RDS instruction controls the AU to accept data on the DP lines. After the data has been received, a TERMIO– strobe is sent to indicate data transfer has occurred.

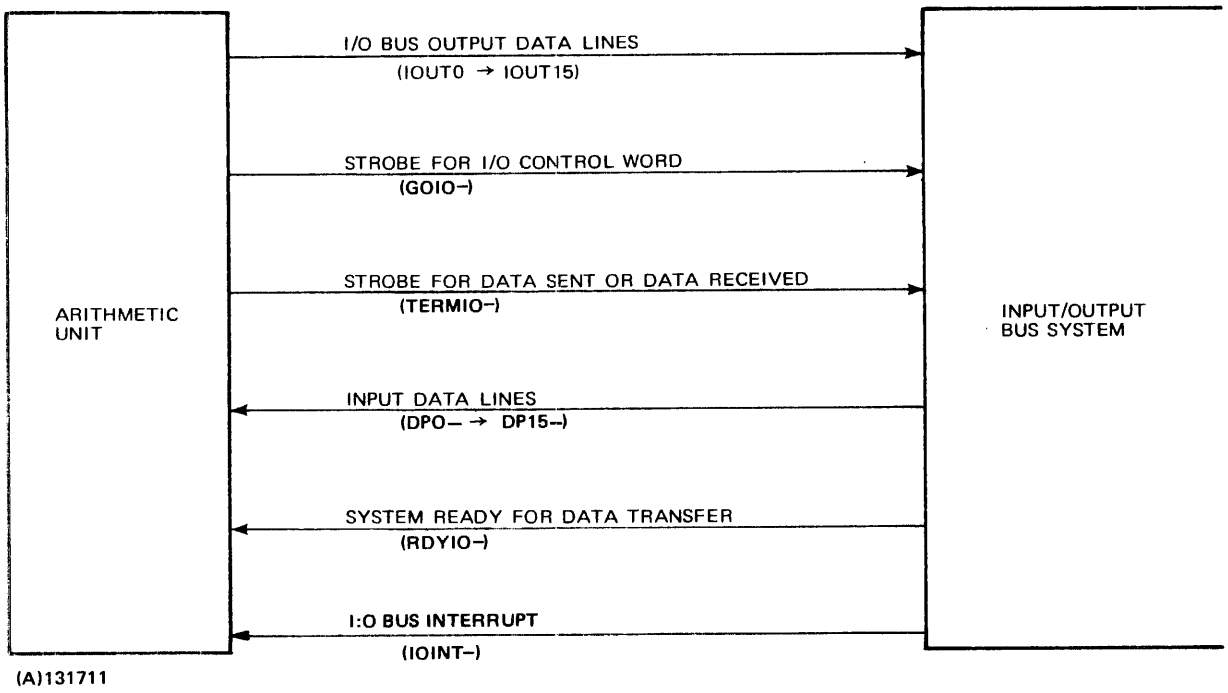


Figure 1-6. Input/Output Bus System Interface Signal Connections

If the BUSY option is used with either the RDS or WDS instructions, the AU transfers the control word and GOIO– as in all cases; however, the AU delays the transfer of data for a maximum of 750 nanoseconds. The AU must receive the RDYIO– signal during the 750 nanosecond period to initiate a data transfer. For detailed I/O bus timing, refer to the *Model 980 Computer Input/Output Manual* (TI Part Number 960694-9701).

1.3.5 **DIRECT MEMORY ACCESS CONTROL.** By executing the Automatic Transfer Instruction (ATI), the AU allows direct memory access by high speed device controllers. The AU continues program execution; however, the DMAC has a higher priority on memory cycles. Under these conditions, cycle times for the AU increase, because the AU must wait for the DMAC device to finish its cycles. The memory controller provides complete memory cycle priority control.

1.3.6 **AUXILIARY PROCESSOR PORT ACTIVATION.** The Auxiliary Processor Port (APP) is similar to the DMAC port in that the AU allows devices on both ports to have memory access. However, when the AU executes the Auxiliary Processor Instruction (API), the AU does not execute subsequent instructions until the APP device has completed its process. Auxiliary processes are treated as optional CPU instructions. When the process is complete, the AU executes the next program instruction. Also, if the APP device is not in the system, the AU takes illegal instruction operation code action as with any other illegal instruction. With respect to DMAC and refresh cycles, the APP requested memory cycles have the same access priority as AU instructions.

Because auxiliary processes can require long execution times, these processes can be aborted by the occurrence of system interrupts. However, it is a function of the APP controller to relinquish access to the system and return system control to the AU in the event of an interrupt. It is only possible for the APP device to lose control immediately if the Privileged Instruction Feature (PIF) is enabled and the APP controller attempts a memory address violation.

1.3.7 **INTERRUPT CONTROL.** There are three standard types of interrupts and a fourth optional interrupt which the AU can control. These interrupts are recognized by the AU in the following order of priority:

- Internal
- Expanded Priority Interrupt System (optional)
- Direct Memory Access Channel
- I/O Bus System.

The internal interrupt can be generated from several sources of equal priority. These sources include:

- Illegal Operation Codes
- Power Failures
- Memory Parity Errors
- Privileged Instruction Feature Memory Address Violations
- Privileged Instruction Feature Instruction Violations.

The occurrence of any or all of these conditions causes the AU to execute the next instruction from memory address 0000_{16}. To determine which condition caused the internal interrupt, status register bits are set to identify each. These status bits are as follows:

STATUS REGISTER BIT	CAUSE
5	PIF Address Violation
6	PIF Instruction Violation
14	Memory Parity Error
15	Power Failure
(NONE OF ABOVE)	Illegal Operation code

The PIF violations and memory parity errors can be disabled, by other status bits, to prevent these violations and errors from causing internal interrupts. These status bits (enable = ONE, disable = ZERO) are shown in table 1-1.

The optional expanded interrupt is generated by priority logic in the expanded interrupt system. When the AU recognizes this type of interrupt, it executes the next instruction from the location specified by the address read in from the expanded interrupt system. This type of an interrupt can occur only if status bit 8 is set.

Table 1-1. Status Register Bits

Bit	Function
0	Comparison Indicator
1	Comparison Indicator
2	Overflow Indicator
3	Carry Indicator
4	PIF Address & Instruction Violation Interrupt Enable/Disable
5	PIF Address Violation Indicator
6	PIF Instruction Violation Indicator
7	I/O Interrupt Enable/Disable
8	Optional Expanded I/O Interrupt Enable/Disable
9	PIF Lower Limit Address Bias Enable
10	Index Indicator
11	Memory Parity Error Interrupt Enable/Disable
12	DMAC Interrupt Enable/Disable
13	
14	Memory Parity Error Indicator
15	Power Failure Indicator

The DMAC can cause an interrupt if status bit 12 is set. This interrupt causes the AU to initiate the transfer of the next instruction out of memory location 0004_{16}.

The I/O bus system interrupts are enabled if status bit 7 is set. The AU initiates the transfer of the next instruction out of memory location 0006_{16} if this type of an interrupt is recognized.

The AU checks for interrupt requests after the execution of each instruction in a program. Exceptions to this rule include WDS, RDS, LSB, SSB, and register-to-register instructions which have the status register as the destination register. If any or all of the interrupts occur, the AU responds to the interrupt with the highest priority. The AU also disables all other interrupt types of the same priority level and lower.

SECTION II

INSTALLATION

2.1 GENERAL

This section describes the installation of the Arithmetic Unit (AU) and Control Console. The installation of the series 980 Computers is covered in the Model 980 (A or B) Computer Maintenance Manual, System Description referenced in the Preface to this manual.

CAUTION

Removal or replacement of the AU or Control Console logic boards require that ac power and battery power must be turned off.

An air baffle (dust cover) mounts over the CPU logic card cage. This baffle must be removed in order to gain access to the AU and Control Console modules. Generally, the CPU should not be operated prior to replacing the baffle to ensure proper cooling air flow.

2.2 ARITHMETIC UNIT INSTALLATION

The AU logic multilayer circuit boards plug into a connector plate mounted in the CPU chassis. Each logic board has two bottom-edge connectors that mate with captive connectors on the connector plate. The slot location for each board is labeled on the card cage frame (A1 or A2). The A1 board is marked with an A1 on the ejector tab and is located in slot A1. Similarly, the A2 board is located in slot A2. The connector numbers associated with slots A1 and A2 differ between the 980A and 980B models. This is shown in figure 2-1. Modules are installed with components facing the front of the machine.

2.2.1 980A CONNECTOR PLATE. Slots A1 and A2 are designated XA7 and XA8, respectively, on the 980A Computer connector plate. Connectors XA7P1 and XA7P2 receive the A1 logic board edge connectors. Slot A2 connectors XA8P1 and XA8P2 receive the A2 logic board edge connectors.

2.2.2 980B CONNECTOR PLATE. Slots A1 and A2 are designated XA6 and XA7, respectively, on the 980B computer connector plate. Connectors XA6P1 and XA6P2 receive the A1 logic board edge connector. Slot A2 connectors XA7P1 and XA7P2 receive the A2 logic board edge connectors.

2.3 AU OPERATION VERIFICATION

After performance of the AU installation procedure the operation of the AU should be verified by executing the applicable CPU Performance Assurance Test (PAT).

2.4 CONTROL CONSOLE INSTALLATION

The Control Console assembly plugs into the slot labeled CP (stamped on the card cage frame). The symbolized panel, with LED displays and switches, faces the front of the computer. The trim frame is aligned to a cutout in the chassis front panel when the console is properly installed.

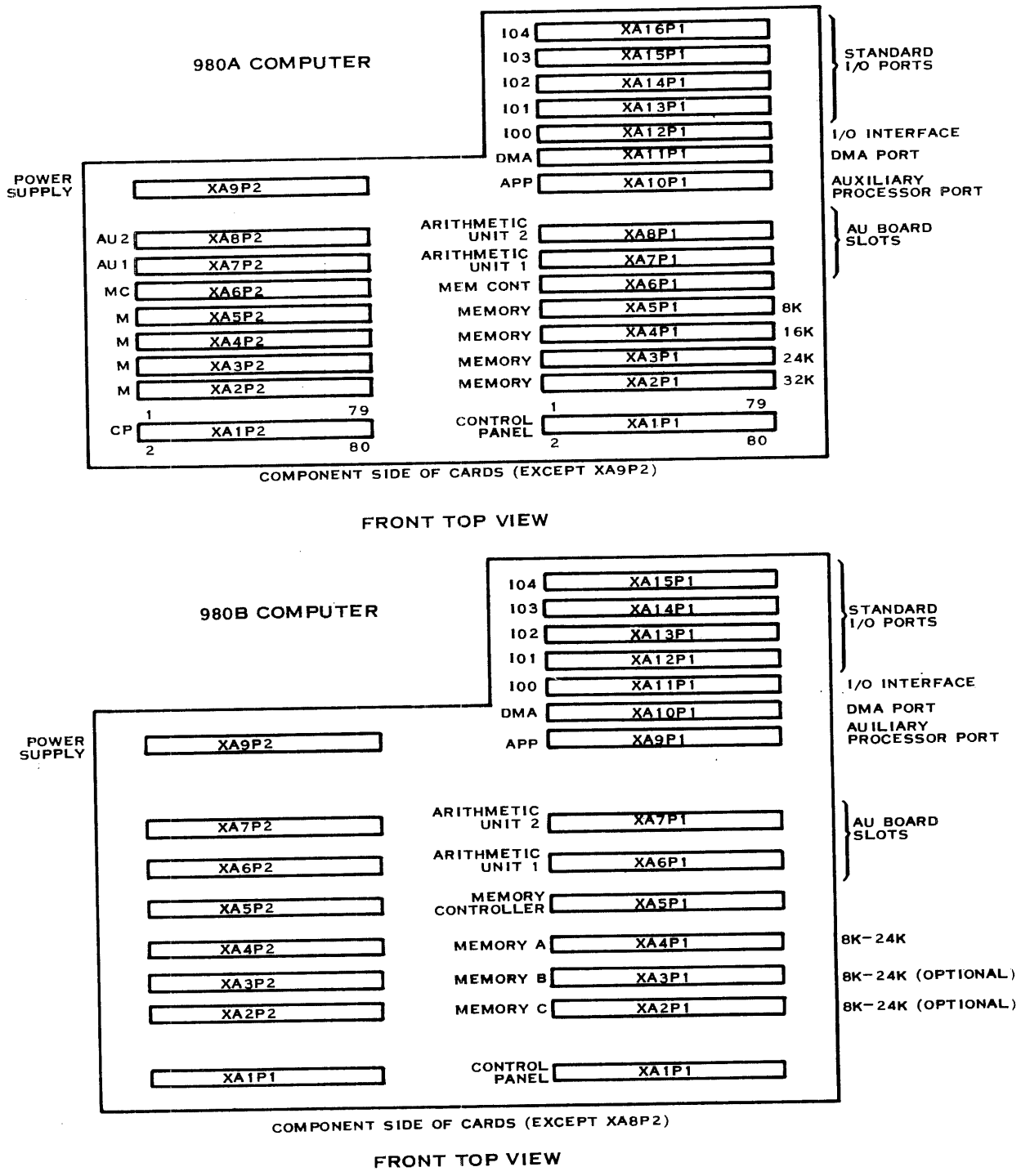


Figure 2-1. Arithmetic Unit Printed-Circuit Card Locations

SECTION III

OPERATION AND PROGRAMMING

3.1 GENERAL

The Control Console provides m lnual control of AU operations, display of system data, and monitoring of machine status. This section describes the functions of the switches and indicators.

3.2 CONSOLE SWITCHES

The Control Console contains logic, switches, and indicators for manual control of the AU. The toggle switches, on the control panel, are arranged in three rows. The top row contains 16 data entry switches; the center row provides control of data display and entry; the lower row provides control of system operation. Figure 3-1 shows the Control Console switches and indicators.

3.2.1 **CONSOLE KEY LOCK SWITCH (LOCK/UNLOCK).** The Console Key Lock Switch enables operation of the Control Console in the UNLOCK position. When the switch is in the LOCK position, all controls are disabled except the 16 data switches, the 4 sense switches, and the Breakpoint/Clear (BKPT/CLR) switch. The contents of the DATA switches cannot be entered when the Key Lock Switch is in the LOCK position, but the breakpoint logic continues to monitor the memory address specified by the data switches. The BKPT/CLR resets the BKPT indicator; however, the system cannot be forced to halt. The key can be removed in either position.

3.2.2 **DATA SWITCHES.** The Data Switches are numbered 0 through 15, left to right. When a switch is up, the corresponding bit position is a logic one; it is a logic zero when the switch is down. These Data Switches allow the selection of data that can be placed in the following locations: the program address register (PC), the machine status register (ST), the memory address register (MA), directly into memory (MD), and registers A, E, X, M, S, L, or B. Data loading occurs in the specified location when the corresponding Enter Switch is actuated to the ENTER position, providing that the AU is in the Halt mode. When the AU is running the Data Switches contain the address being monitored by the breakpoint logic.

3.2.3 **IR SWITCH.** The IR switch is used to display instruction register data on the indicators. The switch has a normal center position, a momentary up position, and a momentary down position. When pushed to the DISPLAY (down) position, the contents of the instruction register are displayed. The up position of the switch is nonfunctional. The display function is only active when the computer is in the Halt Mode.

3.2.4 **PC, A, B, E, L, M, S, X, ST, MA AND MD DISPLAY AND ENTER SWITCHES.** These switches control the following data DISPLAY and ENTER functions, respectively: Program Counter; Registers A, B, E, L, M, S, X and ST; Memory Address register; and Memory Data. Each switch has a normal center position, a momentary up position, and a momentary down position. When any of these switches is pushed to the ENTER (up) position, the current setting of the 16 DATA switches is loaded into the associated register or memory location. The entered value is displayed by the display indicators. The Memory Data (MD) is entered into the address specified by the contents of the Memory Address (MA) register. If the switch is pushed to the DISPLAY (down) position, the contents of the associated register or memory are displayed. All switches in this group only function when the computer is in the Halt Mode.

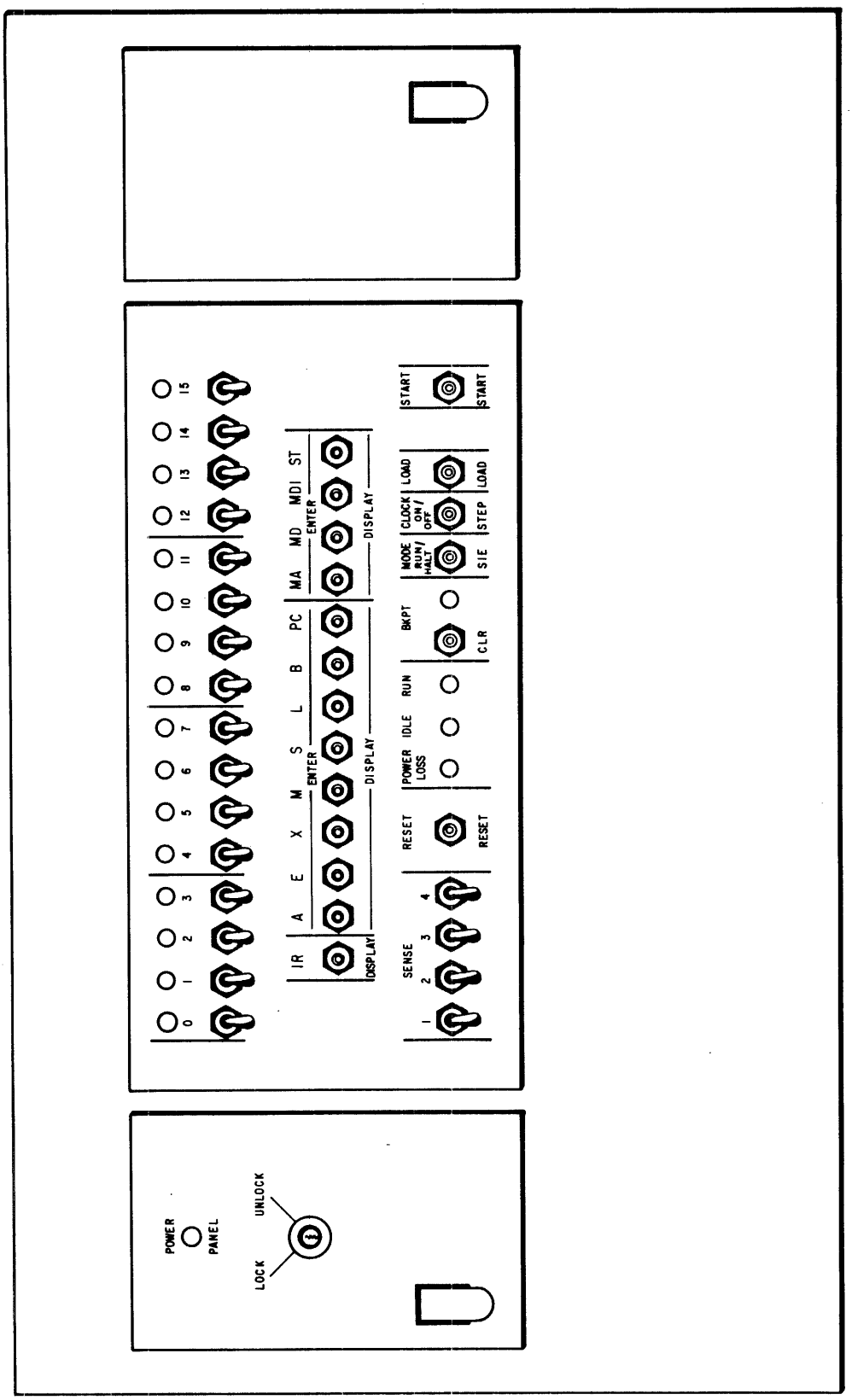


Figure 3-1. Control Console Panel Switches and Indicators

3.2.5 **MDI SWITCH.** The Memory Data and Increment Address (MDI) switch functions as the MD switch that is described in the preceding paragraph for entering or displaying memory data except each time MDI is actuated to either ENTER or DISPLAY, the Memory Address Register is incremented. It is used for loading or displaying consecutive memory locations.

NOTE

If the MDI switch is activated to read location MEM the MA register contains address MEM+1 while the Data Indicators display the contents of MEM.

3.2.6 **SENSE SWITCHES.** The four SENSE switches are two-position toggle switches used by the two sense switch instructions (SSE and SSN). Each SENSE switch is placed in the up position for a logic one or down for a logic zero.

3.2.7 **RESET SWITCH.** The system RESET switch is used to reset major system registers. This switch has a normal center position and a momentary down position. When the switch is pushed to the RESET (down) position, the program counter, status register, memory address register, and display register are cleared. This function can occur if the computer is in the Halt or Run mode, but it is disabled when the panel is locked.

3.2.8 **BKPT SWITCH.** The Breakpoint (BKPT) switch is a three-position toggle switch. If the computer is in the Run or SIE mode and the BKPT switch is in the center position, the BKPT indicator is lighted when the data switch setting is contained in the memory address register. The indicator can be extinguished by pushing the BKPT switch to the CLR (down) position (computer in any mode) without affecting system operation. When the BKPT switch is in the BKPT (up) position and the computer is in the Run mode, the computer Halts and the BKPT indicator lights if the DATA switch setting is contained in the memory address register. The instruction which uses the breakpoint address is completed before the computer Halts. When the panel is locked the Halt cannot occur.

3.2.9 **MODE SWITCH.** The MODE control switch is a three position toggle switch. To start program execution, the MODE switch is placed in the up (RUN) position and the START switch is actuated (placed in momentary down position). The Run mode is entered and the RUN indicator is lighted when the START switch is actuated. When the MODE switch is placed in the center (HALT) position, the system halts when the instruction in progress is completed. To execute a single instruction, the MODE switch is placed in the down (SIE) position and the START switch is actuated.

3.2.10 **CLOCK SWITCH.** The Clock switch is a three-position toggle switch with a normal center position (OFF) an up position (ON) and a momentary down position (STEP). For normal system operation, this switch is in the up (ON) position and the system clock is free running. When the Clock switch is placed in the center (OFF) position, the arithmetic unit clock is stopped. Each time the Clock switch is pushed to the momentary down (STEP) position, a single system clock pulse is generated for the arithmetic unit. The operation of this switch is normally disabled. Refer to the maintenance section for instructions on enabling the Clock Switch using a jumper on the Console printed circuit board.

3.2.11 **LOAD SWITCH.** The LOAD (store bootstrap loaders in memory) switch is a two-position toggle switch with a normal center position and momentary up position. If the system is halted and this switch is pushed to the up position, 256 words of optional firmware program are loaded into memory. The memory location of the first word is always zero.

If the MODE switch is in the RUN position, the computer begins execution at the address in the program counter. If the MODE switch is in the HALT position the computer does not begin execution until the MODE switch is placed in the RUN position and the START switch is actuated (momentary down position).

3.2.12 **START SWITCH.** The Start switch is a two-position toggle switch with a normal center position and momentary down position. If the system is in the Halt mode and this switch is pushed to the down (START) position, the RUN or SIE setting of the MODE switch is enabled.

3.3 CONSOLE DISPLAY INDICATORS

Console indicators display data and machine status. The top row of indicators (DATA) display 16 data bits, 0 through 15, left to right. The remaining four indicators display machine status (e.g., PWR LOSS, IDLE, RUN, and BKPT).

3.3.1 **DATA INDICATORS.** Sixteen Data indicators at the top of the console display machine data. The contents of a memory location or of any one of eleven machine registers can be displayed when the system is in the Halt mode. Data can be entered manually into memory or into any of ten machine registers. When a manual entry is executed, the value is displayed. When the system is in the Run or SIE (Single Instruction Execution) mode, the present instruction address is continually displayed.

3.3.2 **POWER LOSS INDICATOR.** The Power Loss (memory power failure) indicator is on when memory power is lost and ac power is restored. The Power Loss indicator is turned off by activating the System RESET switch.

3.3.3 **IDLE INDICATOR.** The Idle indicator is lighted, and the Run indicator is extinguished, when an Idle instruction causes the computer to halt. The Idle indicator is then extinguished when the computer enters the Run or SIE mode or by system RESET.

3.3.4 **RUN INDICATOR.** The Run indicator is lighted when the system is placed in the Run mode and the START switch is activated. When the system is removed from the Run mode, the Run Indicator is extinguished.

3.3.5 **BKPT INDICATOR.** The BKPT (Breakpoint) Indicator is lighted when the memory address register contents are the same as the data switches and the computer is in the Run or SIE mode. The BKPT indicator is extinguished by setting the BKPT switch to the CLR position.

3.4 MEMORY INITIALIZATION

Memory initialization is performed following power turn on if memory refresh has been lost. Memory loss is signified when the PWR LOSS indicator is lighted. The memory initialization routine writes into every memory location to eliminate parity errors caused by the power loss. Interrupts are masked during initialization to prevent internal interrupt traps.

The routine to initialize memory is loaded into memory by the ROM bootstrap logic from the Control Panel ROM. The following procedure loads and executes the initialize routine.

- Depress RESET switch
- Set PC = $000F_{16}$, the entry point for the initialization routine
- Set the Mode switch to the RUN position (upward)

- Activate the LOAD switch to the momentary up position
- Initialization is now complete. An idle instruction is placed in each location by the routine ($CE00_{16}$).

3.5 ROM BOOTSTRAP OPERATION

The ROM bootstrap loader moves 256 words of Read-Only memory (ROM) into CPU memory starting at memory location 0000_{16}. The bootstrap library contains basic routines used to initiate data input operations from the standard input peripherals. The input data from the peripherals usually is a relocating loader for control of system or diagnostic entry.

3.5.1 BOOTSTRAP INDEX. The first 16 locations of memory, following the bootstrap library, contain a jump for each standard peripheral device. The selected jump instruction transfers program control to the desired bootstrap program. The list below associates the memory locations with their device routines.

Memory Location	Device
0	Model 31, 33, or 44 Diablo Moving-head disc
1	DDC Fixed-head disc
2	800/1600 Magnetic tape
3	Card Reader
4	High-speed paper-tape reader
5	Teletype reader
6	ASR 733 and Teletype reader with communication module interface
7	DS330 Disc system*
8 to E	Left for further expansion
F	Store idles to all of memory

*DS330 is present only in the special ROM shipped with systems shipped with a DS330 Disc System.

The selected bootstrap program is relocated by the program to end at location 87_{16} and it enters data from the device beginning at location 88_{16}. The program expects the length to be the 12th input word that is read into location 93_{16}.

3.5.2 BOOTSTRAP OPERATING PROCEDURE. The following procedure enters the bootstrap, starts bootstrap execution and initiates the data input from the selected peripheral device.

- Ready the (loader) program media in the desired I/O device
- Reset the computer
- Select the Run mode
- Put the memory location assigned to the desired bootstrap in the PC (e.g., PC=0005_{16} for teletypewriter)
- Toggle the Load switch (up)
- After the program media is loaded, the machine enters the idle mode.

3.6 CONTROL CONSOLE CHARACTERISTICS

The Control Console assembly consists of a printed-circuit board assembly, a panel symbolized with control and indicator identification and a trim frame. All console electrical components, switches, and indicators are mounted on the printed circuit board which has two 80-pin, bottom-edge, connectors. The entire console assembly is installed in card connectors XA1P1 and XA1P2. Except for +5 volts and ground, all system interconnections are completed using wire wrap connections on the back panel. The trim frame is aligned to a cutout in the chassis front panel when the control console assembly is properly installed.

SECTION IV

THEORY OF OPERATION

4.1 POWER SUPPLY MASTER RESET AND POWER OFF

The Central Processor Unit (CPU) power supply outputs two signals to the system to indicate the condition of system power.

Signal timing is shown in figure 4-1. The times that are listed in this figure are approximate; because, times are affected by factors such as ac line voltage and power supply loading. However, the sequence of signal transitions always occurs as shown in the diagram.

The master reset signal (MRESET–) is switched to ground when any regulated supply voltage is below tolerance. When system ac power is turned on, the supply voltages reach the regulated level in approximately 600 milliseconds. The MRESET– circuit opens 2 to 3 milliseconds after the +5 Volt supply starts to rise from zero volt. When the circuit opens, all voltages reach regulated levels. When high, the MRESET– signal is connected through a 100 ohm resistor to +5 Volts.

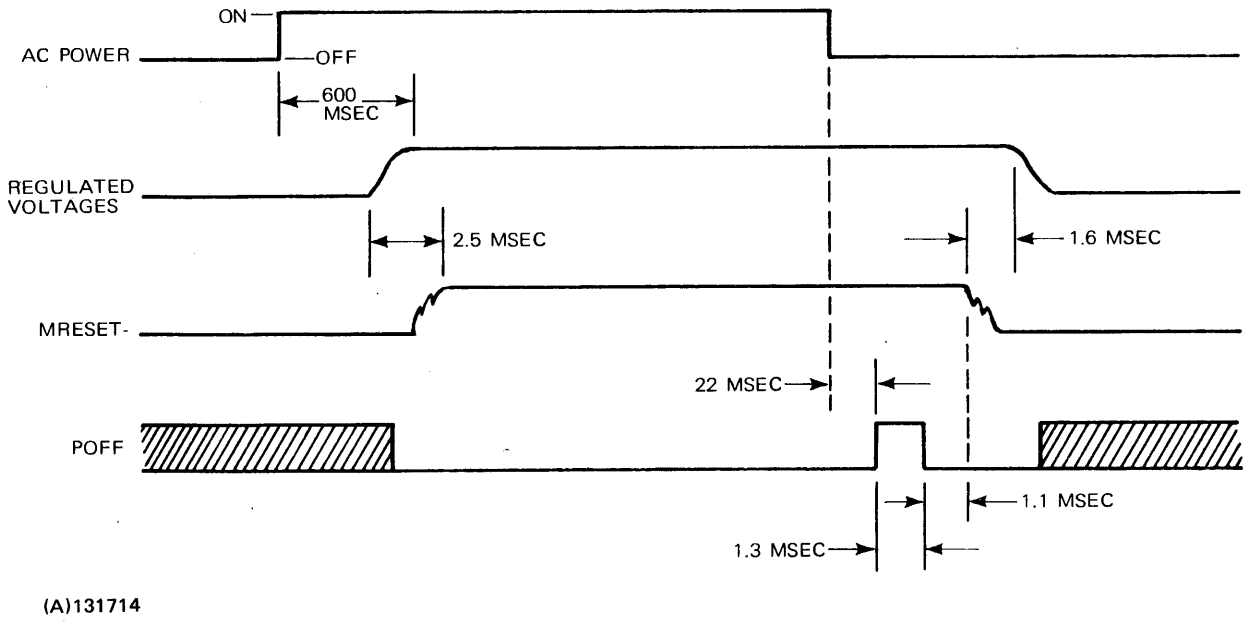


Figure 4-1. MRESET– and POFF Signal Timing

NOTE

The transitions in MRESET– contain noise caused by relay contact bounce in the 980A computer power supply. This noise is removed by filters in the AU where MRESET– is used. A transistor switch replaces the relay in the 980B power supply.

When ac power is turned OFF, MRESET remains high until approximately 16 milliseconds before the regulated supply voltages begin to fall.

The second status signal from the power supply is power off (POFF). The POFF signal is generated when ac power is turned OFF, or when an ac power interruption of sufficient duration occurs to cause loss of regulation. The POFF signal is generated by logic circuits; therefore, the state of POFF is undefined until the supply voltages approach regulated levels. At ac power turn on, POFF settles to a logic ZERO level before MRESET– goes high. When ac power is interrupted, POFF goes to a logic ONE level within 20 to 24 milliseconds after the interruption occurs. The POFF signal remains at a logic ONE level for approximately 1.3 milliseconds and returns to a ZERO level approximately 1.1 milliseconds before MRESET– goes low. The leading edge of the POFF signal causes an internal interrupt in the AU and the trailing edge halts program execution by clearing the run flip-flop. The 1.3 millisecond duration of POFF provides sufficient time to execute an interrupt service routine that is designed to prepare the system for the imminent power failure.

4.2 AU CLOCK SOURCE AND DISTRIBUTION

4.2.1 CLOCK SOURCE SIGNALS. The memory controller board in the CPU generates two separate clock signals for use by the computer system. (Figure 4-2 shows the clock control and clock distribution logic.) The first and most widely used clock signal by the AU is called GCLK–. The GCLK– signal is high for 167 nanoseconds and low for 83 nanoseconds to consume a total period of 250 nanoseconds. This GCLK– clock signal can be interrupted by several conditions. When interrupted, GCLK– remains in a high state as long as these conditions exist.

A STPCK– signal is generated by the AU and is sent to the memory controller as an indication that GCLK– must be stopped. The STPCK– signal stops GCLK– if any one of three conditions exist. The first condition exists when the AU or API has requested a memory cycle with CPUCR– and the memory controller has not sent back the cycle complete signal, CPUCC–. The second condition exists when the AU is in the Step clock mode. When the console signal CLKON is low and the panel is unlocked, GCLK– is stopped. In this state, a single clock pulse is enabled when the momentary STEP position of the CLOCK switch is activated on the console. The signal STEPULS–, which is a synchronous 250-nanosecond pulse, is generated by the control console enabling this clock pulse. A third input signal is available to permit special user conditions to stop GCLK–. The signal EXTSTCK– is available at the bottom edge connector of AU-1 for special use.

The second system clock source signal is SYSCLK–. This clock signal has the same waveform as GCLK–; however, SYSCLK– is used for system logic which cannot have clock interruptions for the same conditions which interrupt GCLK–.

The clock distribution logic minimizes skew by assuring that the clock pulse presented to each flip-flop is delayed by the same number of gate propagations from the primary system clock GCLK–.

4.2.2 **BUFFERED AU CLOCK SIGNALS.** As shown in figure 4-2, there are three types of buffered clock signals used on AU-2. All three of these clock signals are derived from GCLK–. The first type (GCLKA:1, 2, and 3) is generated for use with NAND gates which enable False clocks to trigger positive edge triggered devices. The second type (GCLKB:1 and 2.) is a True clock that is used to trigger negative-edge triggered devices which do not have enable gates. The third type (GCLKC:1-2- and 3-) is a False clock that is used to trigger, positive edge, triggered devices which also do not have enable gates.

The AU-1 logic board uses two types of buffered clocks that are derived from GCLK–. The first type (GCLKE:1 and 2) is used for the same functions as GCLKA:1 and GCLKB:1. The second type is used for the same functions as GCLKC:1. The SYSCLK– signal has limited use on AU-1. The SYSCLK– signal buffering is shown in figure 4-3 as FCLKT and FCLKT–. Signal IOCLK– is buffered SYSCLK– conditioned by STPIOCLR– and distribution to the I/O bus system.

4.3 RUN FLIP-FLOP

4.3.1 **FUNCTION.** The RUN flip-flop acts as a flag that indicates to the AU control logic whether to execute the next instruction in a program or to halt the computer. While the AU is executing instructions, the control logic checks the RUN flip-flop (RUNFFQ) before each instruction is executed. If the RUN flip-flop is set, the AU continues execution. If the RUN flip-flop is reset during an instruction, the AU finishes executing that instruction and then halts. If the AU is in the Halt mode, the control logic also checks the RUN flip-flop. If the RUN flip-flop is set, the AU enters the Run mode and initiates instruction execution. The state of the RUN flip-flop is also used to generate the RUN indicator signal (RUNIND) for the control console. A logic drawing of the RUN flip-flop with associated set and reset logic is shown in figure 4-4.

4.3.2 **SET CONDITIONS.** There are five system conditions which set the RUN flip-flop. The first set condition occurs each time system ac power is turned on, and the power supply master reset signal (MRESET–) goes to a logic Zero. The MRESET– is synchronized and sensed by the RUN flip-flop steering logic as MRESETQ. This condition causes the AU to initially come up in the Run mode when ac power is applied. However, if the MODE switch on the front panel is in the HALT position, the run condition is not maintained.

The second set condition occurs when the front panel bootstrap loader function is activated. State $E1_{16}$ is used in the microsequence to transfer the ROM loader programs into memory. This state is decoded as ME and L1 to set the RUN flip-flop. This function is accomplished to permit the AU to enter the Run mode and execute the loader programs after these programs have been entered into memory.

The third set condition exists when the AU is in the halt mode due to IDL instruction execution and an enabled interrupt is requested. When these conditions exist, the AU can be removed from the idle condition (Halt mode) becuase of an interrupt.

The fourth and fifth set conditions are accomplished by use of the control console. If the AU is in the halt mode (either idle or normal halt) and the PANEL switch is in the UNLOCK position, the run mode may be entered by use of the front panel MODE and START switches. If either the RUN or SIE (single instruction execution) position of the MODE switch is selected and the momentary START switch is activated, the RUN flip-flop is set.

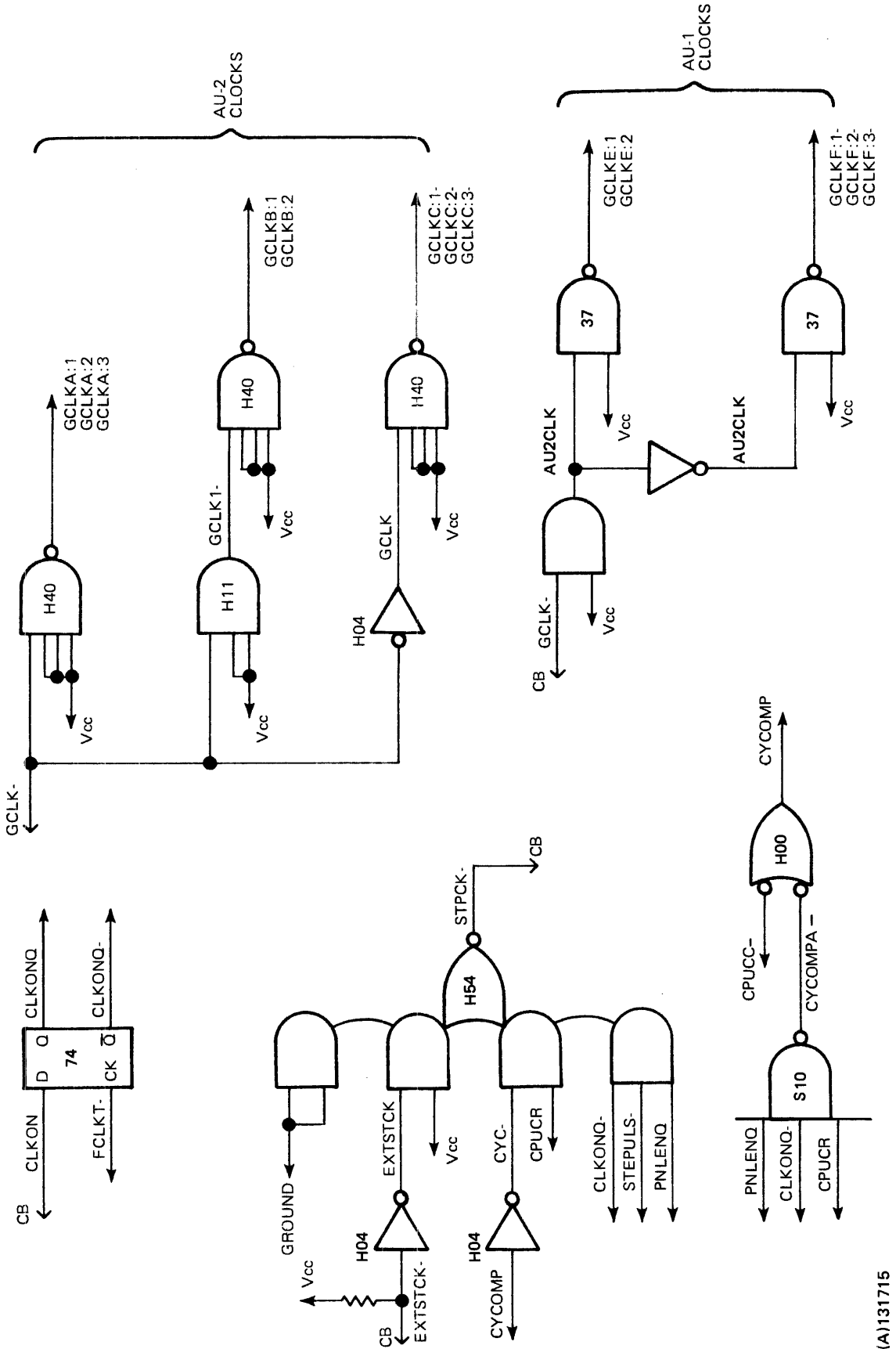


Figure 4-2. Clock Control and Distribution Logic

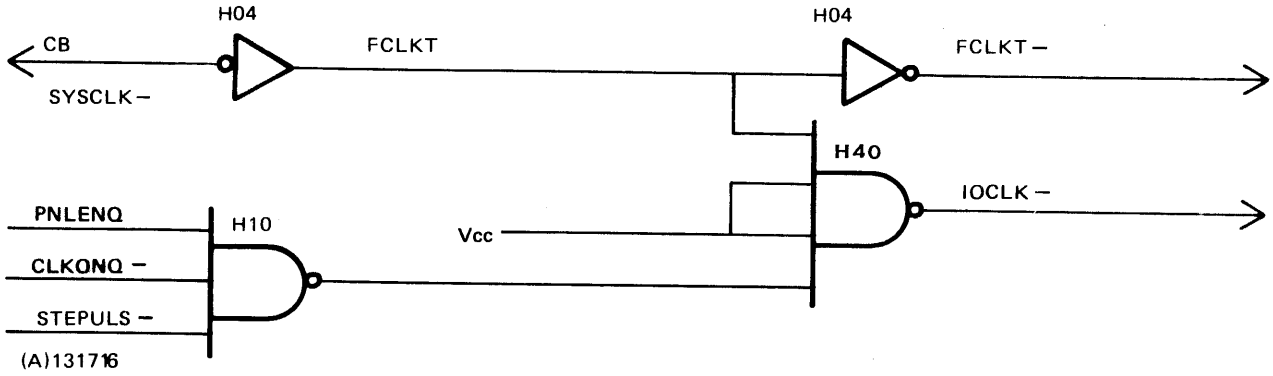


Figure 4-3. SYSCLK– Distribution

4.3.3 **RESET CONDITIONS.** There are seven system conditions which reset the RUN flip-flop. The first reset condition occurs each time power is turned off. A power supply power failure signal (POFF) interrupts the AU when power is about to be lost. The trailing edge of this power failure signal is sensed to reset the RUN flip-flop. This action halts the computer. All power failure interrupts must be processed before this action occurs.

The second reset condition occurs when an illegal instruction is detected in the internal interrupt trap location.

The third reset condition occurs when a parity error is detected in the internal interrupt trap location. Also, if an interrupt trap occurs (i.e., because of an address violation or instruction violation) and the previous memory read detected a memory parity error, then the RUN flip-flop is reset.

For the second and third reset conditions, when the AU has sequenced through the interrupt trap state (FF_{16}), the trap address is for internal interrupts (0002_{16}), and an illegal instruction or memory parity error is detected, the RUN flip-flop is reset.

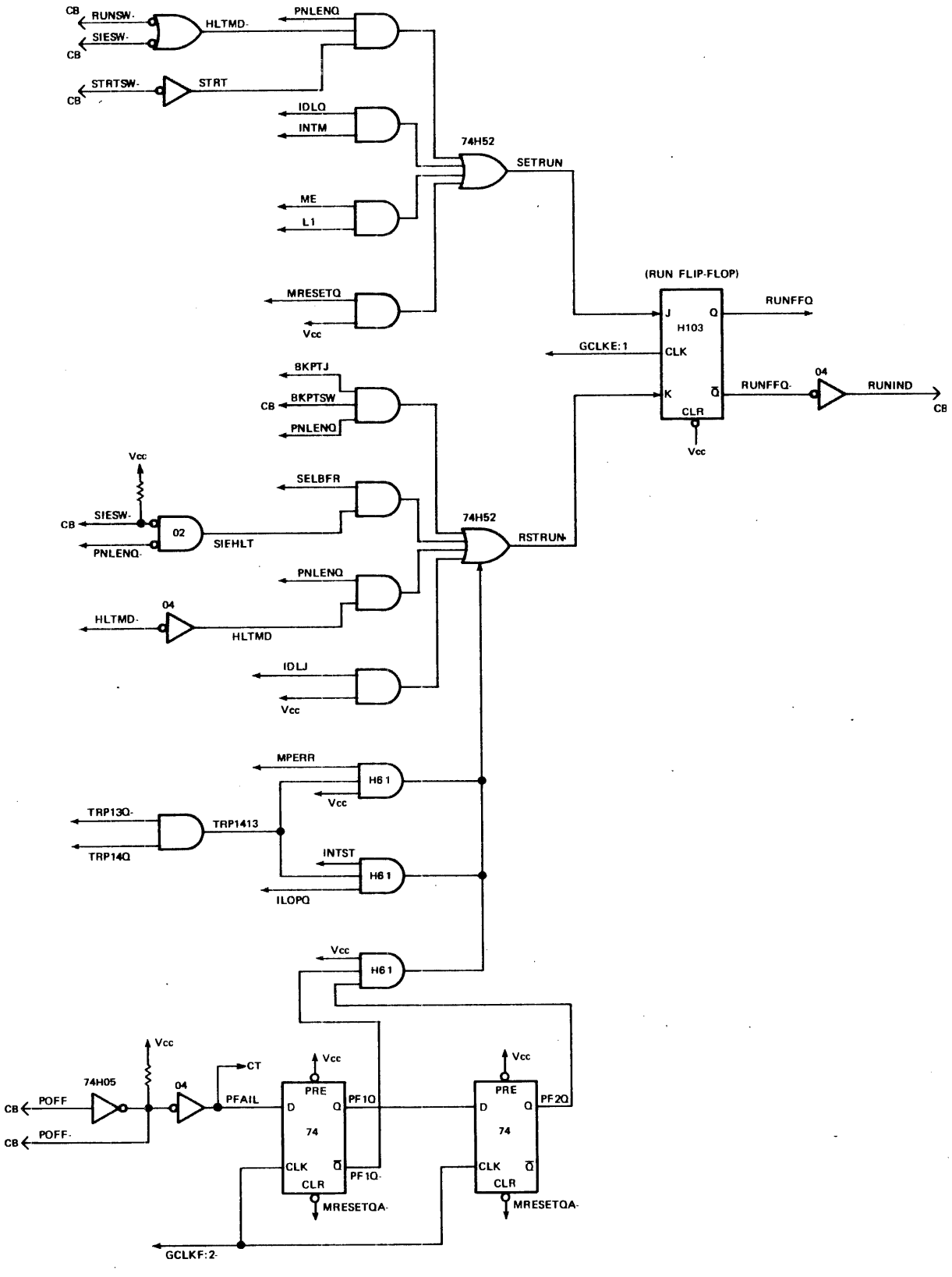
The fourth reset condition occurs when an IDL instruction is executed by the AU. The RUN flip-flop is reset when the AU sequences through the IDL state.

The fifth reset condition occurs when the control console MODE switch is in the HALT (center) position and the PANEL switch is in the UNLOCK position.

The sixth reset condition occurs when the AU is in the SIE mode and instruction execution starts. If the SIE position of the MODE switch is selected with the panel enabled, the RUN flip-flop is reset when the SELBFR signal is generated at the beginning of instruction execution. The machine halts after executing the single instruction.

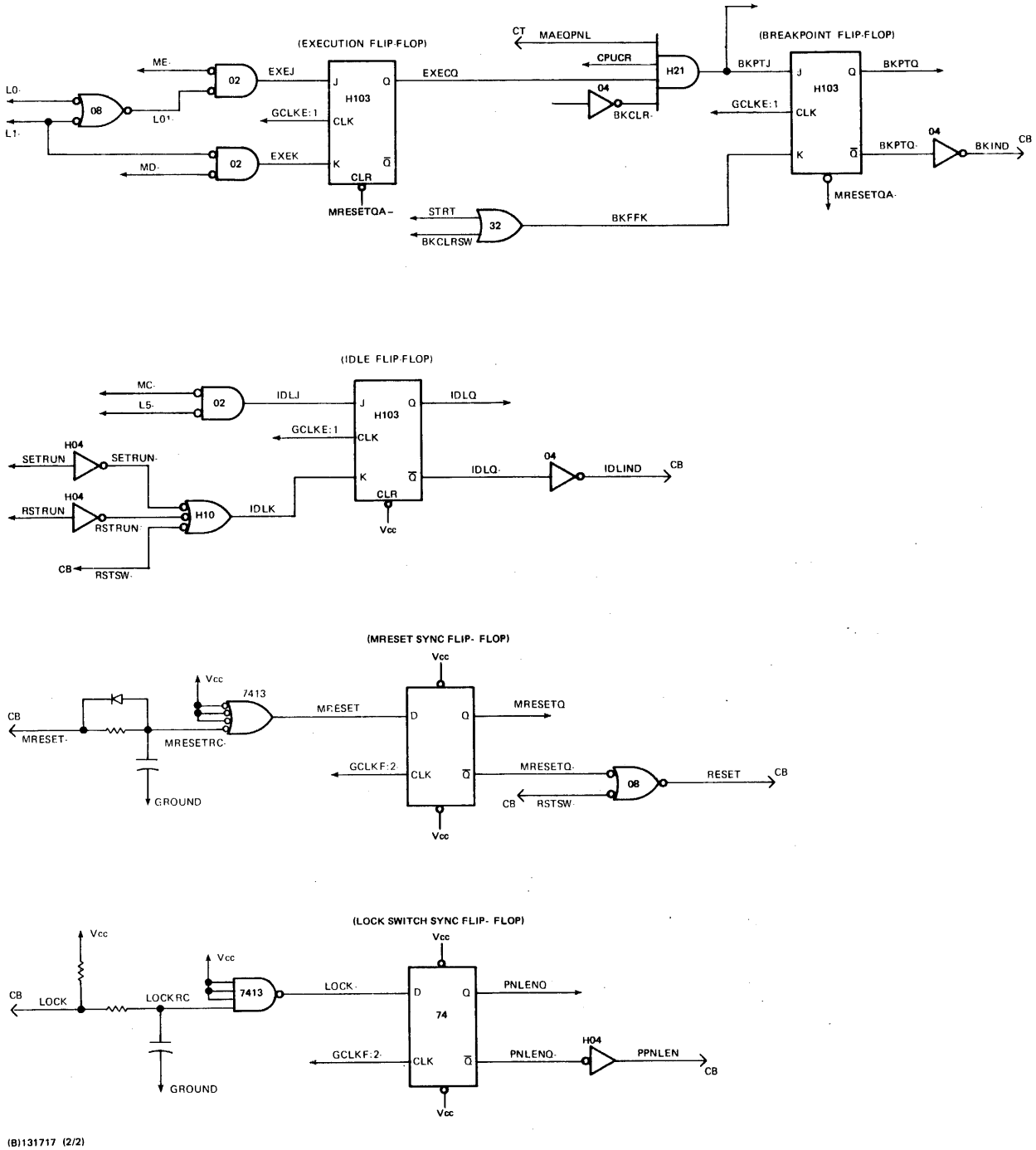
The seventh reset condition occurs when the breakpoint switch is in the BKPT position. When the BKPT switch is set and the memory address register content is equal to the panel switch settings during an instruction memory cycle, the RUN flip-flop is reset and the machine halts if the panel is unlocked.

4.3.4 **ASSOCIATED LOGIC.** The breakpoint flip-flop is controlled by control console and AU breakpoint conditions. The panel switch settings (PNL0→15) and AU memory address contents (MAR0–15) are compared by four 4-bit comparators (SN7485's) on the AU-2 board. When these data words are equal, the signal MAEQPNL is generated and sent to the breakpoint logic on AU-1.



(B)131717 (1/2)

Figure 4-4. Run Flip-Flop Logic Diagram (Sheet 1 of 2)



(B)131717 (2/2)

Figure 4-4. Run Flip-Flop Logic Diagram (Sheet 2 of 2)

4.4 BUS STRUCTURE AND REGISTERS

The Arithmetic Unit (AU) functional diagram with data paths and registers is illustrated in figure 4-5. The data path and registers are primarily implemented on the AU-2 board. The AU has three major data selectors or data buses. Two of the buses (data bus A and data bus B) supply 16-bit data to the Arithmetic Logic Unit (ALU). The ALU performs arithmetic or logical processing of data on data bus A and data bus B. The third bus (data bus C) allows the output bits of the ALU to be shifted or interchanged when processing shift, rotate, or byte instructions. The data bus C lines are returned to the necessary registers as illustrated in figure 4-5.

4.4.1 DATA BUS A. Data bus A is a 16 to 1 data bus that is implemented with two sets of 8 to 1 selectors (SN74151). The respective bit outputs of each pair of 8 to 1 selectors (A1 and A2), are OR'd as illustrated in figure 4-5. The selectors are controlled by bus A select logic that supplies select address lines to the selectors using ROM output bits. Table 4-1 lists the selector output for each select address. The A1 selectors permit portions of the inputs to be enabled to the bus (i.e., A1 output can be the most significant half (MSH) of selected input (bits 0–7) or least significant half (LSH) of selected input (bits 8–15). The A1 selectors can also be all enabled or inhibited. The A2 selectors are either enabled or inhibited.

Several data bus A selector inputs do not originate at the registers. The bus A selector input to pin number 9 (DP0–→DP15–) is the input data to the CPU from the I/O Bus. The bus A selector input A (SETBIT0–→SETBIT15–) originates in the set bit logic and is utilized during the set bit instruction. Trap address logic provides a trap address for interrupts to bus A selector input B. Bus A selector input C (PNL0→PNL15) originates at the control console data switches and is used to manually enter data into the CPU. Bus A selector input D (MEM0→MEM15) is the memory data output lines from the memory controller.

4.4.2 DATA BUS B. Data bus B is a 4 to 1 data bus that is implemented with 4 to 1 selectors (SN74153). Bus B select logic derives select addresses from ROM output bits. Table 4-2 lists the data selected for each bus B select address.

4.4.3 DATA BUS C. Data bus C is a 4 to 1 data bus that is implemented with 4 to 1 selectors (SN74153). Bus C select logic derives select addresses from ROM output bits. Table 4-3 lists the data selected for each bus C select address. The bus C logic allows the ALU outputs to be bit shifted or byte exchanged.

4.4.4 OTHER SELECTORS. Two remaining selectors are also shown in figure 4-5. An I/O output data bus selector allows the Program Counter (PC) or the ALU output to be selected to the console and I/O bus. The control console and I/O device controllers receive data from the CPU on this bus.

A second selector allows the memory controller to receive memory input data from the CPU or Auxiliary Processor Port (APP).

4.4.5 ARITHMETIC LOGIC UNIT. The Arithmetic Logic Unit (ALU) performs arithmetic and logic processing of data from bus A and bus B. The ALU consists of 4-bit arithmetic units (SN74S181) and look-ahead carry generators (SN74182). The ALU control signals in figure 4-5 are ROM generated. Functions for various ALU control signal combinations are listed in table 4-4. The ALU outputs are bus C selector inputs and shift signals which allow bit shifts.

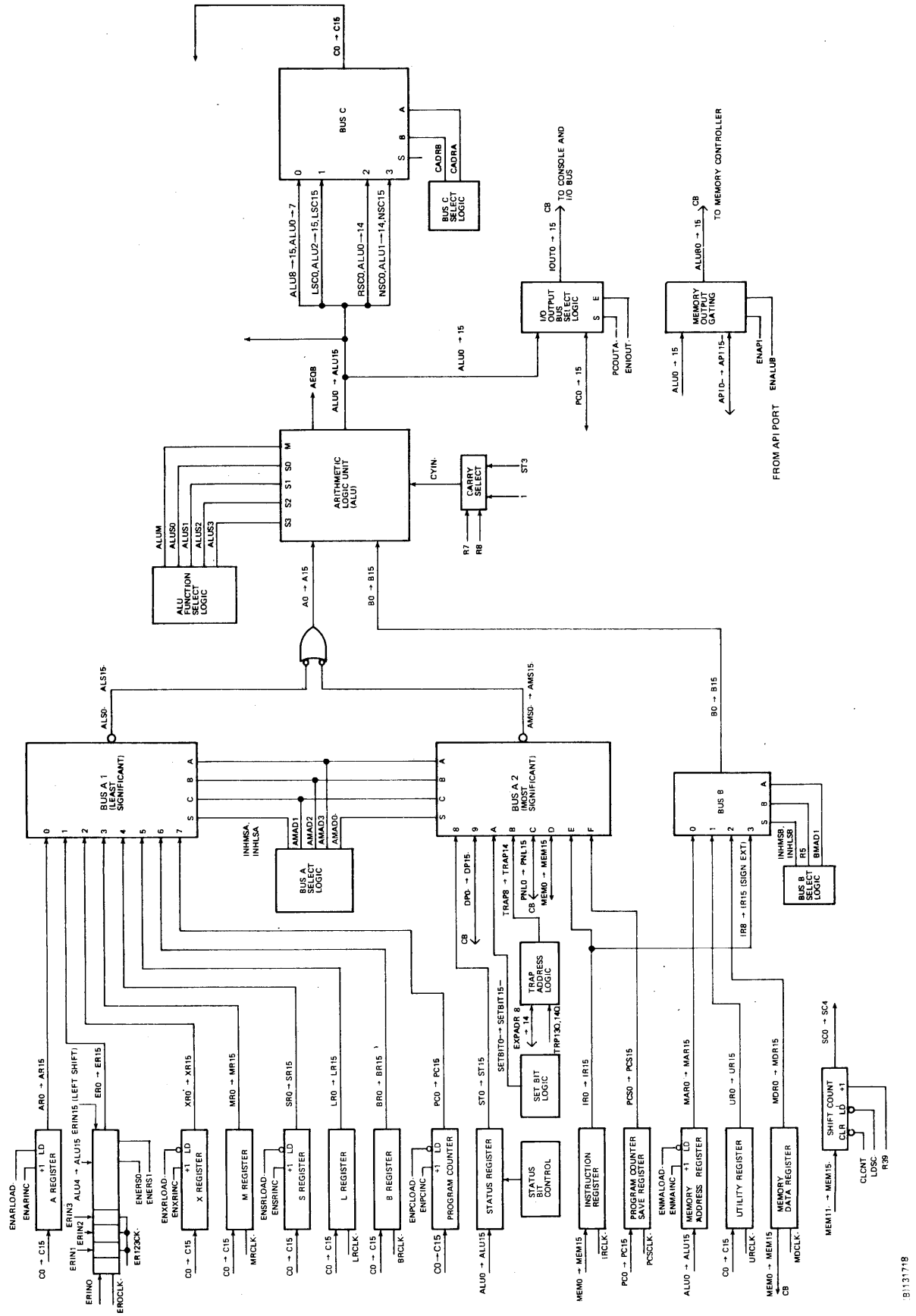


Figure 4-5. AU Functional Block Diagram

Table 4-1. Data Bus A Selection

Bus	Function Selected	Select Address S_XC B A	A0	Input Signals Selected as Outputs (A0→A15) A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
	AR→A	0 0 0 0	AR0															AR15
	ER→A	0 0 0 1	ER0															ER15
	XR→A	0 0 1 0	XR0															XR15
	MR→A	0 0 1 1	MR0															MR15
A1	SR→A	0 1 0 0	SR0															SR15
	LR→A	0 1 0 1	LR0															LR15
	BR→A	0 1 1 0	BR0															BR15
	PC→A	0 1 1 1	PC0															PC15
		S_YC B A																
	ST→A	0 0 0 0	ST0												ST12	0	ST14	ST15
	DP→A	0 0 0 1	DP0–															DP15–
	SETBIT→A	0 0 1 0	SETBIT0–															SETBIT15–
A2	TRAP→A	0 0 1 1	0							0	TRAP8						TRAP14	0
	PNL→A	0 1 0 0	PNL0															PNL15
	MEM→A	0 1 0 1	MEM0															MEM15
	IR→A	0 1 1 0	IR0															IR15
	PCS→A	0 1 1 1	PCS0															PCS15

BUS A1 ENABLES

S_X SMSH	S_X SLSH	A1 SELECT ENABLE
0	0	ENABLE A1
0	1	ENABLE MSH OF A1
1	0	ENABLE LSH OF A1
1	1	INHIBIT A1

BUS A2 ENABLES

S_y	ACTION
0	ENABLE A2
1	INHIBIT A2

NOTES:

1. SELECT ADDRESS DERIVED FROM FOLLOWING ROM BITS
   - S R1 (Refers to tables shown for logical values of R1)
   - C R2–
   - B R3–
   - A R4–
2. SELECT ADDRESS SIGNAL "SX" IS IMPLEMENTED TO ALLOW MOST SIGNIFICANT HALF, LEAST SIGNIFICANT HALF, OR BOTH HALVES OF FIRST 8 REGISTERS TO BE SELECTED TO BUS A. S_y IS EITHER ENABLED OR DISABLED.

Table 4-2. Data Bus B Selection

FUNCTION SELECTED	SELECT ADDRESS S	B	A	INPUT SIGNALS SELECTED AS OUTPUTS (B0→B15) B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
MAR→B	0	0	0	MAR0	←													→	MAR15
UR→B	0	0	1	UR0	←													→	UR15
MRD→B	0	1	0	MDR0	←													→	MDR15
IR→B	0	1	1	IR0	←	IR8 SIGN EXTENDED					→	IR8	←					→	IR15

NOTE: SELECT ADDRESS DERIVED FROM ROM BITS R5 AND R6.
S = INHLSB
A = BMA01 = RGT (IR5 · IR6 · IR7 · R51)
B = R5

4.4.6 **PROGRAM COUNTER AND SAVE REGISTERS.** The Program Counter (PC) is a 16-bit register that normally contains the address of the next instruction. The PC consists of 4-bit synchronous counters (SN74163). These counters can be parallel loaded with ALU output data or can be incremented. The load function is controlled by ROM control (ROM bit 42) or by logic that detects selection of the PC as a source or destination in a register to register instruction.

The output of the PC is one of the bus A selector inputs and is also an input to the Program Counter Save Register (PCS).

The PCS is an internal register that cannot be accessed by the programmer. The CPU utilizes the PCS for temporary storage during a store status block instruction. The PCS consists of D type flip-flops (SN74174). The control ROM generates the PCS clock and the PCS output is an input to the bus A selector.

4.4.7 **INSTRUCTION REGISTER.** The Instruction Register (IR) is a 16-bit register that retains the current instruction the CPU is executing. The IR consists of four registers (SN74175) that are loaded by the ROM generated IR clock with data from the memory output lines. Bus A selector and bus B selector inputs are supplied by the IR.

4.4.8 **ARITHMETIC REGISTER.** The primary arithmetic register or A Register (AR) is a 16-bit register that uses program control for add, subtract, AND, OR, and other functions. The AR consists of 4-bit counters (SN74163) which can be incremented, or parallel loaded with bus C data. Load and increment signals are generated by the control ROM. The AR output is connected to the bus A selector.

4.4.9 **SECONDARY ARITHMETIC REGISTER.** The secondary arithmetic register or E Register (ER) is a 16-bit register that utilizes program control to extend the A register for multiply, divide, and other functions. The ER consists of 4-bit shift registers (SN74194) for ER bits 4 through 15. Bits 0 through 4 are implemented with D type flip-flops and logic gates to handle register shift and arithmetic instructions that utilize ER and other registers. The ER has several clocks which are used for different parts of the ER. The ROM bits 28 through 34 control the operation of the ER.

Table 4-3. Data Bus C Selection

Function Selected	Select Address S	B	A	Input Signals Selected As Outputs Bus C (C0–C15) C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
CIR8–C	0	0	0	ALU8	ALU 9	ALU 10	ALU 11	ALU 12	ALU 13	ALU 14	ALU 15	ALU 0	ALU 1	ALU 2	ALU 3	ALU 4	ALU 5	ALU 6	ALU 7
LSHIFT–C	0	0	1	LSC0	ALU 2	ALU 3	ALU 4	ALU 5	ALU 6	ALU 7	ALU 8	ALU 9	ALU 10	ALU 11	ALU 12	ALU 13	ALU 14	ALU 15	LSC 15
RSHIFT–C	0	1	0	RSC0	ALU 0	ALU 1	ALU 2	ALU 3	ALU 4	ALU 5	ALU 6	ALU 7	ALU 8	ALU 9	ALU 10	ALU 12	ALU 12	ALU 13	ALU 14
ALU–C	0	1	1	NSC0	ALU 1	ALU 2	ALU 3	ALU 4	ALU 5	ALU 6	ALU 7	ALU 8	ALU 9	ALU 10	ALU 11	ALU 12	ALU 13	ALU 14	NSC 15

Notes:

Select address derived from the following:

S Logic 0
B CADRB = R16–*DIVIT–*BYTESHFT–
A CADRA = R17–*MPYIT–*BYTESHFT–

Table 4-4. Arithmetic Logic Functions

SELECTION				ALU FUNCTION		
				M = H	M=L ; ARITHMETIC OPERATIONS	
S_3	S_2	S_1	S_0	LOGIC FUNCTIONS	CYIN = 0	CYIN = 1
L	L	L	L	$F = \overline{A}$	F = A	F = A PLUS 1
L	L	L	H	$F = \overline{A + B}$	F = A + B	F = (A + B) PLUS 1
L	L	H	L	$F = \overline{A}B$	$F = A + \overline{B}$	$F = (A + \overline{B})$ PLUS 1
L	L	H	H	F = 0	F = MINUS 1 (2's COMPL)	F = ZERO
L	H	L	L	$F = \overline{AB}$	$F = A$ PLUS $A\overline{B}$	$F = A$ PLUS $A\overline{B}$ PLUS 1
L	H	L	H	$F = \overline{B}$	$F = (A + B)$ PLUS $A\overline{B}$	$F = (A + B)$ PLUS $A\overline{B}$ PLUS 1
L	H	H	L	$F = A \oplus B$	F = A MINUS B MINUS 1	F = A MINUS B
L	H	H	H	$F = A\overline{B}$	$F = A\overline{B}$ MINUS 1	$F = A\overline{B}$
H	L	L	L	$F = \overline{A} + B$	F = A PLUS AB	F = A PLUS AB PLUS 1
H	L	L	H	$F = \overline{A \oplus B}$	F = A PLUS B	F = A PLUS B PLUS 1
H	L	H	L	F = B	$F = (A + \overline{B})$ PLUS AB	$F = (A + \overline{B})$ PLUS AB PLUS 1
H	L	H	H	F = AB	F = AB MINUS 1	F = AB
H	H	L	L	F = 1	F = A PLUS A*	F = A PLUS A PLUS 1
H	H	L	H	$F = A + \overline{B}$	F = (A + B) PLUS A	F = (A + B) PLUS A PLUS 1
H	H	H	L	F = A + B	$F = (A + \overline{B})$ PLUS A	$F = (A + \overline{B})$ PLUS A PLUS 1
H	H	H	H	F = A	F = A MINUS 1	F = A

*Each bit is shifted to the next more significant position.

4.4.10 BASE, LINK, AND MAINTENANCE REGISTERS. The Base Register (BR), Link Register (LR), and Maintenance Register (MR) are 16-bit registers under program control that are implemented with SN74174 registers. The MR is used for temporary storage. The LR is used to hold return address for subroutine linkage. The BR is used to hold base addresses for BR related instructions. These three registers are loaded utilizing ROM control.

4.4.11 STORAGE REGISTER. The Storage Register (SR) is also used for temporary data storage under program control. The SR is implemented with SN74163 counters. This register can be parallel loaded with bus C data or incremented during byte instructions. Load and increment signals for the SR are ROM generated.

4.4.12 INDEX REGISTER. The Index Register (XR) is a 16-bit register under program control that is used during indexed instructions. The XR uses ROM generated control signals and is also implemented with SN74163 counters.

4.4.13 **STATUS REGISTER.** The Status Register (ST) is a 15-bit register that consists of individual flip-flops. The contents of the ST are used to indicate program and machine status. Five ROM outputs (R23 through R27) control the ST. Outputs R23, R24, and R25 are decoded into seven clock enable signals by an SN7442 4 to 10 line decoder. The remaining two ROM outputs are used for separate clock enable signals. Parallel loading of the ST with ALU data is accomplished with load status instructions or by console operation. Individual bits are set or reset by program or machine condition changes. Status Register bits 0 (ST0) and 1 (ST1) function as comparison indicators to indicate the result of the last compare operation as listed in table 4-5.

Additionally the Auxiliary Processor Port (APP) has access to these bits and can set the state of these bits if the APP external controller requires such action.

The ST0 and ST1 bits are implemented with D type flip-flops with appropriate steering logic for the D input and clock. The compare condition is gated to the D input by an enable signal from the decoder that is described in previous paragraphs. The ROM control is used to enable clock signals to the flip-flop.

Status register bit 2 is the overflow indicator that is turned on or off by instructions which cause overflow. The ST2 bit is implemented with a D type flip-flop and steering logic for the D input and clock. The ST2 bit is set during certain arithmetic shift, two's-complement, left shift, multiply, and divide instructions. The APP also has access to the ST2 bit.

Status Register bit 3 (ST3) is the carry indicator. This bit is also implemented with a D type flip-flop. The ST3 is turned on by any add, subtract, or complement instruction that results in a carry into the sign bit (bit 0) of a register. If these instructions do not result in a carry into the sign bit, the carry indicator is turned off. The carry condition is determined by logic that monitors bit 0 of the bus A and bus B and bit 0 of the sum of bus A data and bus B data at the output of the ALU. The logic equation for carry appears below utilizing bus A bit 0 (A0) and bus B bit 0 (B0), and the ALU bit 0 (ALU0) output:

$$\text{Carry} = \overline{\text{ALU0}}(\text{A0}\overline{\text{B0}} + \overline{\text{A0}}\text{B0}) + \text{ALU0}\ \overline{(\text{A0}\overline{\text{B0}} + \overline{\text{A0}}\text{B0})}$$

The carry bit can also be set by an external controller utilizing the auxiliary processor port during an Auxiliary Processor Initiate (API) instruction.

Status Register bit 4 (ST4) is the Privileged Instruction and Memory Protect (PIF) control bit. If ST4 is set, any attempt to address protected memory or execute privileged instructions will cause an internal interrupt and the CPU to trap to location 2. The ST4 bit is implemented using a D type flip-flop with ROM enables on the D input and clock that allow ST4 to be loaded with ALU4. The output of ST4 is gated to the memory controller. The gate is disabled during interrupts or when an instruction is being acquired in the Single Instruct mode (SIE).

Table 4-5. ST0 and ST1 Comparison

ST0	ST1	RESULTS
0	0	Less than
0	1	Equal to
1	0	Greater than
1	1	Not allowed

Status Register bit 5 (ST5) is the PIF address violation indicator. When a program attempts to address protected memory while ST4 is set, ST5 is set and an internal interrupt occurs. The ST5 bit is implemented using a J-K flip-flop with an address violation (ADRV–) from the memory controller on the J input and status register clear (STLDCLR) on the K input.

Status Register bit 6 (ST6) is the PIF instruction violation indicator. If a program attempts to execute a privileged instruction while ST4 is set, an internal interrupt occurs. The ST6 bit is also implemented using a J-K flip-flop with the instruction violation line from the memory controller on the J-input and status register clear on the K-input. The J-input indicates an instruction violation (INSTV) without an address violation (ADRV–).

Status Register bits 7, 8, 11, and 12 (ST7, ST8, ST11, and ST12) are interrupt enables for the I/O interrupt (ST7), the external interrupt (ST8), the memory parity error interrupt (ST11) and the DMAC interrupt (ST12). These ST bits are implemented using D type flip-flops with ROM control logic.

Status Register bit 9 (ST9) is the PIF lower limit address bias enable. If this bit is set, the contents of the PIF lower limit register plus one is added to all CPU memory access addresses. The ST9 bit is implemented with a D type flip-flop and is gated before being routed to the memory controller. As with the ST4 bit, the gated control is disabled by IFLAG– during interrupt processing.

Status Register bit 10 (ST10) is the index indicator and is implemented with a D type flip-flop. When bit ST10 is set, indexing precedes indirect addressing if used. If ST10 is not set, indexing follows indirect addressing. Unless indirect addressing is used, this bit is ignored.

Status Register bit 14 (ST14) is the memory parity error indicator implemented with a J-K flip-flop. The memory parity error signal (MPERR) from the memory controller sets ST14.

Status Register bit 15 (ST15) is the power fail indicator set by the power supply approximately 1.0 millisecond before power failure occurs causing an internal interrupt. The ST15 bit is implemented with a J–K flip-flop and so is another flip-flop, PFMSKQ, associated with ST15 operation. The power fail signal (POFF) from the power supply is buffered to produce PFAIL and is applied to ST15 and another double flip-flop that supplies a one clock time reset pulse concurrent with the trailing edge of the POFF signal. This reset pulse resets the RUN flip-flop (RUNFFQ). In this buffer, a wired-OR (POFF–) allows other external power supplies (such as an external I/O chassis) to cause a power fail trap. When POFF goes high, to indicate an imminent power failure, ST15 is set which sets PFMSKQ. With PFMSKQ set, the power fail signal is inhibited from ST15 to prevent subsequent power fail traps that could occur since POFF remains a logic One and ST15 is reset after the trap. After approximately 1.0 millisecond, POFF goes low resetting RUNFFQ and halting the CPU. Approximately 1.0 millisecond later, MRESET– goes low to reset the CPU including PFMSKQ. Now the CPU is in a reset state and loss of power can occur without harm.

4.4.14 MEMORY ADDRESS REGISTER. The Memory Address Register (MAR) is a 16-bit register that holds the address for any CPU memory accesses. The MAR is implemented both in the arithmetic unit and the memory controller since there is only one 16-line data bus (ALUB0-ALUB15) to the memory controller. This data bus must carry both the memory access addresses and memory write data.

The MAR in the AU is implemented with four SN74163 counters to permit parallel loading of ALU data and incrementing by a factor of one. The MAR control signals are also ROM generated.

4.4.15 **UTILITY REGISTER.** The Utility Register (UR) is a 16-bit register which is not program accessible. The UR, is implemented with SN74174 hex registers and is used as a temporary internal storage register during certain instructions such as some register-to-register (RR) instructions. The UR can be parallel loaded with bus C data and the UR output is a bus B selector input.

4.4.16 **MEMORY DATA REGISTER.** The Memory Data Register (MDR) is a 16-bit register that holds the memory read data (MEM0–MEM15) from the memory controller.

The MDR is implemented with SN74174 registers which are parallel loaded by the ROM control. The MDR output is one of the bus B selector inputs.

4.4.17 **SHIFT COUNTER.** The Shift Counter (SC) is a 5-bit counter that counts the number of shifts performed during shift, multiply, and divide instructions, to enable the various registers during the Load Register File (LRF) instruction, and to provide the proper timing sequence during Read and Write Direct Single (RDS and WDS) instructions. The SC is implemented with SN74163 counters. During multiply and divide instructions, the SC is cleared and incremented 16 times. This factor is arrived at by decoding $0F_{16}$ on the counter outputs, then allowing one more count. During the LRF instructions, the SC is cleared and incremented to 06_{16} (7 times). During shift instructions, the SC is loaded with one's-complement of the desired shift count. After the one's-complement is loaded, the SC is incremented to 1F to produce the correct number of shifts.

4.5 READ ONLY MEMORY CONTROLLER.

The Read-Only-Memory (ROM) controller generates basic control signals to enable data flow and logic functions in each state of the arithmetic unit microsequence. A functional block diagram of the ROM controller is illustrated in figure 4-6. Another ROM controller is the Basic Function ROM (BFR) which is discussed in Section 4.6.4.

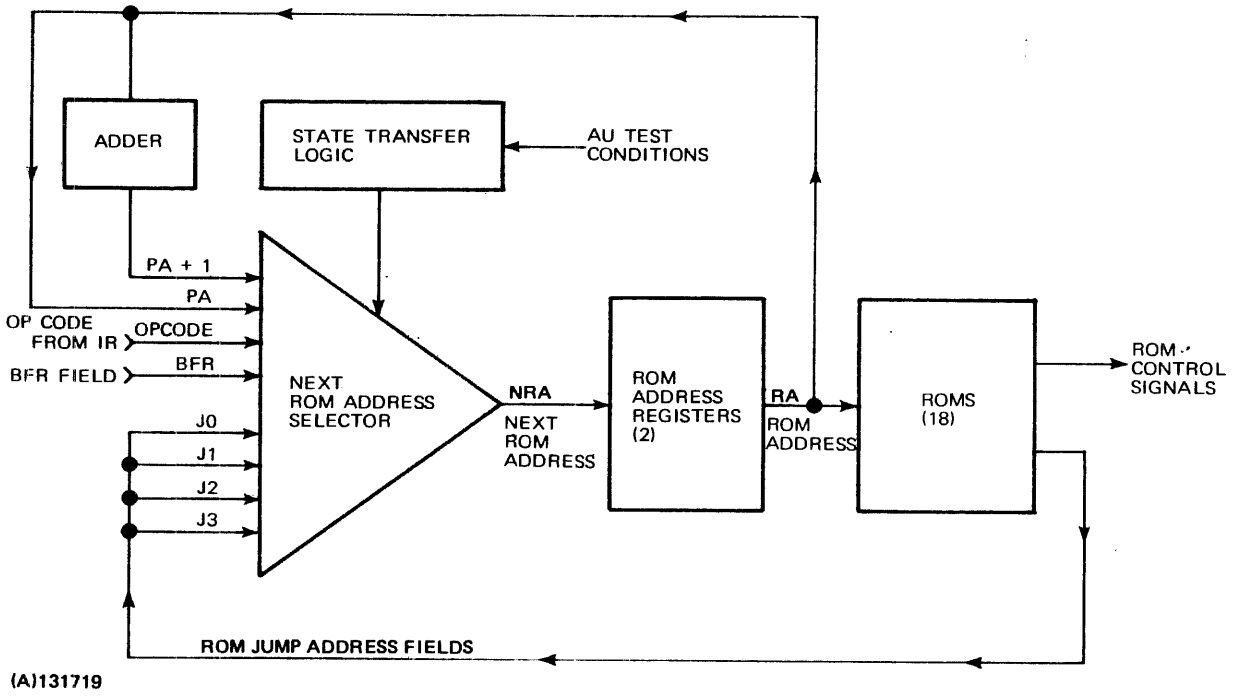


Figure 4-6. ROM Controller Functional Diagram

4.5.1 **READ ONLY MEMORY.** Read-Only-Memory (ROM) is implemented with eighteen 1024-bit Transistor-Transistor Logic (TTL) ROM's (SN74187). Each network is organized as 256 words with 4 bits in each word. The contents of each word is designed to produce the correct control signals for each AU microsequence state. The words are permanently coded in the ROM during manufacturing. The individual networks are operated in parallel to provide the ROM with 256 72-bit words. The basic ROM word format is illustrated in figure 4-7.

The first 56 bits (R1–R56) are distributed throughout the AU control logic. Bits 57 through 72 (R57 through R72) are divided into four (4) fields with four (4) bits in each field. The fields are combined in pairs (8 bits total) to form four jump addresses that are supplied to the next address selector logic. Each of the 18 ROM packages has a unique network number assignment since each contains different data. The ROM output signals, network numbers, TI Part Numbers, and AU board locations are listed in table 4-6.

4.5.2 **ROM ADDRESS REGISTERS.** The present 8-bit ROM address is retained in two duplicate 8-bit registers. One register is located on each AU board. On AU1, the register is implemented with two 4-bit registers (SN74175). On AU2, the ROM address register is implemented with four dual D type flip-flops (SN74H74). Since the register on the AU2 board does not provide sufficient drive to address all 14 ROM networks, buffer inverter gates are utilized on the Q outputs to address five of the AU2 ROMS. The remaining nine AU2 ROMS are driven from the Q output of the ROM address register.

A functional diagram of the ROM address logic is illustrated in figure 4-8. The registers are parallel loaded at each positive-edge of a free-running False system clock signal. The occurrence of ROM address loading defines the point of ROM state transition. The new address that is loaded is obtained from the next ROM address selector.

4.5.3 **NEXT ROM ADDRESS SELECTOR.** While in a given ROM state the Next ROM Address (NRA) to be loaded into the ROM address registers can be any one of nine possible addresses. These nine addresses are provided to the ROM address registers by an 8 to 1 selector that is implemented with SN74H52 and SN74H61 expandable AND/OR gates. The implementation of the most significant bit of the Next ROM Address (NRA) is shown in figure 4-9. The ninth address is produced by forcing the output of the selector to state 00. This is caused by a signal (FORCE00) generated during PIF violations and master reset conditions. Signal FORCE00 disables all enables to the AND/OR gates (i.e., SELJ0, SELJ1, etc.).

Table 4-7 lists the signals which are enabled by the select signals. The select signals are generated by the test logic that is described in the next section of this manual.

Incremented ROM addresses that are used in microsequencing the AU will affect only the least significant 4 bits of the Present Address (PA). To generate the PA+1 value, the least significant half of PA is added to 0 with a carry-in using a SN7483 4-bit adder. The adder outputs are supplied to the next address selector as RAINC4-7.

4.5.4 **STATE DECODERS.** The test logic that determines which select to use for the NRA is designed to utilize the outputs of two 4 to 16 line decoders (SN74154) that decode each half of the ROM address into 16 lines. The negative outputs of the most significant half decodes are labeled MO– to MF–. The least significant half are labeled LO– to LF–. The outputs are inverted and both polarities are made available to decode any desired state.

4.5.5 **STATE TRANSFER LOGIC.** The state transfer logic utilizes the proper state decode outputs, pertinent data conditions, and external inputs to provide the correct NRA selector enables. A functional diagram of the state transfer logic is illustrated in figure 4-10.

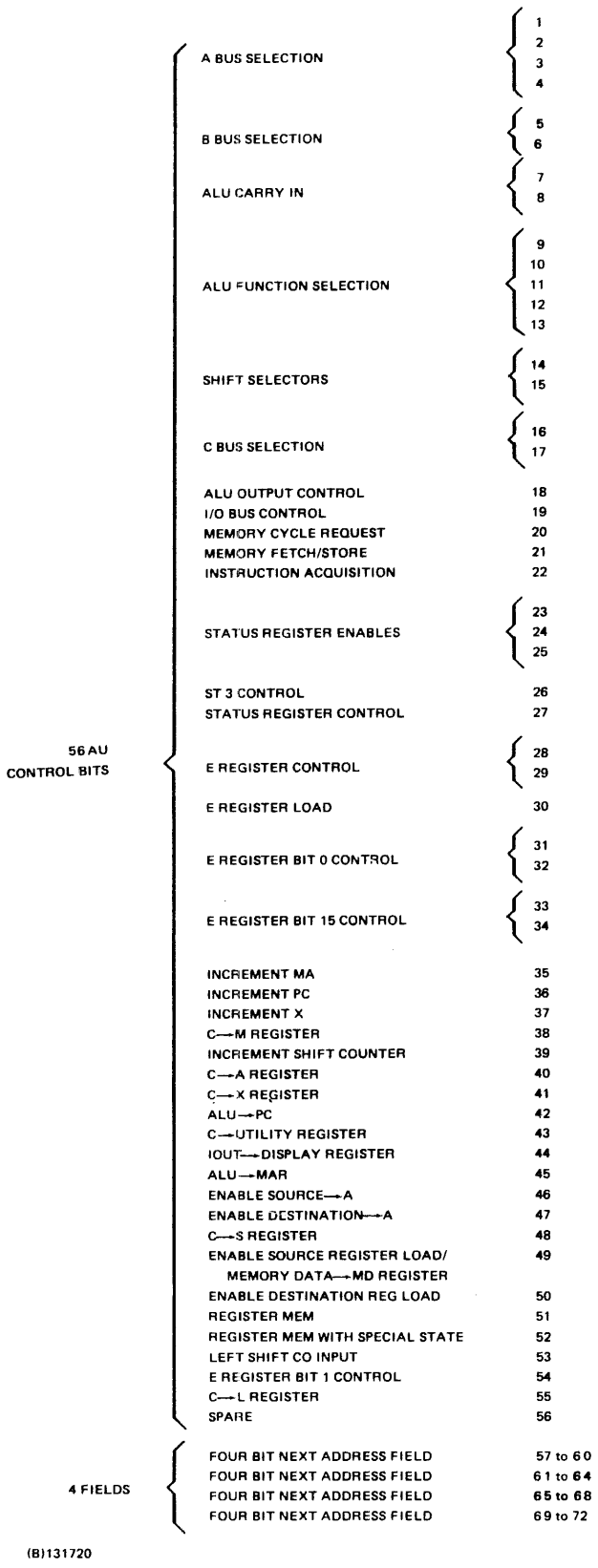


Figure 4-7. Basic ROM Word Format

Table 4-6. Control ROM Network and Part Numbers

OUTPUT SIGNALS	NETWORK NUMBER	TI PART NUMBER	LOCATION		
			BOARD	ROW	COLUMN
R1 – R4	SN37287N	226604–0027	AU2	18	02
R5 – R8	SN37288N	226604–0028	AU2	14	02
R9 – R12	SN38963N	226604–0051	AU2	09	01
R13 – R16	SN38964N	226604–0052	AU2	12	01
R17 – R20	SN38965N	226604–0053	AU2	19	01
R21 – R24	SN38966N	226604–0054	AU2	15	02
R25 – R28	SN38967N	226604–0055	AU2	15	01
R29 – R32	SN37294N	226604–0034	AU2	17	02
R33 – R36	SN37295N	226604–0035	AU2	19	02
R37 – R40	SN38968N	226604–0056	AU2	10	01
R41 – R44	SN37297N	226604–0037	AU2	13	01
R45 – R48	SN38969N	226604–0057	AU2	17	01
R49 – R52	SN37299N	226604–0039	AU2	18	01
R53 – R56	SN37404N	226604–0040	AU2	11	01
R57 – R60	SN37405N	226604–0041	AU1	16	02
R61 – R64	SN37406N	226604–0042	AU1	15	03
R65 – R68	SN37407N	226604–0043	AU1	17	02
R69 – R72	SN38970N	226604–0058	AU1	17	03

The test logic section contains a combination of logic that is used to test conditions required for some ROM states used in the microsequence. The test logic generates seven select signals and the FORCE00 signal as shown in figure 4-10. The eighth select signal (SELJ0) is generated when all other select signals are absent. This logic is also used for the default path.

4.6 SPECIAL CONTROL LOGIC

The Arithmetic Unit (AU) has several additional groups of miscellaneous control logic. These control logic groups include:

- Decision dependent logic
- Interrupt Control logic
- Skip logic
- Basic Function ROM.

4.6.1 DECISION DEPENDENT LOGIC. The decision dependent logic is a logic block that combines ROM output line, ROM state decodes, and AU data information to form some of the data path register and selector control signals. Decision dependent logic is implemented on both AU boards.

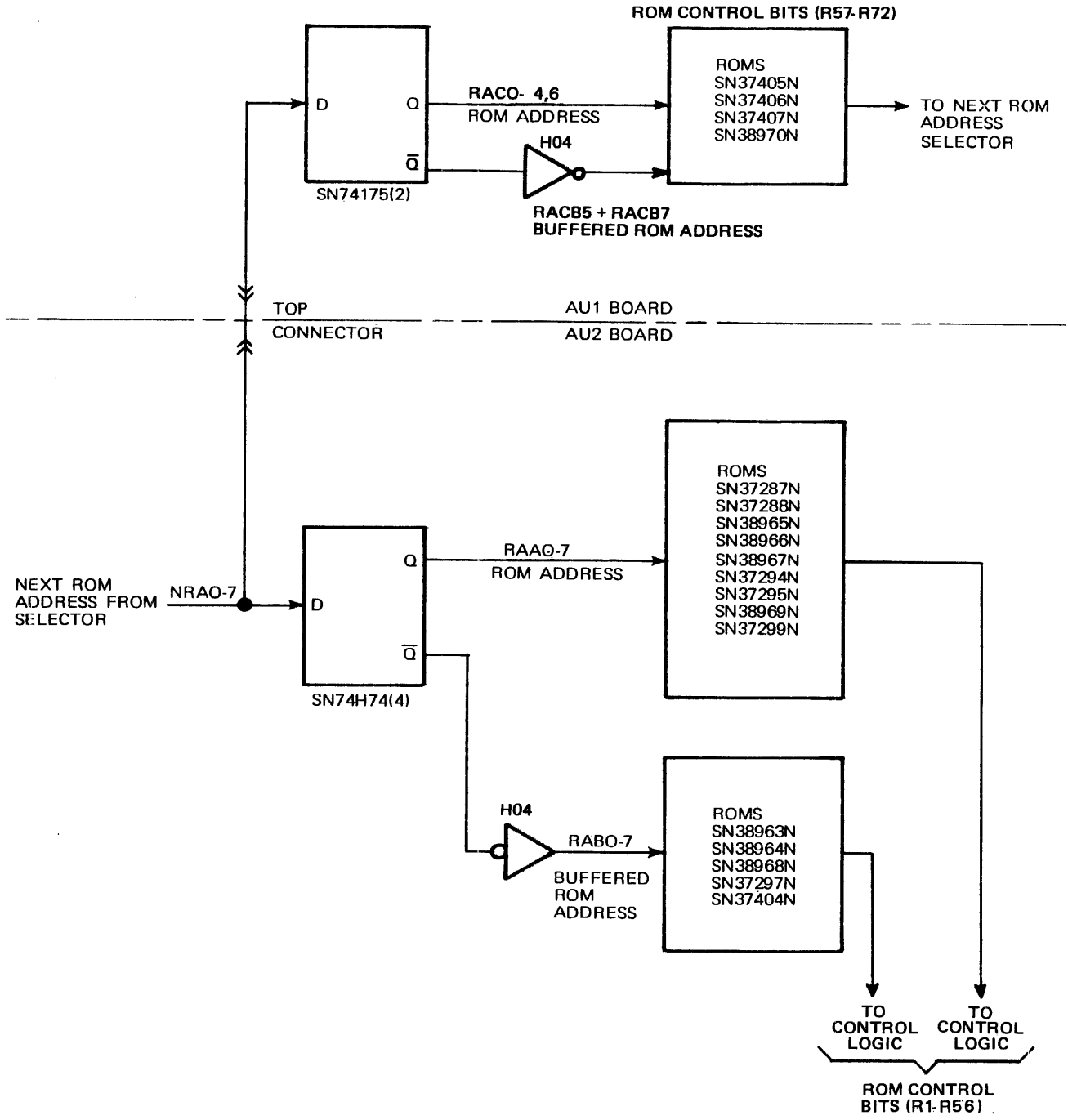


Figure 4-8. ROM Address Functional Diagram

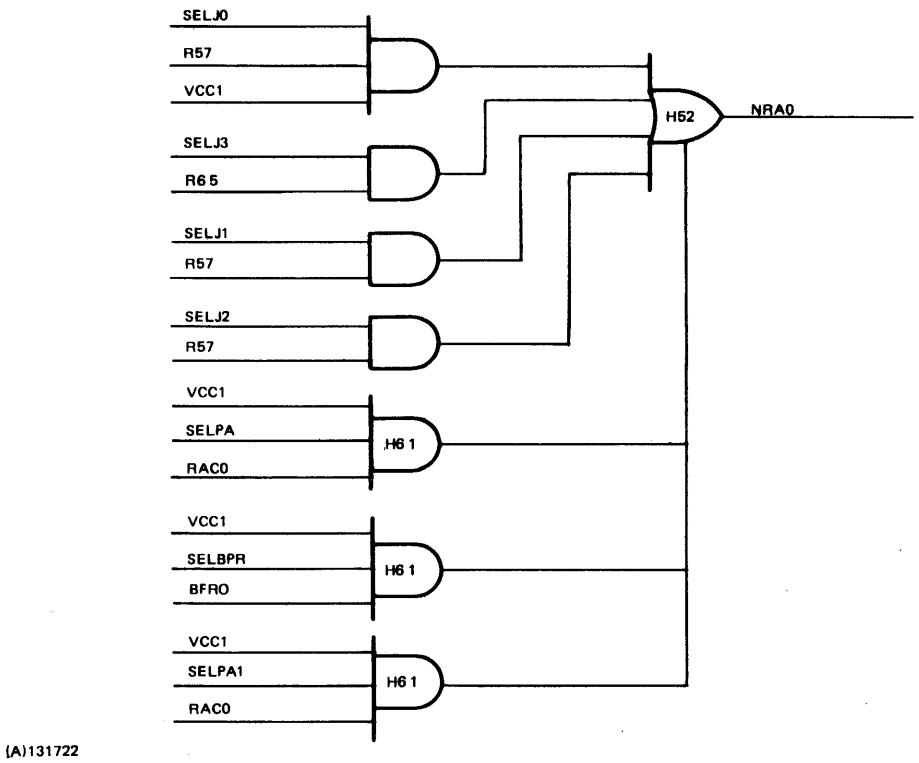


Figure 4-9. Next ROM Address Selection - Implementation of Most Significant Bit

Table 4-7. Signals Selected as Next ROM Addresses

NEXT ADDRESS NAME	SELECT SIGNAL	DATA SELECTED FOR NEXT ROM ADDRESS							
		NRA0	NRA1	NRA2	NRA3	NRA4	NRA5	NRA6	NRA7
JUMP ADDRESS 0	SELJ0	R57	R58	R59	R60	R61	R62	R63	R64
JUMP ADDRESS 1	SELJ1	R57	R58	R59	R60	R65	R66	R67	R68
JUMP ADDRESS 2	SELJ2	R57	R58	R59	R60	R69	R70	R71	R72
JUMP ADDRESS 3	SELJ3	R65	R66	R67	R68	R69	R70	R71	R72
PRESENT ADDRESS	SELPA	RAC0	RAC1	RAC2	RAC3	RAC4	RAC5	RAC6	RAC7
PRESENT ADDRESS PLUS ONE	SELPA1	RAC0	RAC1	RAC2	RAC3	RAINC4	RAINC5	RAINC6	RAINC7
INSTRUCTION OP CODE (REGISTER MEMORY (RM) INST.)	SELOP	0*	1	**	IR0	IR1	IR2	IR3	IR4
INSTRUCTION OF CODE (OTHER THAN RM INST.)	SELOP	0*	1	NC	NC	IR5	IR6	IR7	IR8
BASIC FUNCTION ROM ADDRESS	SELBFR	BFR0	BFR1	BFR2	BFR3	BFR4	BFR5	BFR6	BFR7

* Indicates this bit does not have an expander gate for this input; therefore, the select will cause a ZERO output for this bit.

** The "OR" of IR0 and IR1 is connected to this input.

4.6.2 **INTERRUPT CONTROL LOGIC.** The interrupt control logic is illustrated in figure 4-11. The DMAC, I/O, and EXPANDER interrupts are implemented by three D type flip-flops with buffered inputs that have pull-ups to inhibit operation if any of these three device types are not in the system. Each flip-flop output is enabled to the trap address logic by two signals. The first signal is the pertinent status register control bit and the second is an inhibit signal which disables the interrupts during a register-to-register instruction with a status register destination. The enabled signals are coded along with other signals that indicate an internal interrupt and an interrupt state to form two signals (T40R6 and T20R6). These signals set the proper trap address in the trap address flip-flops (TRP13Q and TRP14Q) which form bits 13 and 14 of the trap address. The address signals are OR'ed with the optional expanded interrupt address lines to form TRAP8=14. TRAP15 is always a ZERO since all trap addresses are even.

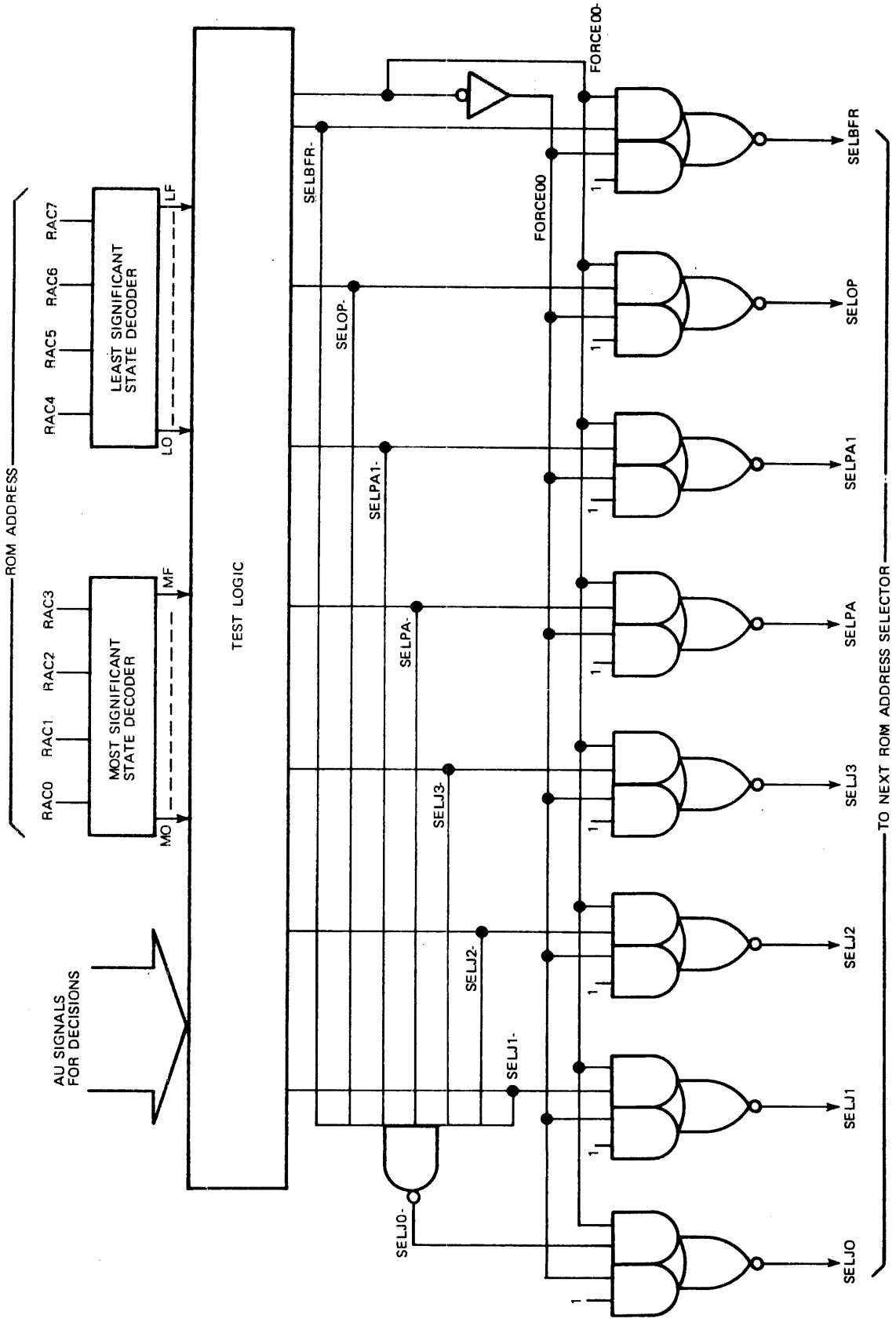


Figure 4-10. State Transfer Functional Diagram

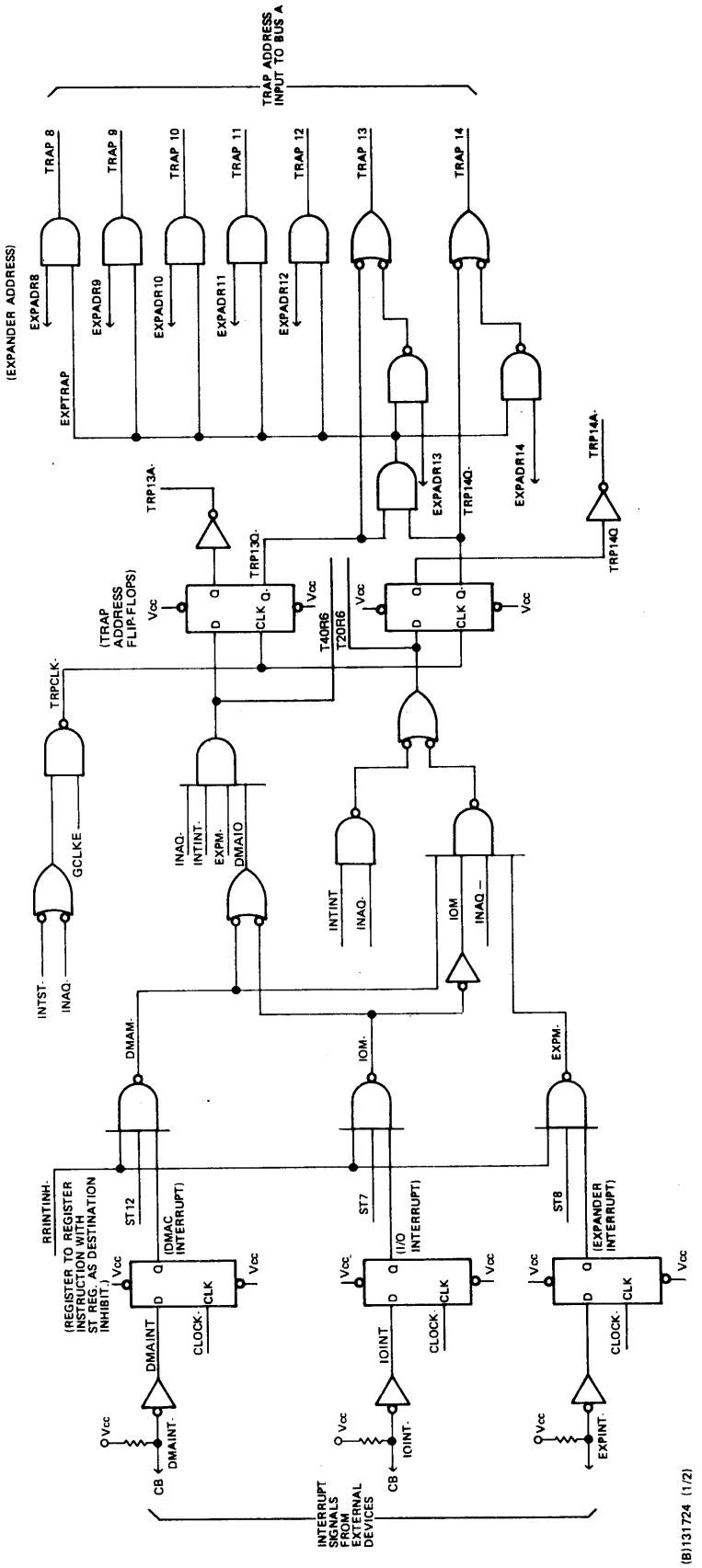


Figure 4-11. Interrupt Control Logic Diagram (Sheet 1 of 2)

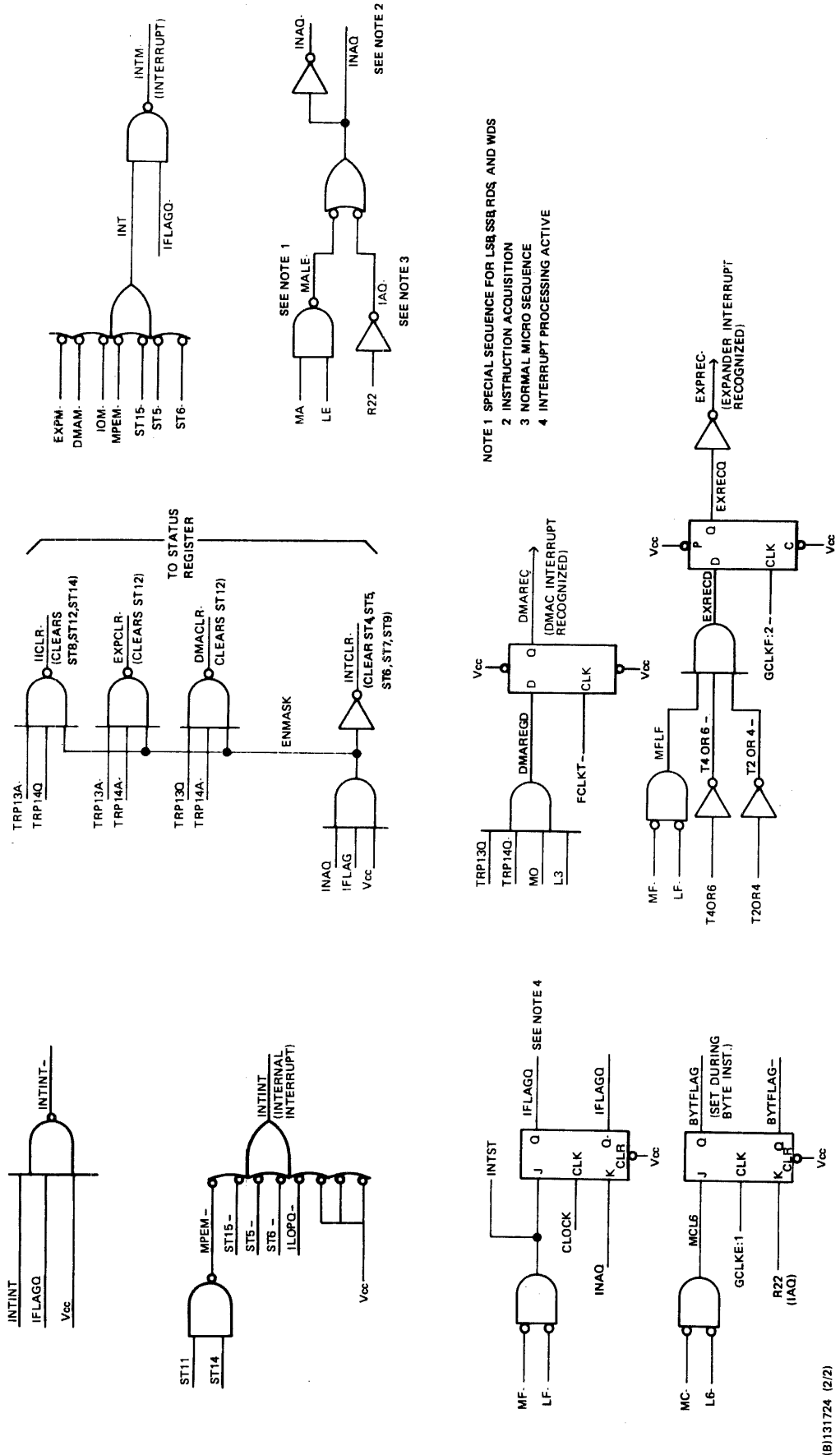


Figure 4-11. Interrupt Control Logic Diagram (Sheet 2 of 2)

The expander address lines (EXPADR8-14) are enabled when EXPTRAP goes to a logic one. The EXPTRAP lines goes to a logic one only during an expanded interrupt option trap. The TRAP8 through TRAP14 lines are inputs to the bus A selector as shown in figure 4-5. This logic includes two interrupt lines (INTM– and INTINT) that indicate an interrupt of any kind (INTM–), an internal interrupt (INTINT), and several flip-flops. The IFLAGQ flip-flop is set at the entry state (state FF) to the interrupt microsequence and is used to indicate an interrupt is being processed. The BYTFLG flip-flop is set during byte processing instructions. If the BYTFLG flip-flop is set and an interrupt occurs during a byte instruction, proper exit from the byte microsequence occurs. The DMAREC and EXPREC flip-flops are used to supply one clock time interrupt recognized signals to the DMAC or expanded interrupt option. The remaining gates form a group of clear signals which are supplied to the status register. The clear signals (IICLR–, EXPCLR–, DMACLR–, and INTCLR–) reset the proper status register bits as each type of interrupt is processed. The INAQ signal is used primarily by the memory controller to establish when an instruction is being acquired by the normal microsequence (R22, IAQ–) or by a restricted instruction (state decode of AE, MALE–).

4.6.3 SKIP LOGIC. The skip logic utilizes five 8 to 1 selectors (SN74151) and several decoders implemented with gates to test conditions required for skip instructions. Two of the selectors select the proper Utility Register (UR) bit for the test bit instructions. The remaining three selectors select the tested condition for each of the three types of skip instructions (register skips, indicator skips, or bit test skips). Table 4-8 lists the conditions tested by the three skip selectors. The outputs of the selectors are OR'ed to form the skip final (SKIP) which is utilized in the test logic.

4.6.4 BASIC FUNCTION ROM. The Basic Function ROM (BFR) is a small ROM controller which performs several functions during instruction execution. The BFR is implemented with two 256 X 4 TTL ROMS and one 32 X 8 TTL ROM. The BFR basically monitors operational code bits from memory and generates the next AU microsequence state for the instruction fetches. Table 4-9 lists the BFR ROM output signatures, TI part numbers, network numbers and AU board locations.

The two 256 X 4 ROMS (SN37409N and SN38971N) are addressed by combinational logic and memory bits 5 through 10. The single 32 X 8 ROM (SN37415N) is addressed by memory bits 0 through 4 and supplies the signals listed in Table 4-10.

The outputs of the 256 X 4 ROMS form one of the next ROM address (NRA) selector inputs and are enabled onto the NRA bus by SELBFR. The BFR3-7 outputs are also wire AND'ed with a set of open collector gates that allow the operational code in memory bits 0 through 4 to be gated on the BFR3-7 lines. State FF is forced on BFR0-7 by two signals: ILLOP from the 32 X 8 ROM and ILLOPA generated by memory line decoding gates. These signals are connected to the 256 X 4 ROM enable inputs. When either input is held high, the 256 X 4 ROM outputs go to an all ONEs condition; i.e., state FF. The BFR logic additionally contains three D type flip-flops that store the RR, BSDQD, and DBL signals from the 32 X 8 ROM. The BFR4-7 lines also have a wired AND signal (FORCZ inverted with open collector inverters) which causes the BFR4-7 lines to go to ZERO for any shift instruction with a zero shift count except ALA and ALD instructions.

4.7 ARITHMETIC UNIT MICROSEQUENCE.

The AU microsequence defines the step-by-step sequence of the ROM through firmware control signal contents. Detailed operation of the AU is specified by the microsequence. Basic AU operation cycles are divided into major decisions and events which are performed by the ROM microsequence. The basic AU cycle flowchart is illustrated in figure 4-12.

Table 4-8. Skip Selector Conditions

IR10	IR9	IR8	TEST SIGNAL SIGNATURE	CONDITION TESTED
			REGISTER SKIPS*	
0	0	0	AEQZ	BUS A EQUAL ALL ZERO'S
0	0	1	AEQO	BUS A EQUAL ALL ONE'S
0	1	0	A15	BUS A BIT 15 EQUAL ONE
0	1	1	ALS0	BUS A BIT 0 EQUAL ONE
1	0	0	AEQZ–	BUS A NOT EQUAL ALL ZERO'S
1	0	1	AEQ0–	BUS A NOT EQUAL ALL ONE'S
1	1	0	ALS15–	BUS A BIT 15 NOT EQUAL ONE
1	1	1	ALS0–	BUS A BIT 0 NOT EQUAL ONE
			INDICATOR SKIPS*	
0	0	0	SLT	COMPARE BIT (ST0, ST1) SET FOR LESS THAN
0	0	1	SEQ	COMPARE BITS SET FOR EQUAL
0	1	0	SGT	COMPARE BITS SET FOR GREATER THAN
0	1	1	ST2	STATUS REGISTER BIT 2 (ST2) EQUAL ONE
1	0	0	SGE	COMPARE BITS SET FOR GREATER THAN OR EQUAL
1	0	1	SEQ–	COMPARE BITS SET FOR NOT EQUAL
1	1	0	SLE	COMPARE BITS SET FOR LESS THAN
1	1	1	ST2–	STATUS REGISTER BIT 2 NOT EQUAL ONE
			BIT SKIPS*	
0	0	0	SSNEG	SENSE SWITCH NOT EQUAL
0	0	1	SSNEG	SENSE SWITCH EQUAL
0	1	0	ST3–	STATUS REGISTER BIT 3 (ST3) NOT EQUAL ONE
0	1	1	ST3	STATUS REGISTER BIT 3 EQUAL ONE
1	0	0	BITEQO–	SELECTED BIT NOT EQUAL ONE
1	0	1	BITEQO	SELECTED BIT EQUAL ONE
1	1	0	N.C.	NOT CONNECTED
1	1	1	N.C.	NOT CONNECTED

* DETERMINED BY DECODING IR BITS 3, 6, 8, 11.

Table 4-9. Basic Function ROM Network and Part Numbers

OUTPUT SIGNALS	NETWORK NUMBER	TI PART NUMBER	TYPE	LOCATION		
				BOARD	ROW	COLUMN
BFR0- BFR3	SN37409N	226604–0045	256X4	AU1	9	4
BFR4–BFR7	SN38971N	226604–0046	256X4	AU1	8	4
SEE BELOW*	SN37415N	226748–0012	32X8	AU1	6	11

*Seven of eight outputs are used, these are:
DBL, RR, BSDQD, C8OP, INSRTOPA-, SPLIMM-, and ILLOP-.

Table 4-10. 32X8 ROM Output Signals

32X8 OUTPUT BIT	SIGNATURE	FUNCTION
Y1	NONE	Not used.
Y2	DBL	Indicates double word instructions.
Y3	RR	Indicates register to register instructions.
Y4	BSDQD	Indicates BIX, DMT, IMO, BRL, BRU, STA, STE, STX, DST, DSB DLD, DAD instructions.
Y5	C8OP	Indicates shift instructors.
Y6	INSRTOPA–	Indicates when operational code should be used to determine part of next ROM state.
Y7	SPLIMM–	Indicates special immediate instructions, such as: DSB, DLD, DAD, CPL, BRU, IOR, and AND.
Y8	ILLOP	Indicates operational code is illegal and causes 256x4 to go to interrupt state FF.

4.7.1 POWER UP SEQUENCE. When power is switched on, the power supply master reset (MRESET–) signal forces the ROM address to 00 via the FORCE00 signal as explained in the section on the Next ROM Address selector theory of operation. This is the initial state for the AU ROM. The AU remains in state 00 for the duration of the reset signal. If the MODE switch on the control console is in the RUN position, the RUNFF is set and the AU executes instructions from memory location 0 when the power supply releases MRESET. The MRESET signal also causes status register bits 4, 5, and 6 to reset in state 00. Since state 00 is encountered for conditions other than power up, status register bits 5 and 6 (ST5 and ST6) are examined and a interrupt trap is initiated if a PIF violation has occurred.

4.7.2 **BASIC INSTRUCTION CYCLES.** The basic instruction cycle begins with instruction acquisition as the instruction appears on the memory lines, the BFR logic provides the first microsequence state for the AU if the operational code is legal (Reference paragraph 4.6.4). The RUN flip-flop (RUNFF) is then tested to determine if the system has halted. If RUNFF is set and no interrupts have occurred, the AU acquires the operand and executes the instruction if required. Since the PC and MAR have been incremented, the AU can acquire the next instruction and repeat the instruction cycle.

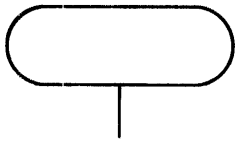
4.7.3 **INTERRUPT CYCLE.** Several paths for processing interrupts are shown in figure 4-12. The first trap occurs at power up as a result of the MRESET from the power supply. (Reference paragraph 4.7.1.) As each instruction is acquired from memory, two interrupt checks are performed. The first check is performed as the operational codes appear on the memory lines. At this time, the BFR determines the first AU microsequence state unless the operational code is illegal (Reference paragraph 4.6.4). If the operational code is illegal, a trap occurs. The trap address logic determines the trap address and the interrupt microsequence state initiates the execution of the instruction at the trap location. If this instruction is also illegal, RUNFF is reset and the AU enters the front panel routine.

The second interrupt check occurs following the illegal instruction operation code test when the RUNFF signal is True.

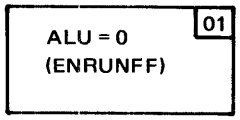
If an interrupt has occurred, the AU enters the interrupt cycle where a check is performed to detect a PIF violation. If the interrupt is a PIF violation, status register bits 5 or 6 (ST5 or ST6) are set and the interrupt is processed as before. If the interrupt is not a PIF violation, the normal interrupt cycle occurs as illustrated in figure 4-12. During each interrupt cycle, the status register bits which enable each type of interrupt are reset for interrupts of the type being processed or of a lower priority.

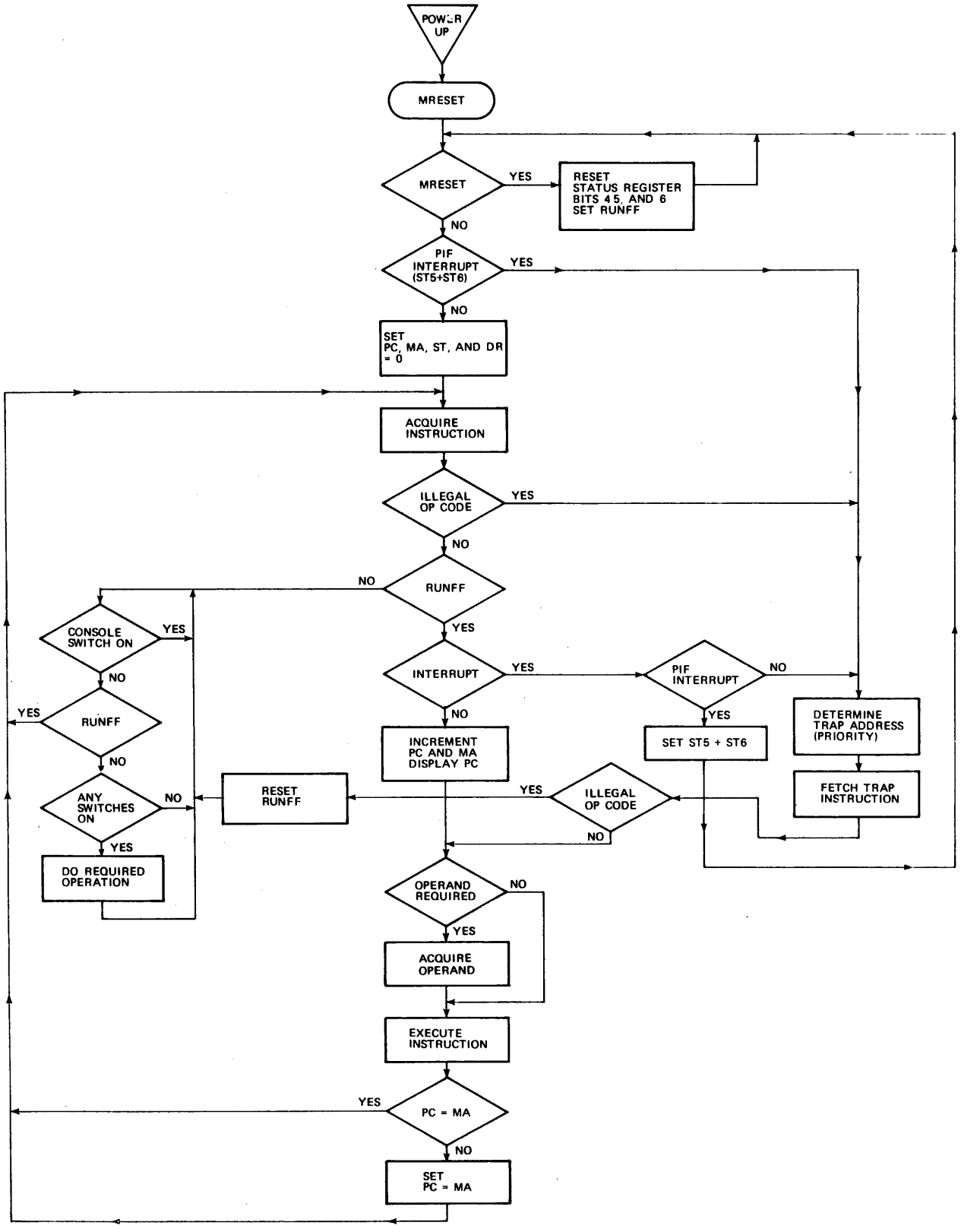
4.7.4 **CONSOLE SEQUENCE.** After every instruction execution, the RUN flip-flop (RUNFF) is tested. If RUNFF has been cleared, the AU enters the control console subroutine. After the console operations are performed and RUNFF is set by the START switch or any other method, the AU returns to the instruction acquisition and execution loop.

4.7.5 **ARITHMETIC UNIT FLOWCHARTS.** Flowcharts for the AU microsequence comprise figure 4-14. These flowcharts contain all information necessary to trace the AU functional sequence. The flowcharts consist of a series of symbols which can be used to verify CPU operation. The following symbol indicates the initial entry point to the flowcharts.



Each rectangular block represents the ROM state indicated in the upper right corner of the block in hexadecimal.

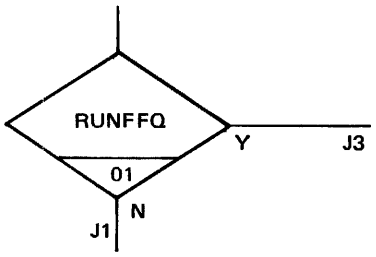




(A) 131725

Figure 4-12. Basic Arithmetic Unit Flowchart

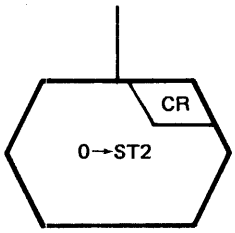
Two types of entries are shown in the ROM state block. The functions which are performed by ROM output bits as listed in table 4-12 are noted in the block (example has ALU = 0). The functions listed in parentheses (ENRUNFF) are performed by other logic resulting from special system conditions. The ROM state address is noted in the upper right corner of the block (01). The decision symbol as illustrated



contains two sections. The upper section indicates the decision signal (RUNFFQ) and the lower section indicates the ROM state in which the decision is made (01). The Y and N indicate Yes and No decisions. The remaining symbols (J1 and J3) indicate the next ROM address (NRA) select signal used for transfer to the next state. There are three exit and entry symbols as noted below:

The symbol at the left indicates an exit and entry to another page in the flowcharts. The upper section of the symbol (1C in example) denotes the page (1) and section of the page (C) for the next part of the flowchart. The section of page symbol, C in this example, is not a vertical or horizontal coordinate, but it does serve to provide a key to locate a specific flowchart. On the page indicated, the same symbol appears at the location where the flowchart continues. The lower section of the symbol (10 in example) is the next ROM state.

Decision dependent logic is represented by the subroutine symbol:



Entries, such as 0--ST2, represent conditional enables that can occur at a particular state depending upon the indicated decisions or conditions. The state indicated, OC for example, shows the state of the conditional enable. This state is identical to that shown in the previous rectangular block because another clock is not required to cause the enable. These symbols do not represent a state transition. Some of the flowchart symbols contain an "XX" instead of the active ROM state or next ROM state. Since these decisions and operations occur numerous times, each operation is illustrated only once and the correct state should be inserted in place of the

"XX" when encountered in the flowcharts. The second symbol shown in the preceding example on the right indicates a transfer to another section of the flowcharts that is controlled by the Basic Function ROM (BFR). The triangular symbol contains either SELECT BFR or SELECT OP depending on whether the BFR logic uses the BFR or the instruction operational codes to generate the next ROM state address (NRA). The next section, which describes the starting points in the flowcharts, will clarify the use of this triangular transfer symbol.

The AU flowcharts which are included at the end of paragraph 4.7.5 have several entry points. The entry point depends basically on the type or location of instruction that is to be acquired from memory. Figure 4-13 indicates the starting points and the state transfers performed by the BFR.

Table 4-11 is used in conjunction with figure 4-13 to provide an instruction versus flowchart page for instruction execution states crosslisting.

- Determine starting point as noted in figure 4-13.
- Start at the specified state (and page) and follow the flowchart until SELECT BFR transfer occurs.
- Consult figure 4-13 to determine how the BFR transfers to the next section of the flowcharts. Follow the flowchart as specified in Figure 4-13.
- If figure 4-13 indicates direct transfer to table 4-11, return to the table and locate the appropriate instruction execution states page number.
- If table 4-11 indicates another state and page, follow the flowchart on that page until a SELECT OP symbol occurs. Then consult table 4-11 for the appropriate instruction execution states page number. Thus, whenever SELECT OP is encountered, table 4-11 can be used to determine the appropriate execution states.

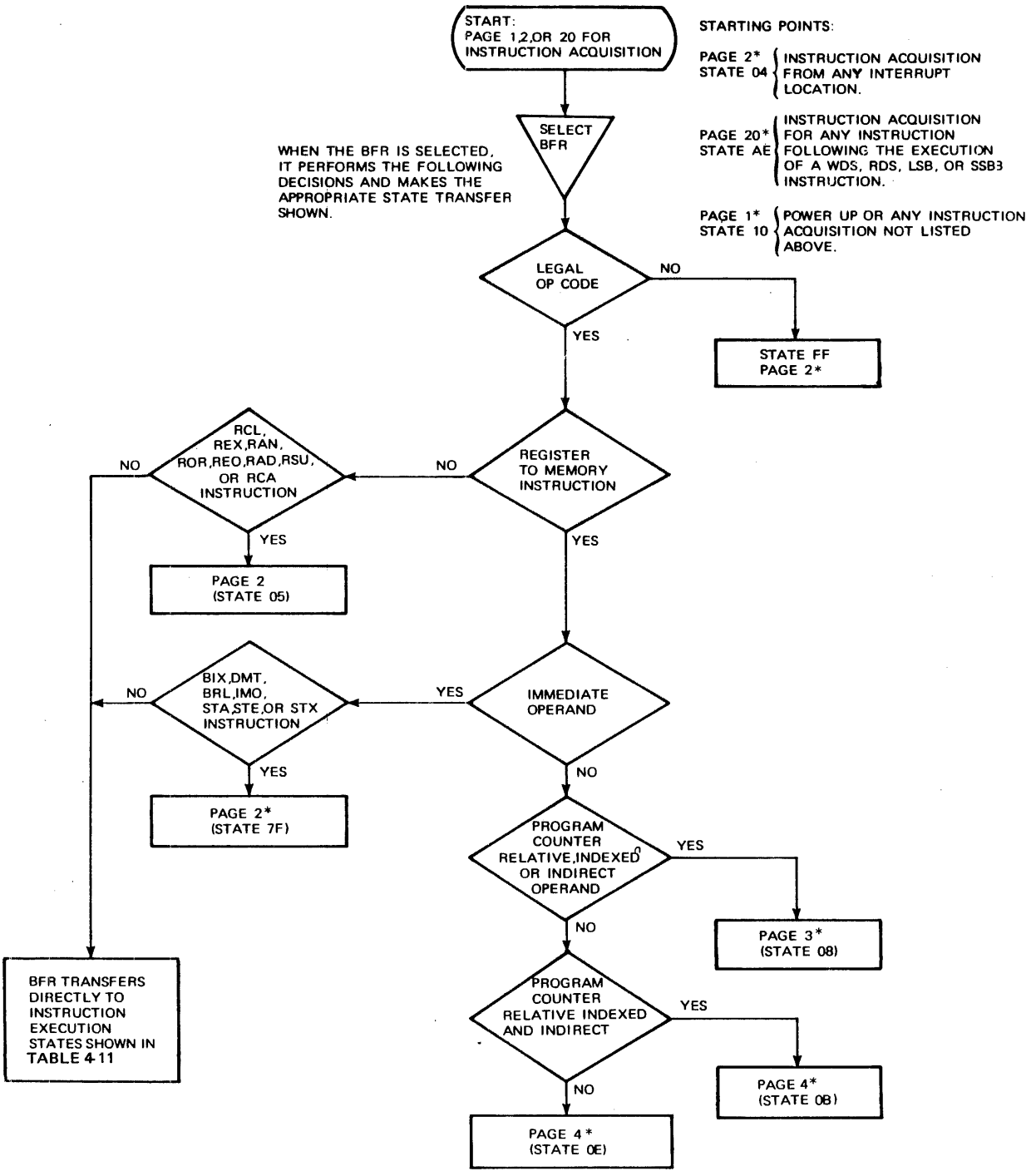
Whenever a CYCOMP decision is shown after a STORE or FETCH operation in the AU flowcharts, it should be noted, the cycle complete signal from the memory controller is not used to initiate the transfer shown. Instead, the memory controller gates the clock off until the requested memory cycle is complete as described in Section I of this manual. Thus, when clocks are not available, the AU will not transfer states until the memory cycle is complete; therefore, the loop waits as illustrated in the flowcharts.

A description of flowchart control signals under ROM control is included in table 4-12. The Model 980A flowcharts (figure 4-14) follow table 4-12.

4.8 CONTROL CONSOLE THEORY

Organization of the control console and the interface between the console and other sections of the computer system are illustrated in figure 4-15. All signals between the console and other sections of the system are routed through the bottom edge connector of the console printed circuit board, the mainframe connector plate wiring, and the bottom edge connector of other system components.

The above referenced figure also shows the major logic blocks within the AU that are involved in communication with the console.



*PAGES AND STATES REFER TO FIGURE 4- 14

(A)131726

Figure 4-13. Flowchart of Basic Function ROM Decisions

Table 4-11. Instruction Execution States Cross Reference

INSTRUCTION	EXECUTION STATES	
	BEGINNING PAGE	BEGINNING STATE
ADD	6	64
ALA*	25	14
ALD	25	52
ALD,C=0	27	50
AND	11	67
AND,IMM	11	27
API	30	C1
ARA*	25	11
ARD	25	51
ARD,C=0	27	50
ATI	30	BE
BIX	14	68
BRL	14	6E
BRU	17	6F
BRU,IMM	17	2F
CLC	31	C7
CLD*	27	1E
CPA	12	6D
CPL	12	6C
CPE,IMM	12	2C
CRA*	26	17
CRB*	26	1D
CRD*	27	1F
CRE*	26	17
CRL*	26	1C
CRM*	24	1A
CRS*	27	1B
CRX*	26	19
DAD	16	77
DAD,IMM	16	37
DIV	9	6B
DLD	15	76
DLD,IMM	15	36
DMT	13	69
DSB	17	75
DSB,IMM	16	35
DST	14	74
IDL	30	C5
IMO	13	6A
IOR	12	66
IOR,IMM	12	26
LDA	5	60
LDE	5	61
LDM	6	63
LDX	5	62

Table 4-11. Instruction Execution States Cross Reference (Continued)

INSTRUCTION	EXECUTION STATES	
	BEGINNING PAGE	BEGINNING STATE
LLA*	26	15
LLD*	26	16
LRA*	25	12
LRD*	25	13
LRF	21	A9
LSB, LSR	20	A0
LTD	28	21
LTZ	28	21
MPY	7	73
MVC	31	C6
NRM	29	25
RAD	19	41
RAN	18	4D
RCA	19	48
RCL	17	4C
RCO	18	47
RDE	19	4E
RDS	22	AB
REO	19	45
REX	18	4F
RIN	19	46
RIV	18	44
RMO	18	4A
ROR	18	49
RSU	19	40
RTO	28	31
RTZ	28	31
SABO	35	FC
SABZ	35	FC
SEQ	20	0F
SEV	20	0F
SGE	20	0F
SGT	20	0F
SLE	20	0F
SLT	20	0F
SMBO	35	F7
SMBZ	35	F7
SMI	20	0F
SNC	20	0F
SNE	20	0F
SNO	20	0F
SNV	20	0F
SNZ	20	0F
SOC	20	0F
SOD	20	0F

Table 4-11. Instruction Execution States Cross Reference (Continued)

INSTRUCTION	EXECUTION STATES	
	BEGINNING PAGE	BEGINNING STATE
SOO	20	0F
SOV	20	0F
SPL	20	OR
SRF	21	A7
SSB	20	A3
SSE	20	0F
SSN	20	0F
STA	11	70
STE	11	71
STX	11	72
SUB	6	65
SZE	20	0F
TABO	35	FC
TABZ	35	FC
TMBO	35	F7
TMBZ	35	F7
WDS	22	AB
All Shifts(*), C=0 except ALD, ARD, LTD, LTZ, RTO, RTZ	27	10
LTO, LTZ, C=0	28	20
RTO, RTZ, C=0	28	30
Front Panel	36	DO

The console does not contain control logic other than circuits for switch contact debouncing signals and synchronizing these signals to the system clock. Console operations control is accomplished within the AU. When a switch is actuated to initiate a console operation, the synchronized switch signal is detected by the ROM controller in the AU. The ROM controller then executes data transfers to complete the operation and enter an idle state. The ROM controller remains in this state until the switch is released.

Logic implementation and location of the console components are illustrated in TI Drawing 960742 and 960740, respectively. Console component reference designators that are referenced in the discussions which follow are those used on the TI Logic Drawing 960742 found in the Electrical Drawings manual. Logically and functionally the Control Consoles for the 980A and 980B are identical. The mechanical configurations are different. The assembly drawings of the control consoles are 960740 where -0001 references the 980A configuration and -0002 references the 980B configuration.

Table 4-12. Flowchart Control Signal Description

FLOWCHART ENTRY*	CONTROL DESCRIPTION	ROM OUTPUT SIGNALS				
	DATA SELECTED TO BUS A:	R1,	R2-,	R3-,	R4-,	
AR→A	A Register	0	0	0	0	
ER A	E Register	0	0	0	1	
XR→A	X Register	0	0	1	0	
MR→A	M Register	0	0	1	1	
SR→A	S Register	0	1	0	0	
LR→A	L Register	0	1	0	1	
BR→A	B Register	0	1	1	0	
PC→A	Program Counter	0	1	1	1	
ST→A	Status	1	0	0	0	
$\overline{DP}$→A	I/O Data (Negative)	1	0	0	1	
SETBIT→A	Setbit Data	1	0	1	0	
TRAP→A	Trap Address, Bits 8–14	1	0	1	1	
PNL→A	Panel Data	1	1	0	0	
MEM→A	Memory Data	1	1	0	1	
IR→A	Instruction Register	1	1	1	0	
PCS→A	Program Counter Save Register	1	1	1	1	
	DATA SELECTED TO BUS B:		R5,	R6-		
MAR→B	Memory Address Register		0	1		
UR→B	Utility Register		0	0		
MDR→B	Memory Data Register		1	1		
SD→B	Signed Displacement (IR 8–15, Sign Extented)		1	0		
	ALU CARRY IN:		R7,	R8		
CYIN=0	Carry In = 0		0	0		
CYIN=1	Carry In = 1		0	1		
CYIN=ST3	Carry In = ST3		1	0		
	Unused		1	1		
	ALU DATA SELECTED WITH CARRY IN (SHOWN WITH CYIN=0 ARITH. OPERATIONS):	R9-,	R10-,	R11-,	R12-,	R13-,
ALU = A	Bus A	1	1	1	1	1
ALU = A+B	Bus A OR'ed with Bus B	1	1	1	0	1
ALU = A+$\overline{B}$	Bus A OR'ed with Inuerted Bus B	1	1	0	1	1
ALU = -1	Minus 1 (2's complement)	1	1	0	0	1
ALU = A PLUS $\overline{AB}$	Bus A Plus Inverted and of A and B	1	0	1	1	1
ALU = (A+B) PLUS $\overline{AB}$	A OR'ed with B Plus Inverted and of A and B	1	0	1	0	1
ALU = A-B-1	A minus B minus 1	1	0	0	1	1
ALU = A$\overline{B}$-1	A and Inverted B minus 1	1	0	0	0	1
ALU = A PLUS AB	A Plus and of A and B	0	1	1	1	1
ALU = A PLUS B	A Plus B	0	1	1	0	1
ALU = (A+$\overline{B}$) PLUS AB	A OR'ed with Inverted B Plus and of A and B	0	1	0	1	1
ALU = AB-1	A and B minus 1	0	1	0	0	1

Table 4-12. Flowchart Control Signal Description (Continued)

FLOWCHART ENTRY*	CONTROL DESCRIPTION	ROM OUTPUT SIGNALS				
		R9–	R10–	R11–	R12–	R13–
ALU = A PLUS A	A Plus A Shifted Right	0	0	1	1	1
ALU = (A+B) PLUS A	A OR B Plus A	0	0	1	0	1
ALU = (A+$\overline{B}$) PLUS A	A OR Inverted B Plus A	0	0	0	1	1
ALU = A-1	A minus 1	0	0	0	0	1
	NO CARRY IN (LOGICAL OPERATIONS)	R9–	R10–	R11–	R12–	R13–
ALU = $\overline{A}$	Inverted A	1	1	1	1	0
ALU = $\overline{A+B}$	Inverted OR of A and B (NOR)	1	1	1	0	0
ALU = $\overline{A}$+B	Inverted A or B	1	1	0	1	0
ALU = 0	ZERO	1	1	0	0	0
ALU = $\overline{AB}$	Inverted and of A and B (NAND)	1	0	1	1	0
ALU = $\overline{B}$	Inverted B	1	0	1	0	0
ALU = A⊕B	Exclusive or of A and B	1	0	0	1	0
ALU = A$\overline{B}$	A and Iverted B	1	0	0	0	0
ALU = $\overline{A}$+B	Inverted A or B	0	1	1	1	0
ALU = $\overline{A⊕B}$	Inverted A exclusive– or B	0	1	1	0	0
ALU = B	B	0	1	0	1	0
ALU = AB	A and B	0	1	0	0	0
ALU = 1	ONE	0	0	1	1	0
ALU = A+$\overline{B}$	A or Inverted B	0	0	1	0	0
ALU = A+B	A or B	0	0	0	1	0
ALU = A	A	0	0	0	0	0
	SHIFT SELECTORS:		R14	R15		
	RIGHT SHIFT C0 INPUT:					
ALU0→RSC0	ALU Bit 0		0	0		
0→RSC0	ZERO		0	1		
ALU15→RSC0	ALU Bit 15		1	0		
ER15→RSC0	ER Bit 15		1	1		
	LEFT SHIFT C15 INPUT:					
0→LSC15	ZERO		0	0		
ER1→LSC15	ER Bit 1		0	1		
ER0→LSC15	ER Bit 0		1	0		
	Zero		1	1		
	LEFT SHIFT C0 INPUT		R53			
ALU1→LSC0	ALU Bit 1		0			
ALU0→LSC0	ALU Bit 0		1			
	DATA SELECTED TO BUS C		R16	R17		
ALU→C	ALU Bits 0-15		1	1		
RSHIFT→C	RSC0, ALU0-14		1	0		
LSHIFT→C	LSC0, ALU2-15, LSC15		0	1		

Table 4-12. Flowchart Control Signal Description (Continued)

FLOWCHART ENTRY*	CONTROL DESCRIPTION	ROM OUTPUT SIGNALS		
		R16	R17	
CIR8→C	ALU8-15, ALU0-7	0	0	
	OUTPUT ENABLES:	R18-		
ENALUB	Enable ALUB0-15 to memory	0		
	Disable ALUB0-15	1		
		R19-		
ENIOUT	Enable I/O Bus Out	0		
	Disable I/O Bus	1		
	MEMORY CYCLE CONTROL:			
	CYCLE REQUEST:	R20		
CPURC	Cycle Requested	1		
	No Action	0		
	STORE FETCH:	R21-		
STORE	Store	0		
FETCH	Fetch	1		
	INSTRUCTION ACQUISTION	R22		
IAQ	Aquire Instruction	1		
	No Action	0		
	STATUS REGISTER ENABLE:	R23,	R24,	R25,
	No Action	0	0	0
ENASOV,ENST2	Enable Add, Subtract Overflow and ST2	0	0	1
ENTCOV, ENST2	Enable Two's Complement Overflow and ST2	0	1	0
ENLSOV, ENST2	Enable Left Shift Overflow and ST2	0	1	1
ENMPOV, ENST2	Enable Multiply Overflow and ST2	1	0	0
ENDVOV, ENST2	Enable divide Overflow and ST2	1	0	1
ENARITH, ENST0 ENST1		1	1	0
ENLOGIC, ENST0, ENST1		1	1	1
			R26-	
ENST3	Enable Status Register Bit 3		0	
	No Action		1	
			R27-	
ENSTLOAD	Enable Status Register Load for All Bits		0	
	No Action		1	
	E Register Enable:	R28-,	R29-	
ENERRIGHT	ER Right Shift	0	1	
ENERLEFT	ER Left Shift	1	0	

Table 4-12. Flowchart Control Signal Description (Continued)

FLOWCHART ENTRY*	CONTROL DESCRIPTION	ROM OUTPUT SIGNALS	
		R28-,	R29-
ENERLOAD	ER Load	0	0
	No Action	1	1
			R30
ENER0LOAD	ER Bit 0 Load:		1
	No Action		0
	E REGISTER BIT 0 INPUT:	R31,	R32-
ALU0→ER0	ALU Bit 0	0	1
ALU15→ER0	ALU Bit 15	0	0
ER1→ER0	ER Bit 1	1	1
ALU14→ER0	ALU Bit 14	1	0
	E REGISTER BIT 1 INPUT:		R54
ER0→ER1	ER Bit 0		0
A15	BUS A Bit 15		1
	E REGISTER BIT 15 INPUT:	R33-,	R34
0→ER15	0	1	1
1→ER15	1	1	0
UR0→ER15	UR Bit 0 (Contains ALU0)	0	1
SR16 or AR15→ER15	SR16 or AR16 (State Dependant)	0	0
	REGISTER INCREMENTS:		
ENMAINC	Memory Address Register		R35
ENPCINC	Program Counter		R36-
ENXRINC	X Register		R37
SCINC	Shift Counter		R39
	REGISTER LOADS :		
ENARLOAD	A Register with Bus C		R40-
ENXRLOAD	X Register with Bus C		R41-
ENMRLOAD	M Register with Bus C		R38-
ENSRLOAD	S Register with Bus C		R48-
ENLRLOAD	L Register with Bus C		R55-
ENPCLOAD	Program Counter with ALU Data		R42-
ENVRLOAD	Utility Register with Bus C		R43
ENMDLOAD	Memory Data Register with Memory Data		R49
ENDRLOAD	Display Register with IOUT Data		R44-
ENMALOAD	Memory Address Register with ALU Data		R45-

Table 4-12. Flowchart Control Signal Description (Continued)

FLOWCHART ENTRY*	CONTROL DESCRIPTION	ROM OUTPUT SIGNALS
	SOURCE-DESTINATION CONTROL:	
ENSOURCE→A	Enable Source Register to Bus A	R46
ENDEST→A	Enable Destination Register to Bus A	R47
ENSOURCE LOAD	Enable Source Register Load	R49
ENDEST LOAD	Enable Destination Register Load	R50
	REGISTER - MEMORY CONTROL	
NOT IN	Register Memory without Special	
FLOW	Immediate State	R51
CHARTS	Register Memory with Special	
	Immediate	R52-
	SPARE	R56
	NEXT ROM ADDRESS JUMPFIELD:	
J0	Jump Field 0	R57-R60, R61-R64
J1	Jump Field 1	R57-R60, R65-R68
J2	Jump Field 2	R57-R60, R69-R72
J3	Jump Field 3	R65-R68, R69-R72

*Flowchart Entries excluded from this table are generated by logic which is not ROM controlled.

4.8.1 **PANEL LOCK FUNCTION.** A LOCK signal, from the PANEL switch mounted on the chassis front panel is routed through the connector plate wiring to a flip-flop in the arithmetic unit. This flip-flop is clocked with the system clock to generate a synchronized signal (PNLENQ) which is True when the PANEL switch is in the UNLOCK position. This signal (PNLENQ) is applied to the control console where it is used to inhibit all control signals from the console to the AU when the PANEL switch is in the LOCK position. The PNLENQ signal is also used on the AU board to prevent the run flip-flop from clearing when the MODE switch is in the HALT position and the system is running while the console is locked.

4.8.2 **CONSOLE SWITCH DEBOUNCE AND SYNCHRONIZATION.** Debounce and synchronization logic is illustrated in TI Drawing Number 960742.

4.8.2.1 **Function Switches.** Signals from the function switches (S5, S9, or S11 through S23) are enabled to the AU by a delay and synchronization circuit consisting of transistor Q1; gates Z2, Z1, Z7, and Z12: and flip-flops in network Z4. When any of these switches are activated the input current from the pertinent AU decision logic gate and the related pullup resistor causes Q1 to generate a delay pulse from a circuit formed with Z1, Z2, and two resistor-capacitor filters. The output pulse as illustrated in the switch timing diagram in figure 4-16 is synchronized to the system clock by the first flip-flop in network Z4. The flip-flop output performs two functions. The first function is to enable an open-collector inverter (Z12) to pull the common line to the switches (S5, S9, and S11 through S23) to ground. This grounding action causes the decision

logic in the AU to sense a switch activation. The second function is to gate the flip-flop signal (FSWON) to the AU that causes the AU ROM controller to remain in state D0 after completing the operation initiated by the switch activation.

The START switch (S10) is also connected to this same pulse circuit. However, in this configuration the second flip-flop in Z4 is used to generate a delayed 1-clock time signal (STRTSW) that is supplied to the AU. The STRTSW signal sets the RUNFF and the AU ROM controller enters the instruction acquisition microsequence.

4.8.2.2 **Breakpoint and Clock Switches.** The Breakpoint (BKPT) and CLOCK switches also utilize debounce circuits as illustrated in TI Drawing Number 960742. These circuits provide a delayed bounce-free signal as the switches are activated.

NOTE

The CLOCK switch function is inhibited if the clock switch jumper is connected to E80 instead of E81. If this jumper is connected to E80 the OFF and STEP positions of the CLOCK switch are inoperative.

The flip-flop circuit which uses Z15 and gate Z1 is a pulsing circuit that is designed to provide a 1-clock time pulse when the CLOCK switch is activated in the STEP position.

4.8.2.3 **Sense Switches.** The four sense switches have individual latches that are forced to the correct state when these switches are activated. The latches are implemented with cross-coupled NAND gates (SN7400N). The latch outputs are synchronized and stored in a 4-bit storage register (SN74175N). The storage register outputs are used by the AU decision logic for the sense switch instructions.

4.8.2.4 **Mode Switch.** The MODE switch is a two position switch that provides ground directly to the AU logic. Since the RUN or SIE positions do not cause AU action until the START switch is activated. RUNSW and SIESW signal filtering is not required before being sent to the AU logic.

4.8.3 **CONTROL OPERATIONS.** Control operations that are initiated from the console include starting and halting program execution, single instruction execution, breakpoint operation, bootstrap loader operation, and single clock operation. All other console operations involve the display of data, or the entry of data switch settings into the system. Refer to TI schematic diagrams 960742, figure 4-15 and the AU flowcharts in figure 4-13 sheets 36 to 40 for details for the Control Console operations.

4.8.3.1 **Program Start and Halt.** When the system is halted, the AU ROM controller enters the D0 state of the console control subroutine. In state D0, a test is performed to detect the activation of console data entry or display switches as indicated by the FSWON signal. If a switch is actuated, the controller idles in state D0 until the switch is released and FSWON returns to ZERO.

From state D0, the controller enters a loop (states D1 through D9) in which all console control switch signals are tested. When all switches are deactivated, the controller continually cycles through this loop.

The condition of the RUN flip-flop in the AU is tested in state D1. When the system is started, the RUNFF signal becomes True and when the controller enters state D1, it transfers to state E0 from which interrupt processing or instruction acquisition begins. When the controller leaves the console operation subroutine, none of the console switches except the MODE and Clock switches can effect system operation.

When the console MODE switch is placed in the HALT position, the AU run flip-flop clears and the controller transfers to state D0 as previously described.

4.8.3.2 **Reset.** The console RESET switch function is designed to clear MAR, PC, ST, and DR. This function is accomplished when the RESET switch signal (RSTSW) is detected in state D1. The controller enters state E2 where the ALU is set to an all ZERO condition to enable the MAR, PC, ST, and DR. When the next system clock occurs, these registers are loaded with zero's and the ROM controller returns to state D0 where it remains until the RESET switch is released.

4.8.3.3 **Single Instruction Execution.** To execute only one instruction, the console MODE switch is placed in the SIE position and the START switch is activated. This causes the RUNFF to set, the AU microsequence to leave the control console subroutine, and an instruction to be acquired from the memory location in the Program Counter (PC). As the BFR selects the first state for instruction execution, the RUNFF is reset causing the AU to re-enter the console subroutine after executing the instruction. (Reference paragraph 4.6.4.) The next program instruction can be executed by activating the START switch.

4.8.3.4 **Clock Control.** System clock signals are generated on the memory controller board and are controlled by the clock logic in the AU. This logic utilizes two signals from the control console. If the clock jumper is placed in the correct location (no jumper between location E80 and E81, reference TI Drawing Number 960742), the clock signal (CLKON) is generated only when the CLOCK switch on the console is in the ON position. If the jumper is connected between location E80 and E81, CLKON is generated regardless of the CLOCK switch position.

When CLKON disables the clock logic in the AU, the AU microsequence halts and will not continue until further clock pulses are provided by placing the CLOCK switch in the ON or STEP positions. In the STEP position, the control console logic generates a 1-clock time pulse (STEPULS–) that is used in the AU clock logic to advance the AU one state. This 1-clock time pulse also appears on the clock line to the DMAC, APP, and I/O ports. The memory controller continues to refresh the MOS memory boards while the clocks are off. The controller also executes a memory cycle if a STEP clock advances the AU into a ROM state calling for a memory cycle.

4.8.3.5 **Data Display and Indicators.** The control console Data Register (DR) receives buffered ALU data through the wire-wrapped backplane. The DR is implemented with 4-bit registers (SN74175) and is loaded with gated ALU data (1OUT0-15) by the clock enable signal (ENDRCK–) when the CLOCK switch is ON. When the CLOCK switch is off, the DR is continuously loaded with ALU data.

The 16-light emitting diode (LED) display is driven directly with the DR outputs using 160-ohm current limiting resistors. The remaining LED displays are: breakpoint (BKPT) indicator, POWER LOSS indicator, RUN indicator, and IDLE indicator. These LED displays are also driven with logic gates and limiting resistors.

4.4.3.6 **Data Entry and Display.** The function switches that are described in paragraph 4.8.2.1 are designed to allow data from the control console to enter the AU and memory for display by the Display Register (DR). When the display function is actuated, the control console microsequence causes the AU to enable the proper register through the ALU for display by the DR. The AU flowcharts (figure 4-13) indicates this action in the control console subroutine beginning with state D0.

When the function switches are moved to the ENTER position, the AU microsequence enables control console data to enter the correct register. The register load signal is also enabled at this time. The data switches S24 through S39 (labeled 0 through 15 on the control console) have center poles that are driven by open collector inverters that can disable the data switches during bootstrap loader ROM operation. The inverters normally remain at a logic zero allowing the data switch setting to determine the logic state of the console data output lines (PNL0-15).

The MD and MDI switches allow the data switch settings to be loaded into memory. The MDI switch causes the same action as the MD switch. The MDI switch also conditions the memory address (MA) register to permit successive memory locations to be displayed or loaded without reloading the MA.

4.8.3.7 **Bootstrap Loader ROM.** The control console contains four ROM networks (Z16, Z17, Z18, and Z19) that contain bootstrap loader programs allowing programs to be loaded by several types of peripheral devices. The loader programs are written in the first 256 memory locations when the LOAD switch is actuated. The LOAD switch signal LDBTSW– causes the AU microsequence to clear the loader address register that is in the console logic (Z5 and Z6) and disable the data switches by removing the logic zero signal from the center poles of the data switch. With the data switches disabled, the loader ROMs are enabled and the ROM outputs are stored in successive memory locations as the loader address register is incremented from 0 to 255. At this point, the loader address register generates a signal (LRAMAX) that causes the microsequence to reenter the console subroutine. If the MODE switch is set to RUN, the system begins to execute instructions from the location specified by the PC, normally zero.

4.8.3.8 **Breakpoint.** The breakpoint feature allows the operator to determine when any memory location is addressed and also halts the system if necessary. When the Breakpoint (BKPT) switch is in the CLR position, no action occurs. If the switch is placed in the center position, the BKPT indicator is illuminated when the AU addresses the memory location specified by the data switches. If the BKPT switch is in the up (BKPT) position, the AU halts with the indicator illuminated. The AU contains a 16-bit comparator (4-SN7485N) that continually compares console data (PNL0-15) from the data switches with the contents of the CPU memory address register (MAR). When the BKPT switch is in the center position and the comparator senses equal MAR and PNL data, the breakpoint flip-flop (BKPTQ) is set in the AU. The BKPTQ flip-flop provides a buffered signal to the Control Console which lights the BKPT indicator. If the BKPT switch is in the upper position, the indicator is illuminated and RUNFF is reset by the same signal which sets the BKPTQ flip-flop. Once the machine halts, the machine is restarted with the START switch which sets RUNFF. When the panel LOCK signal is True, the center position and BKPT position of the breakpoint switch have identical functions. The indicator illuminates when the addresses match, but a Halt does not occur.

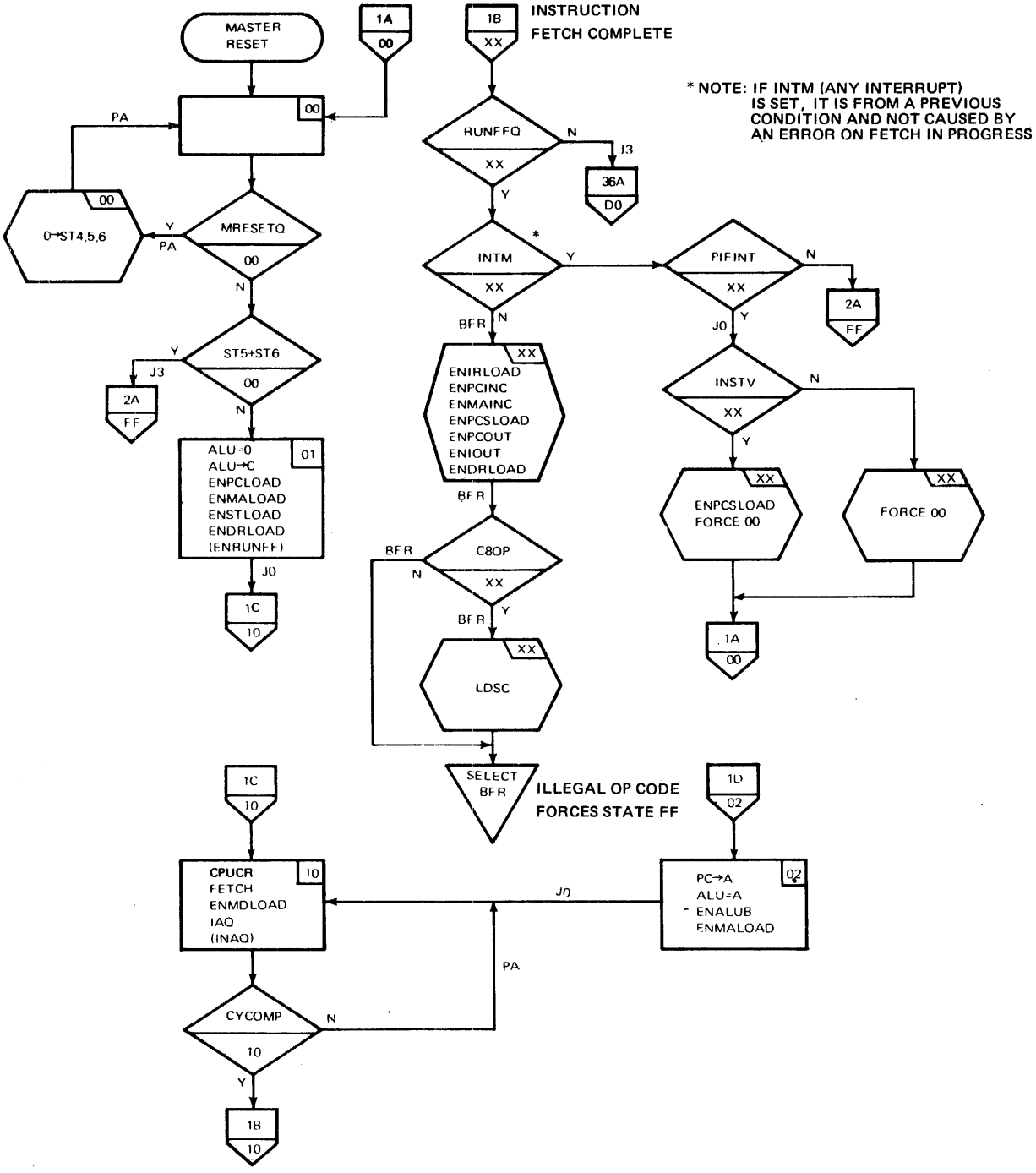


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 1 of 40)

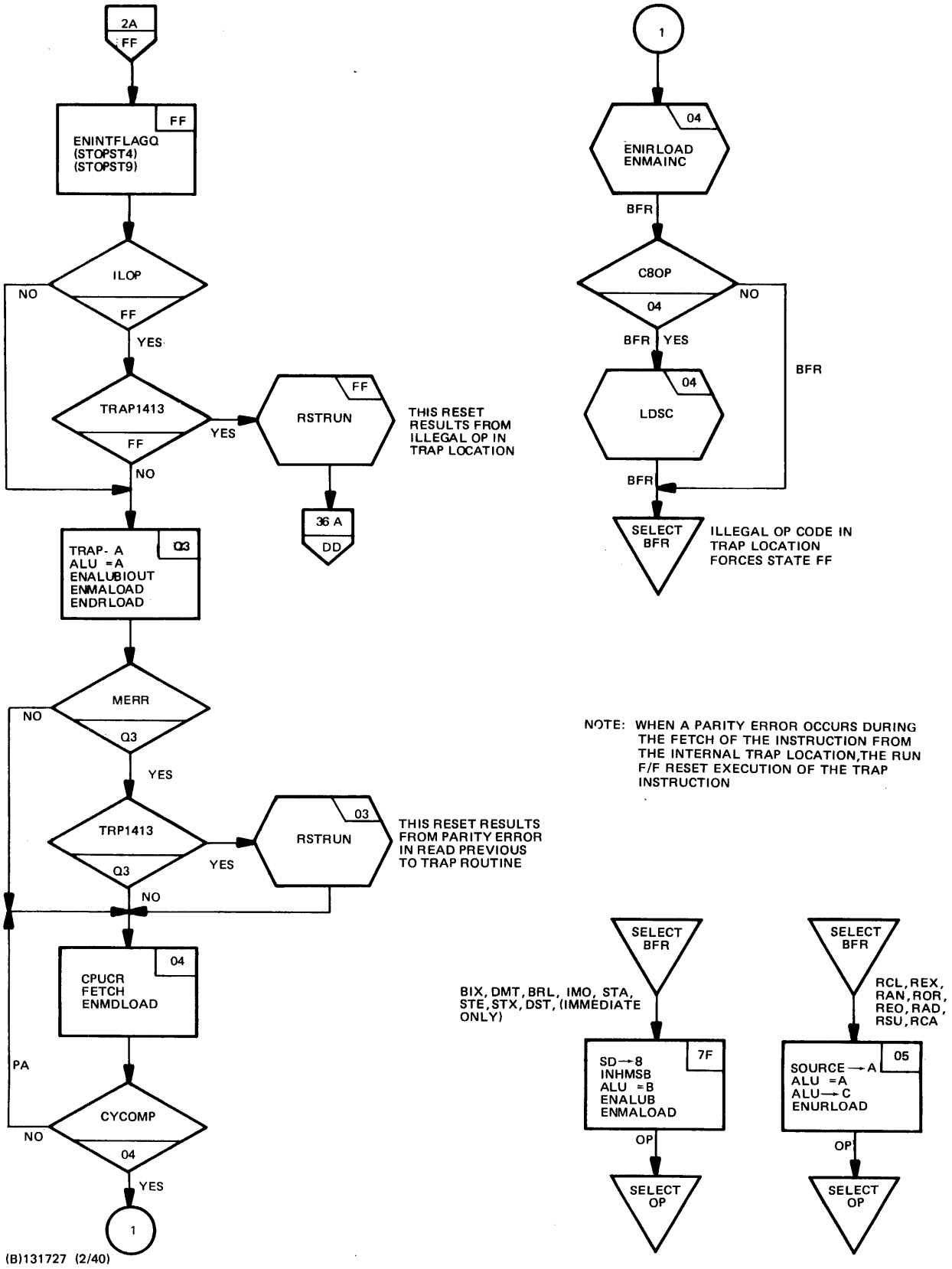
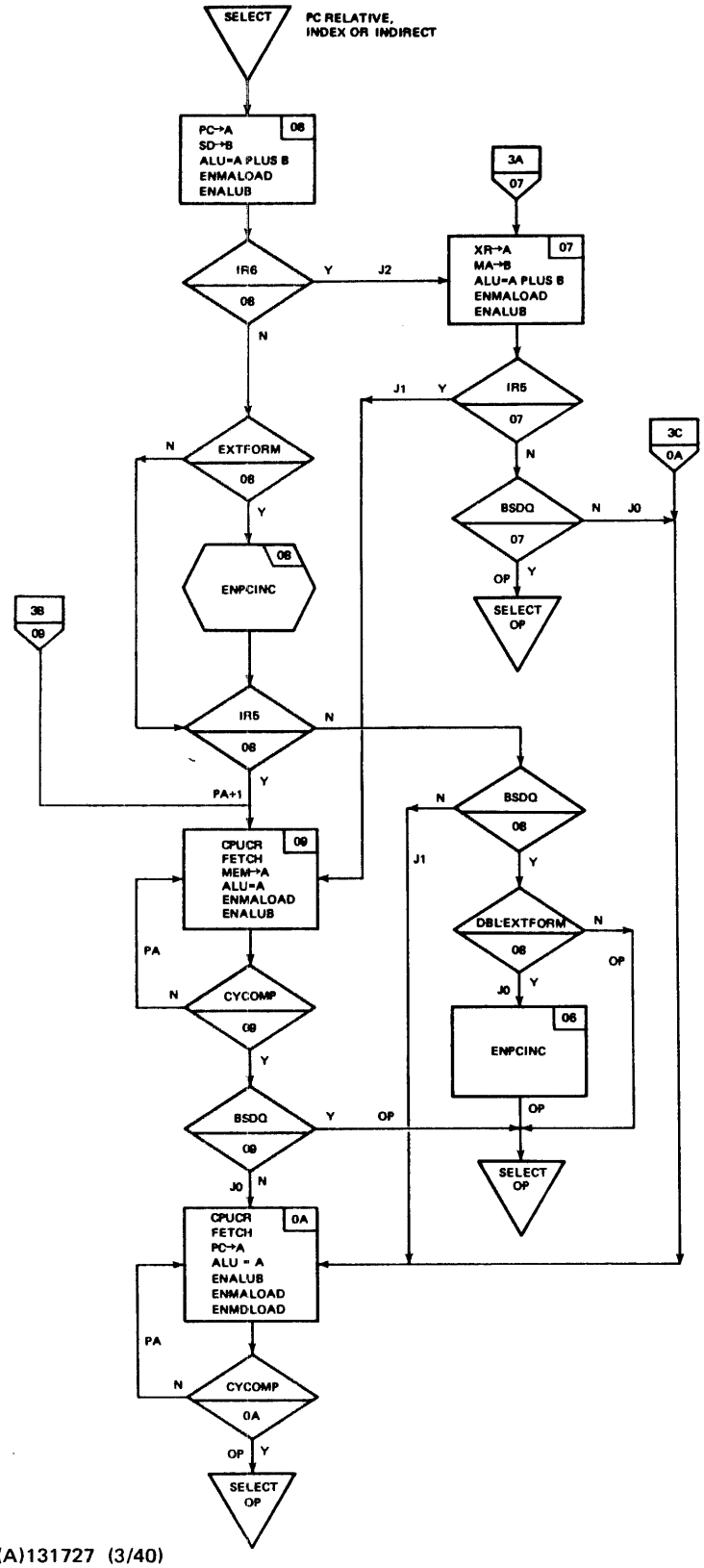


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 2 of 40)



(A)131727 (3/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 3 of 40)

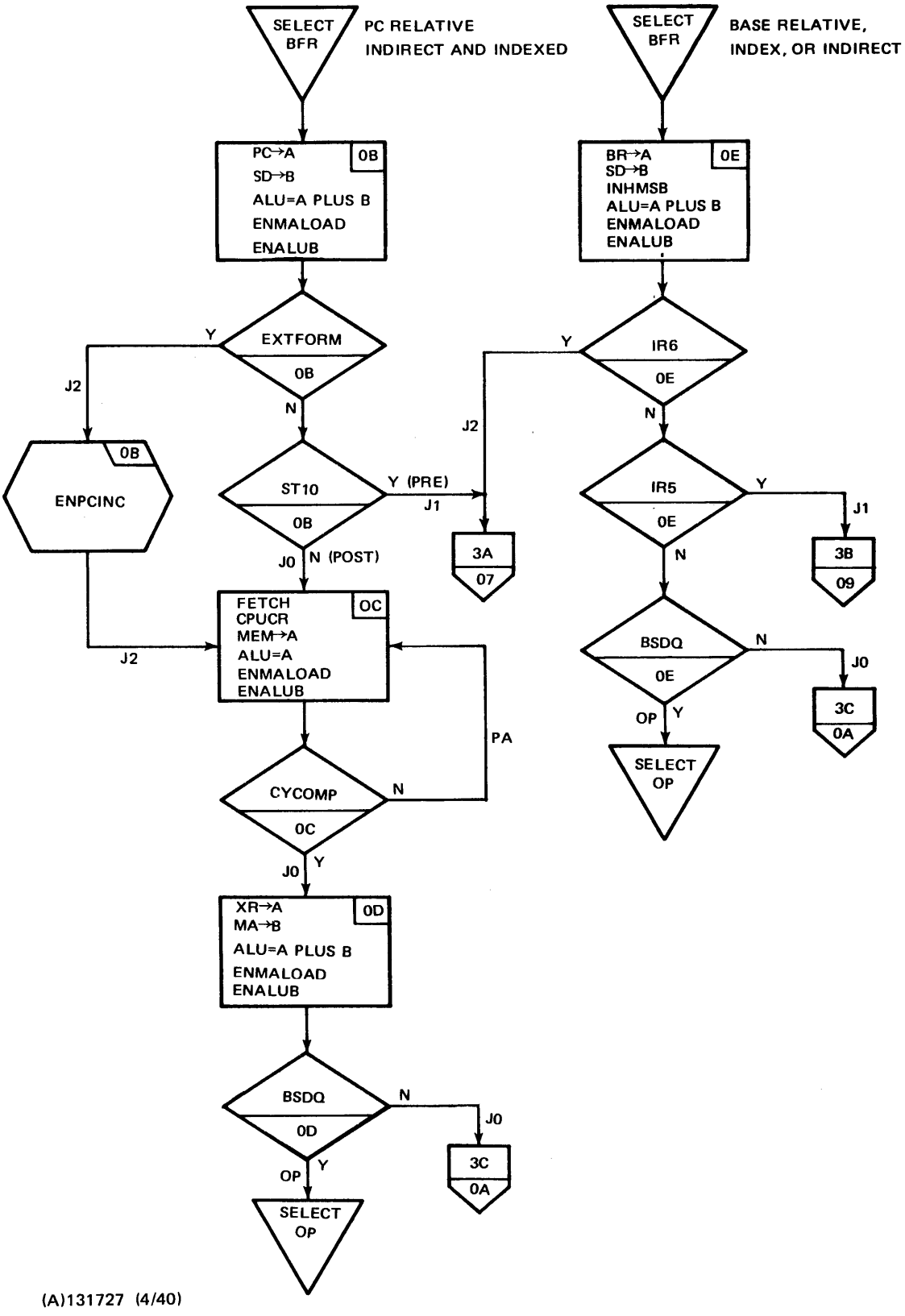
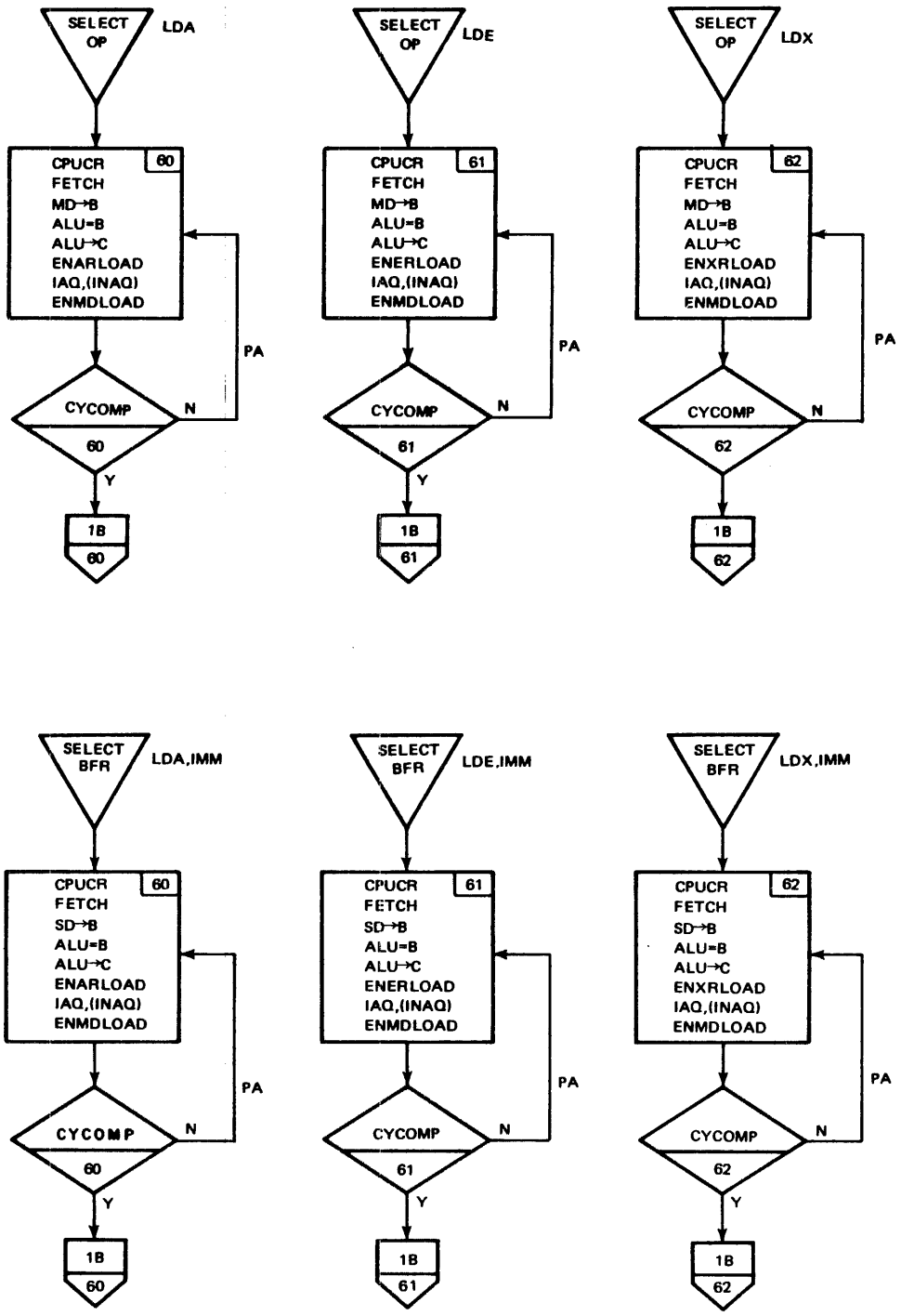


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 4 of 40)



(A)131727 (5/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 5 of 40)

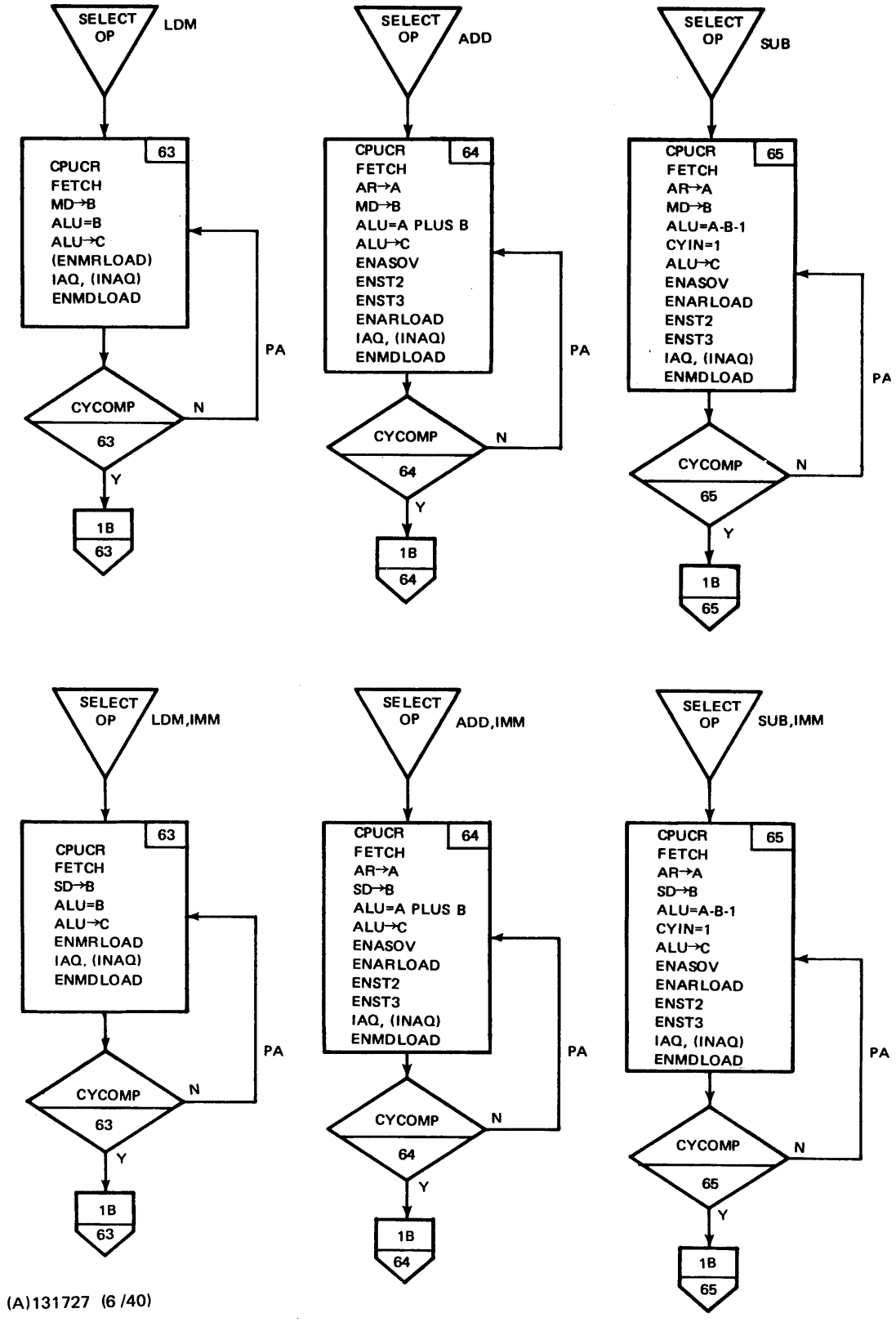
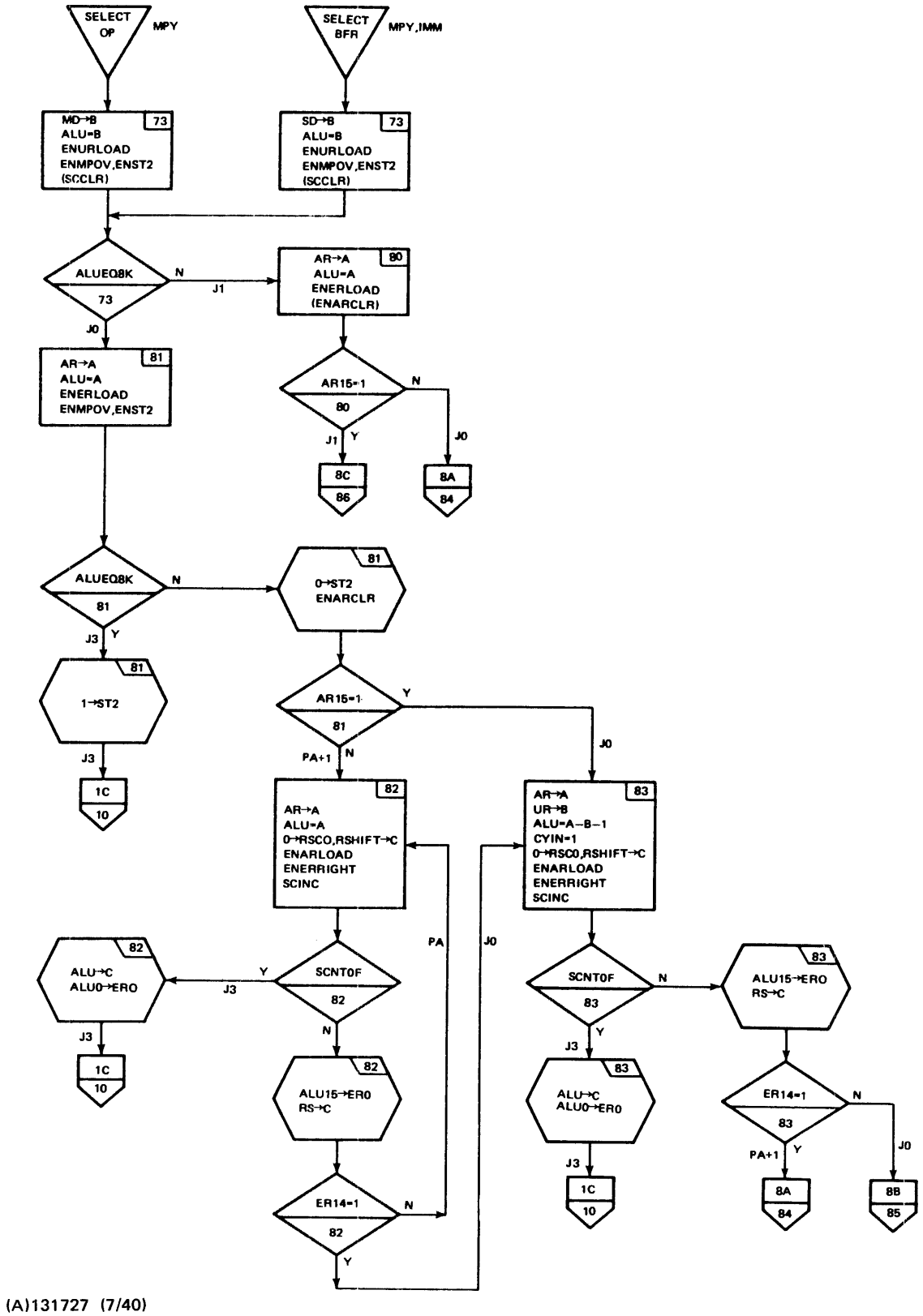
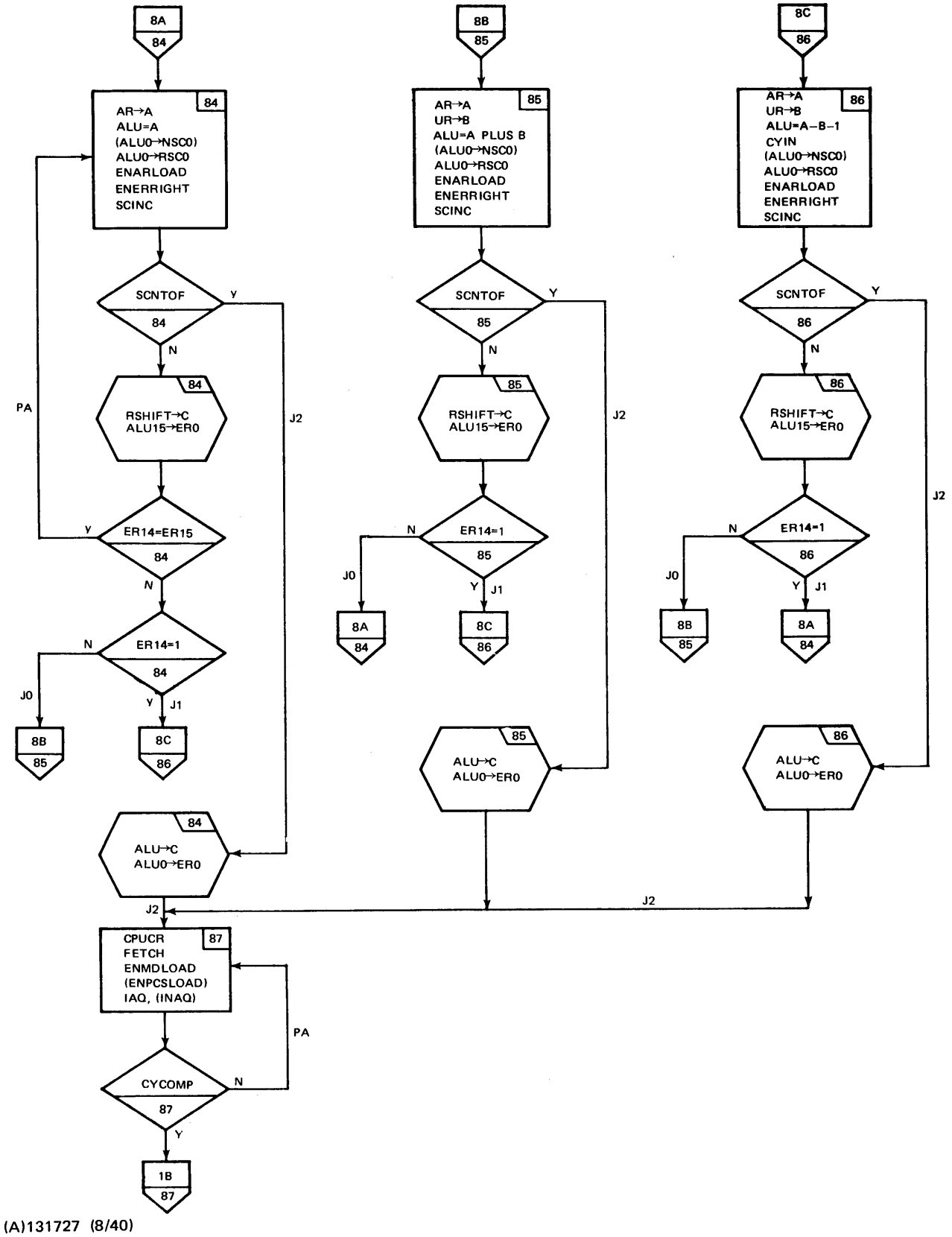


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 6 of 40)



(A)131727 (7/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 7 of 40)



(A)131727 (8/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 8 of 40)

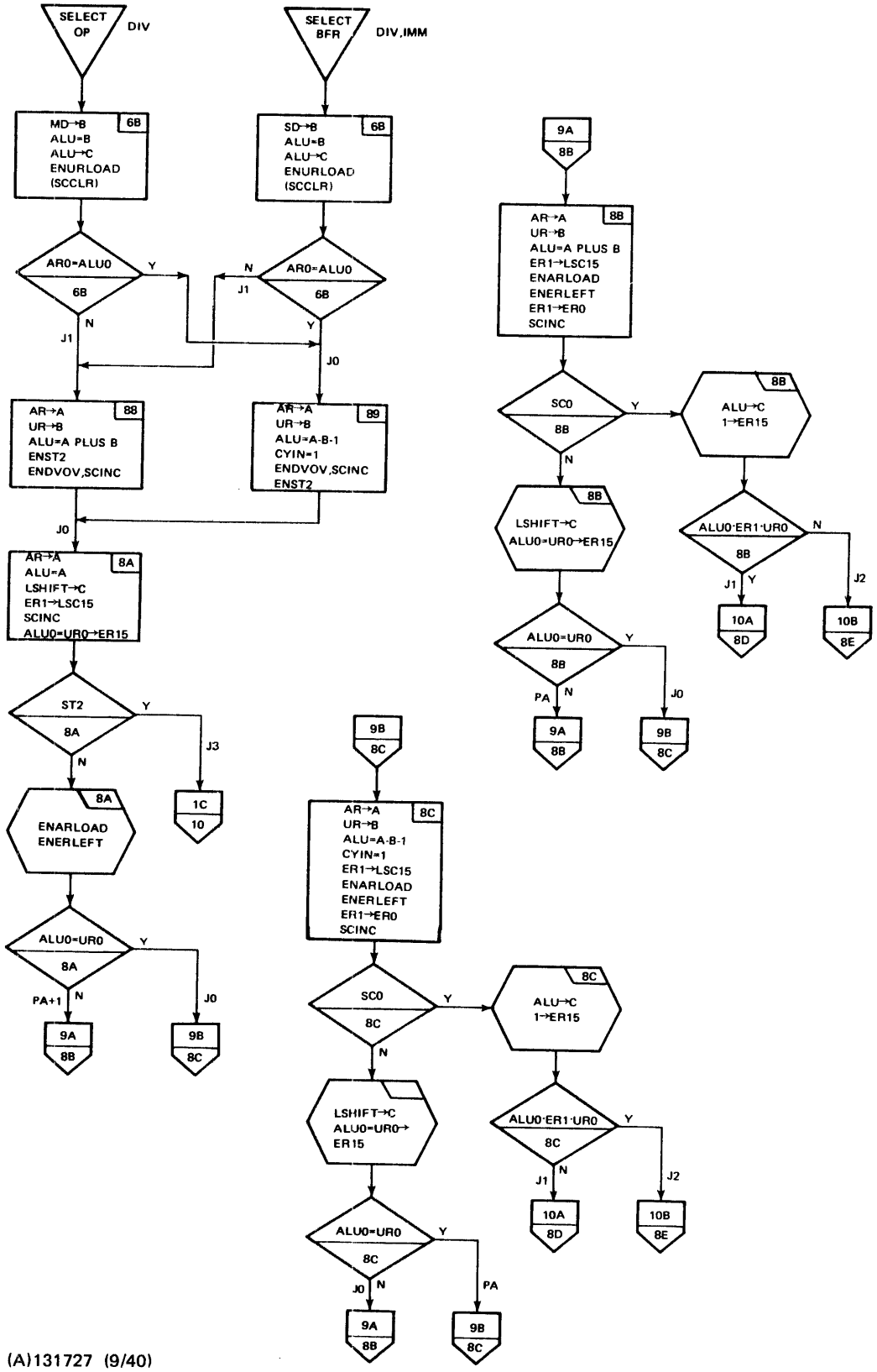
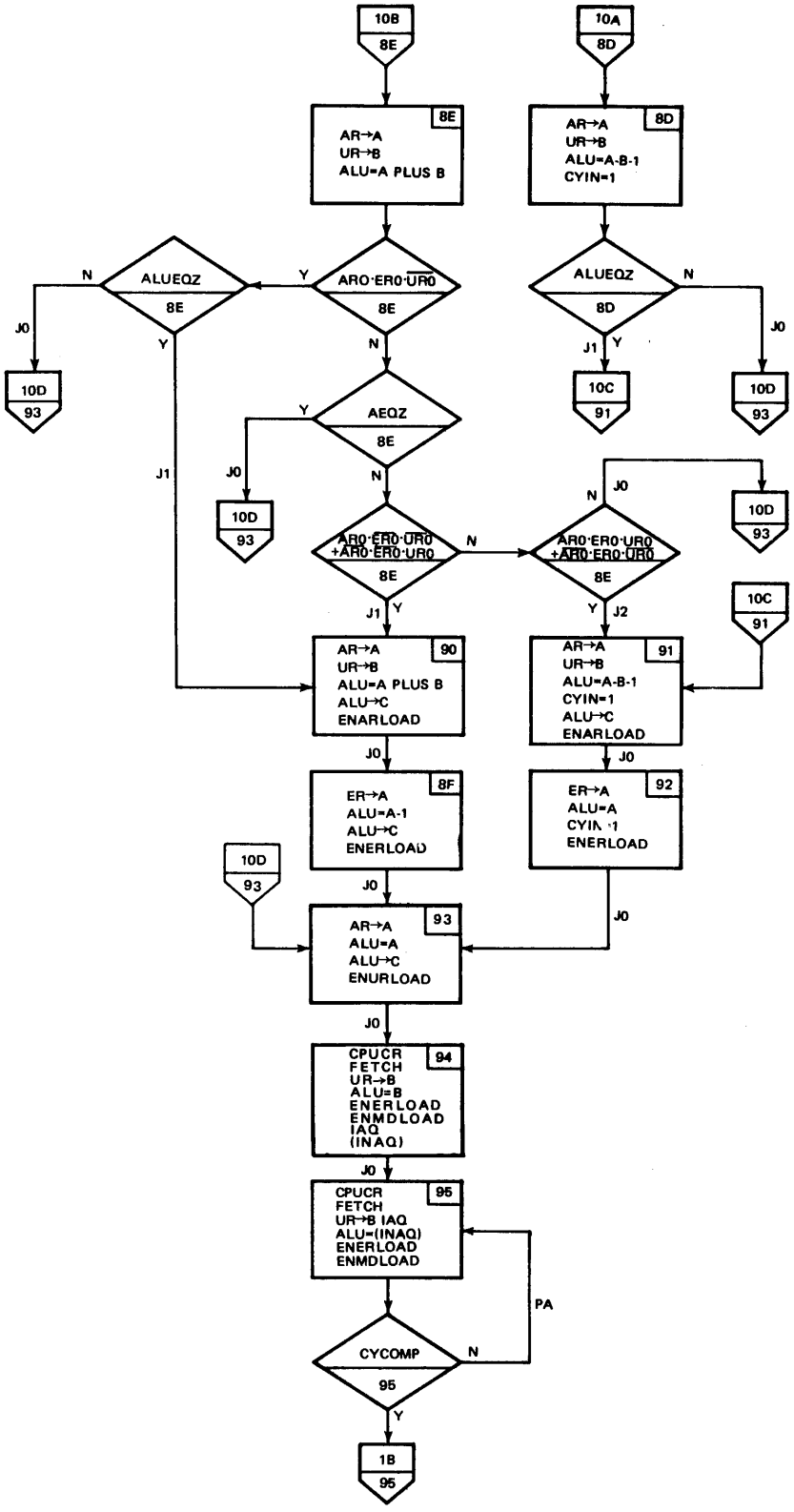
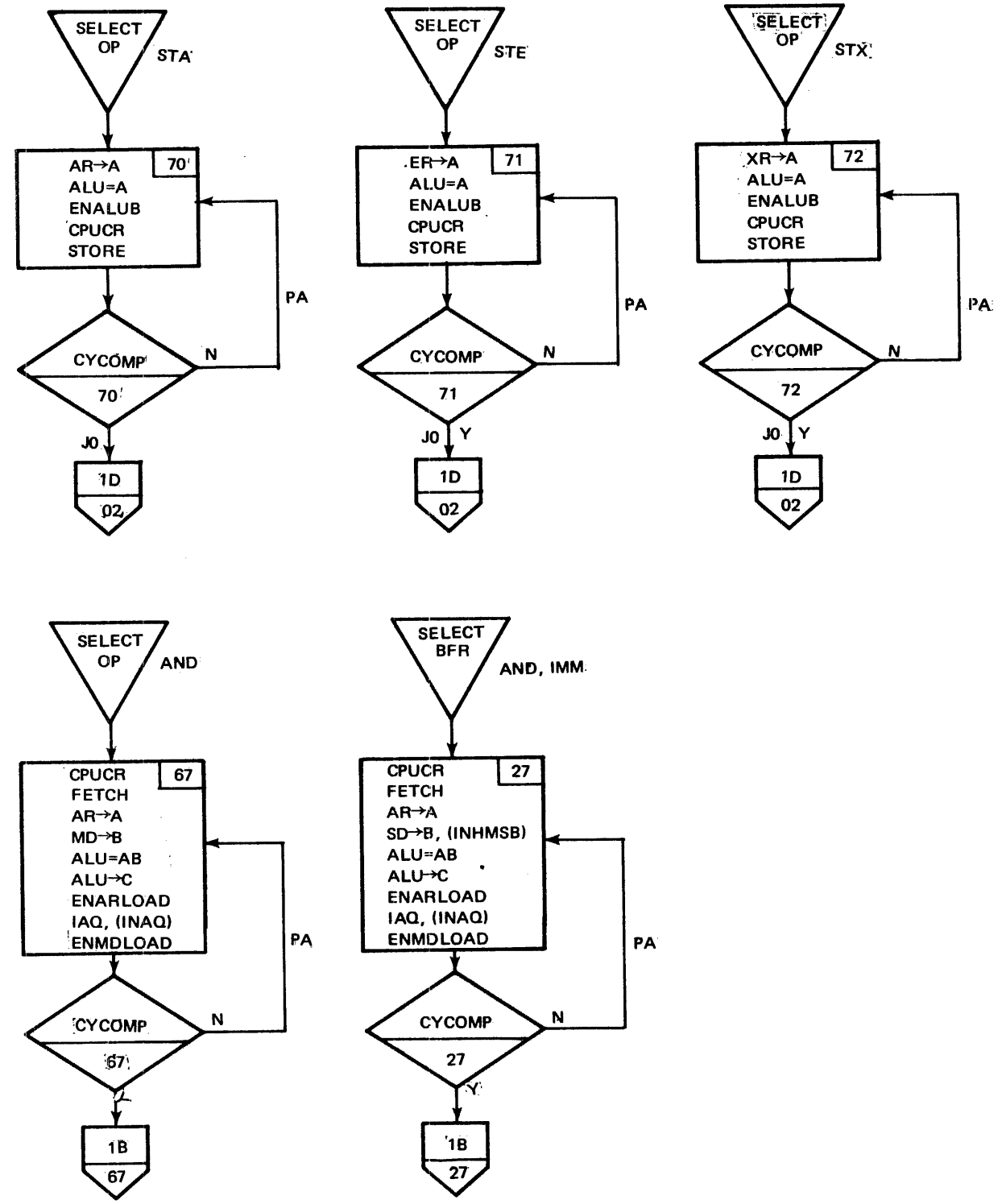


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 9 of 40)



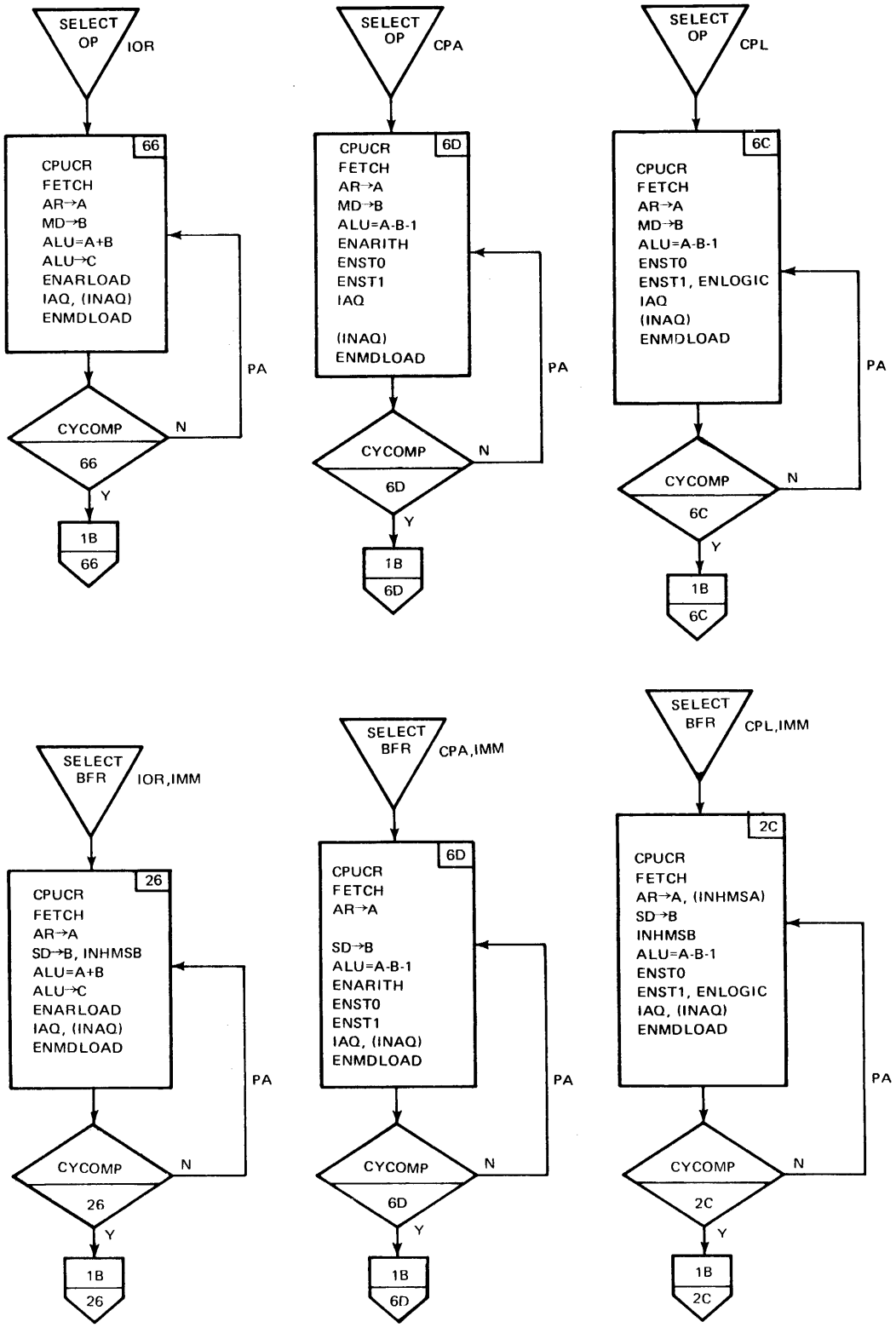
(A)131727 (10/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 10 of 40)



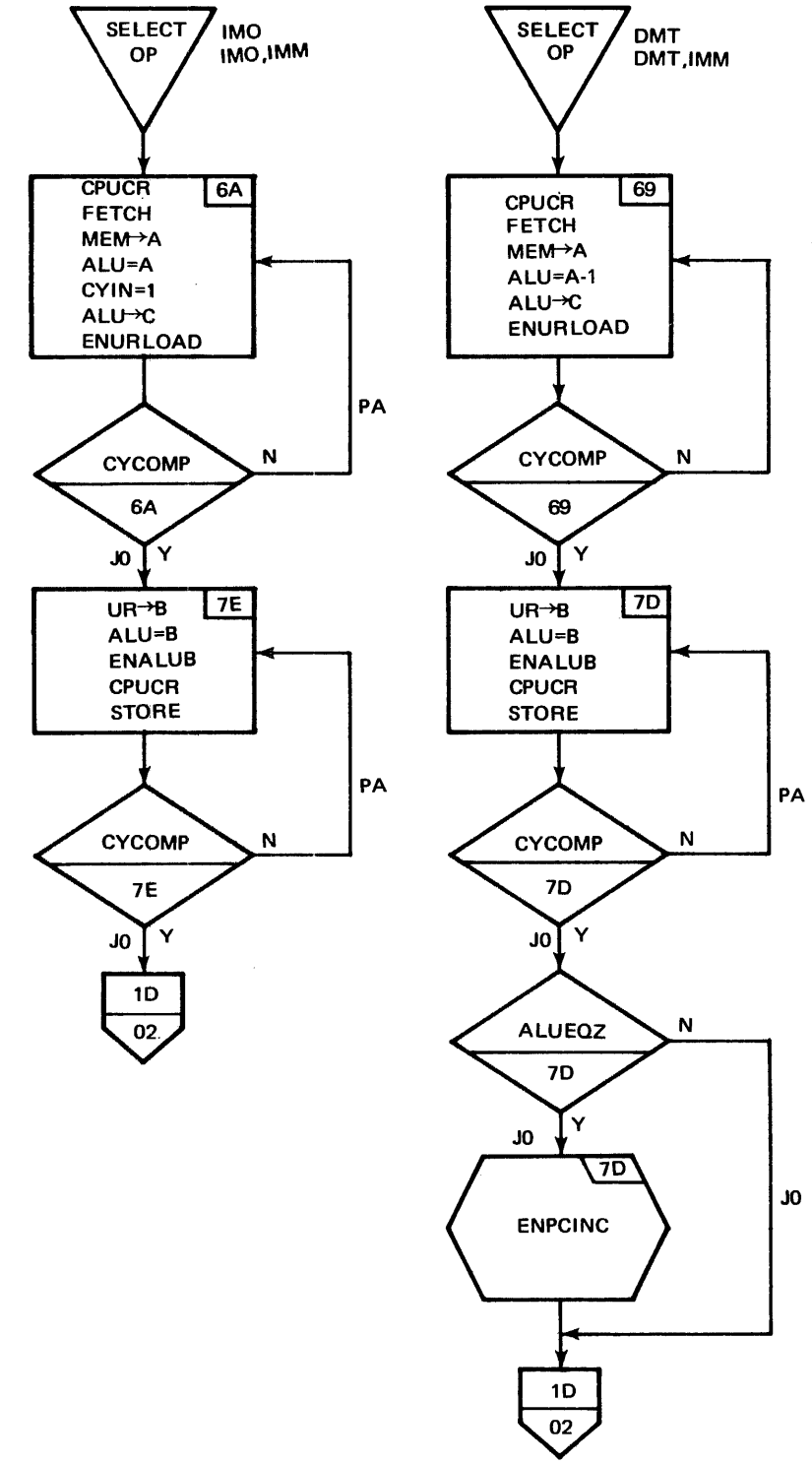
(A)131727 (11/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 11 of 40)



(A)131727 (12/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 12 of 40)



(A)131727 (13/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 13 of 40)

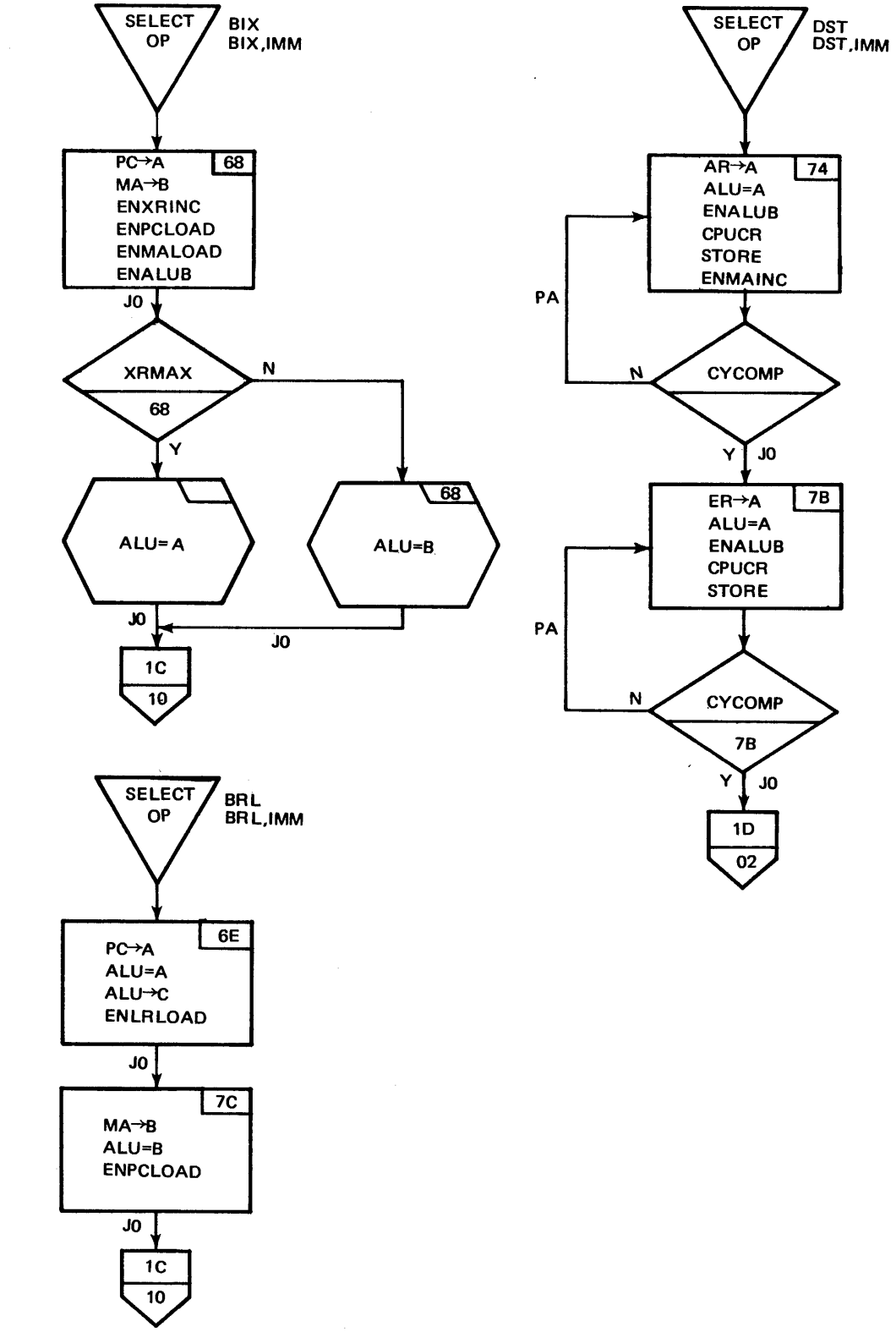
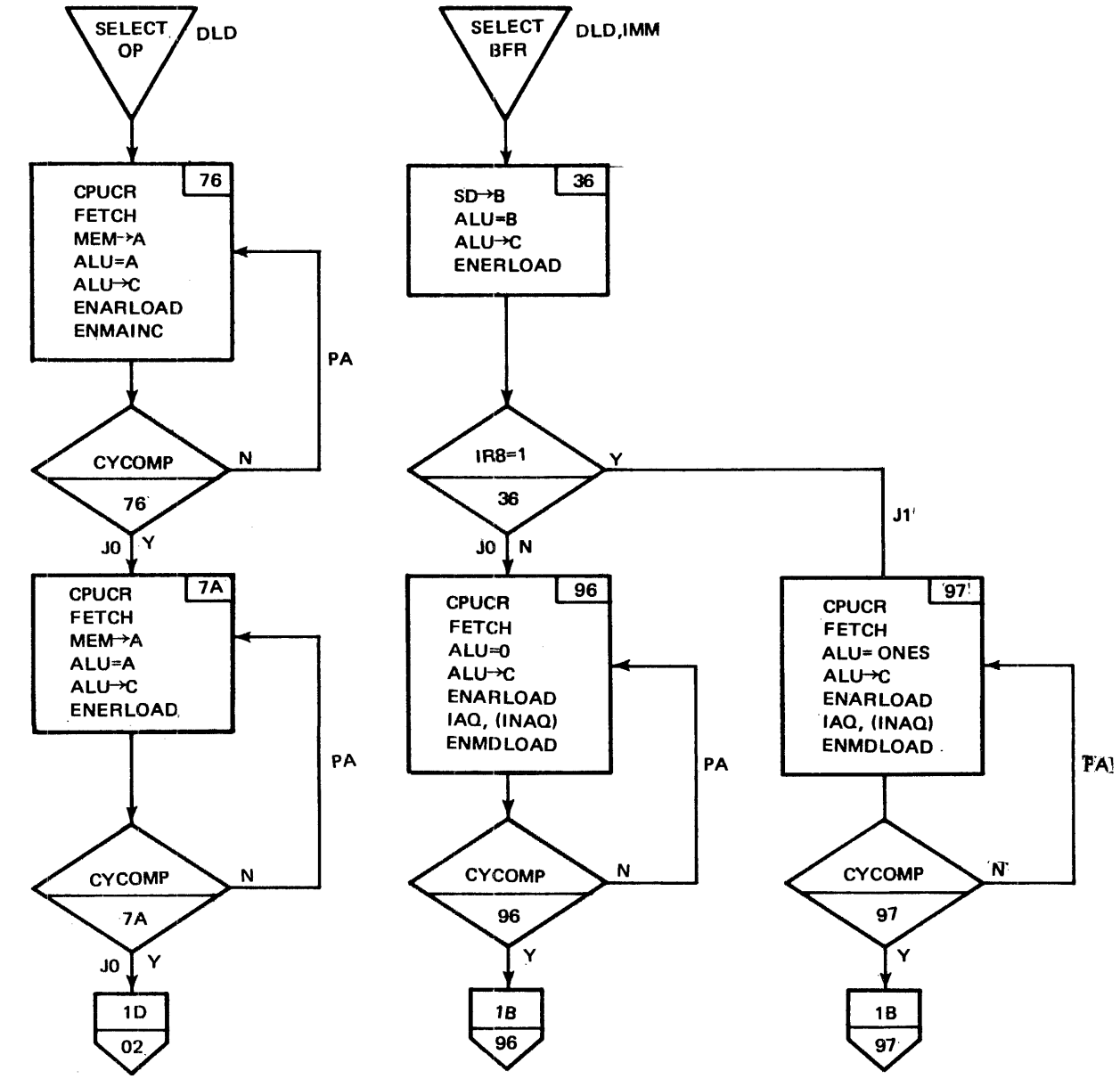
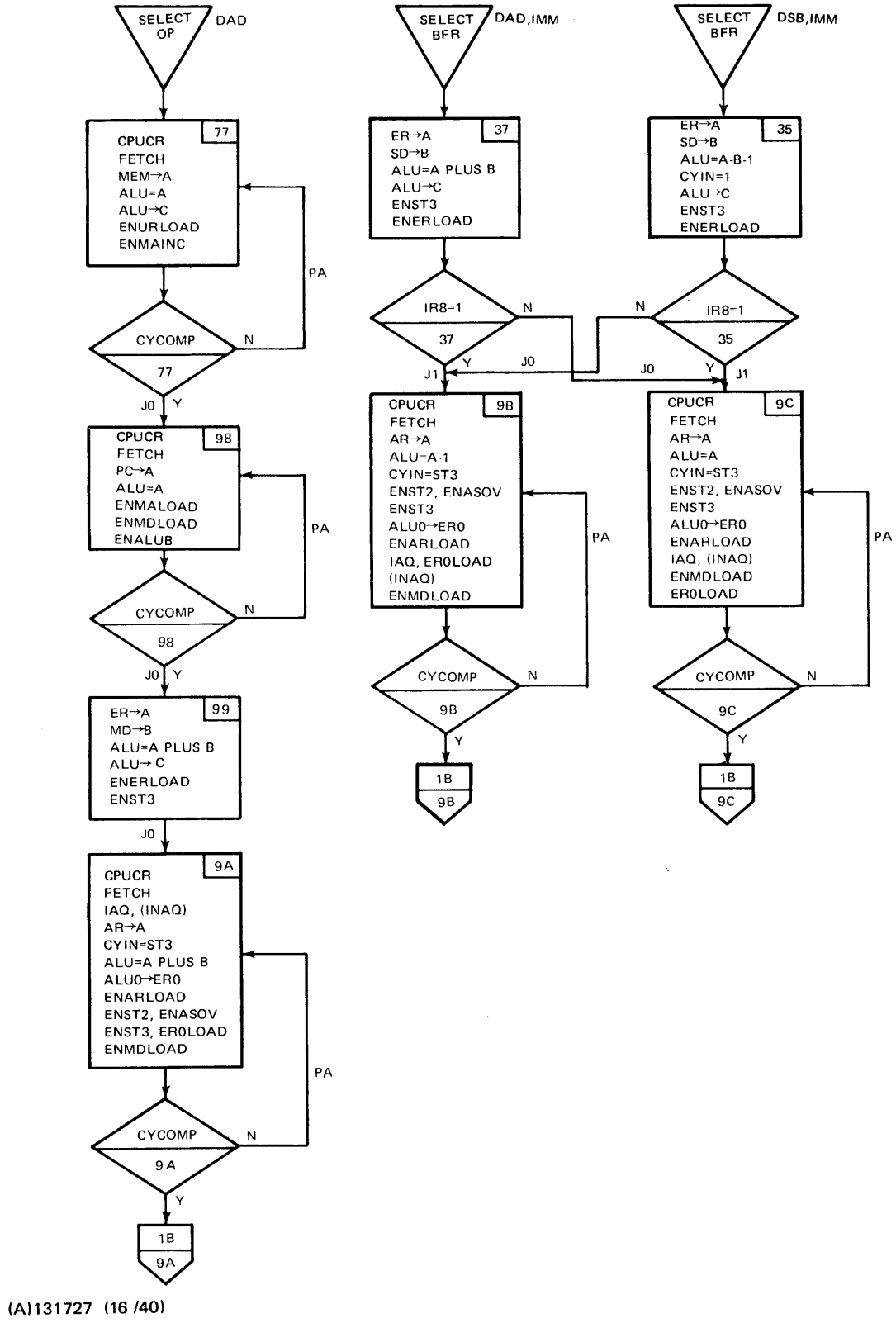


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 14 of 40)



(A)131727 (15/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 15 of 40)



(A)131727 (16 /40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 16 of 40)

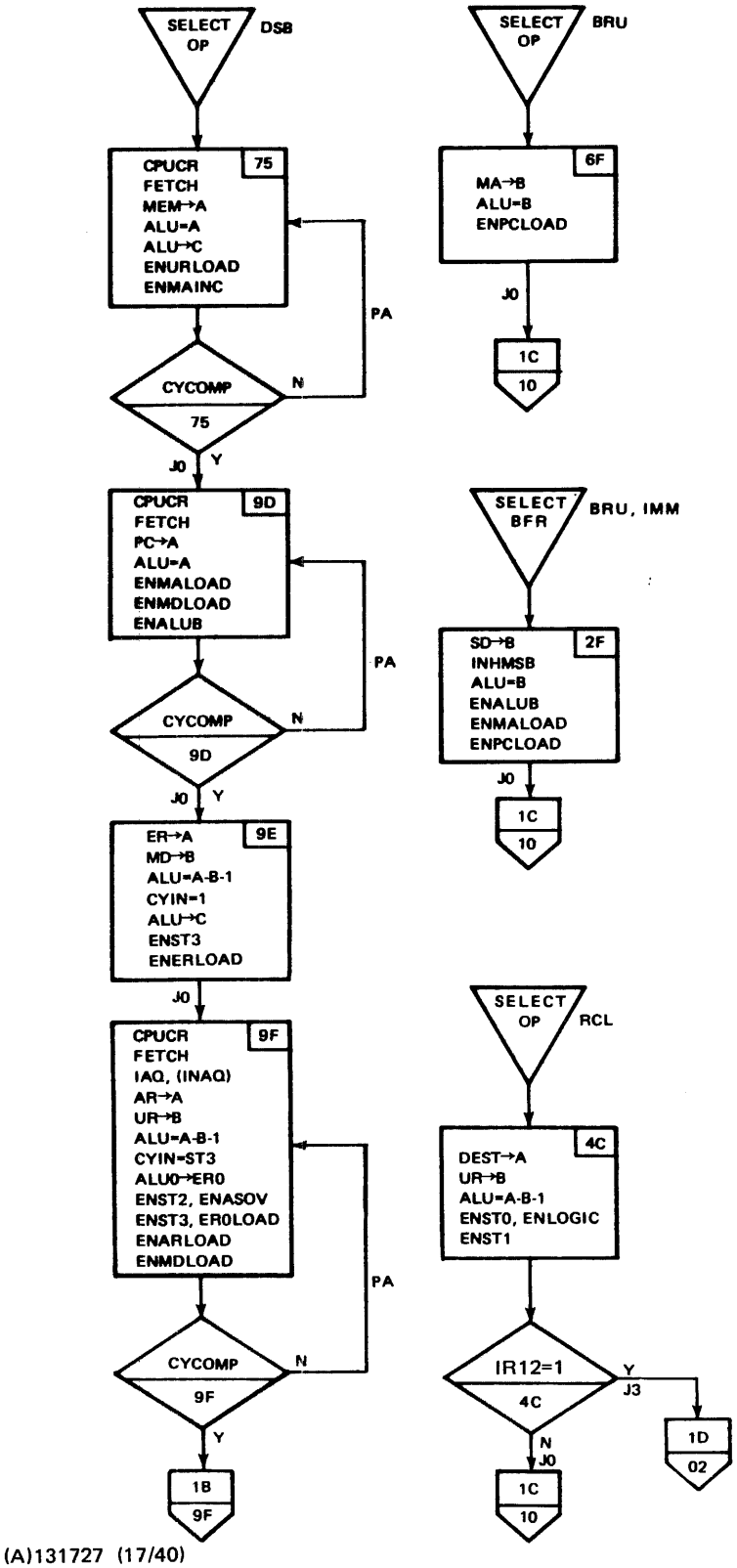
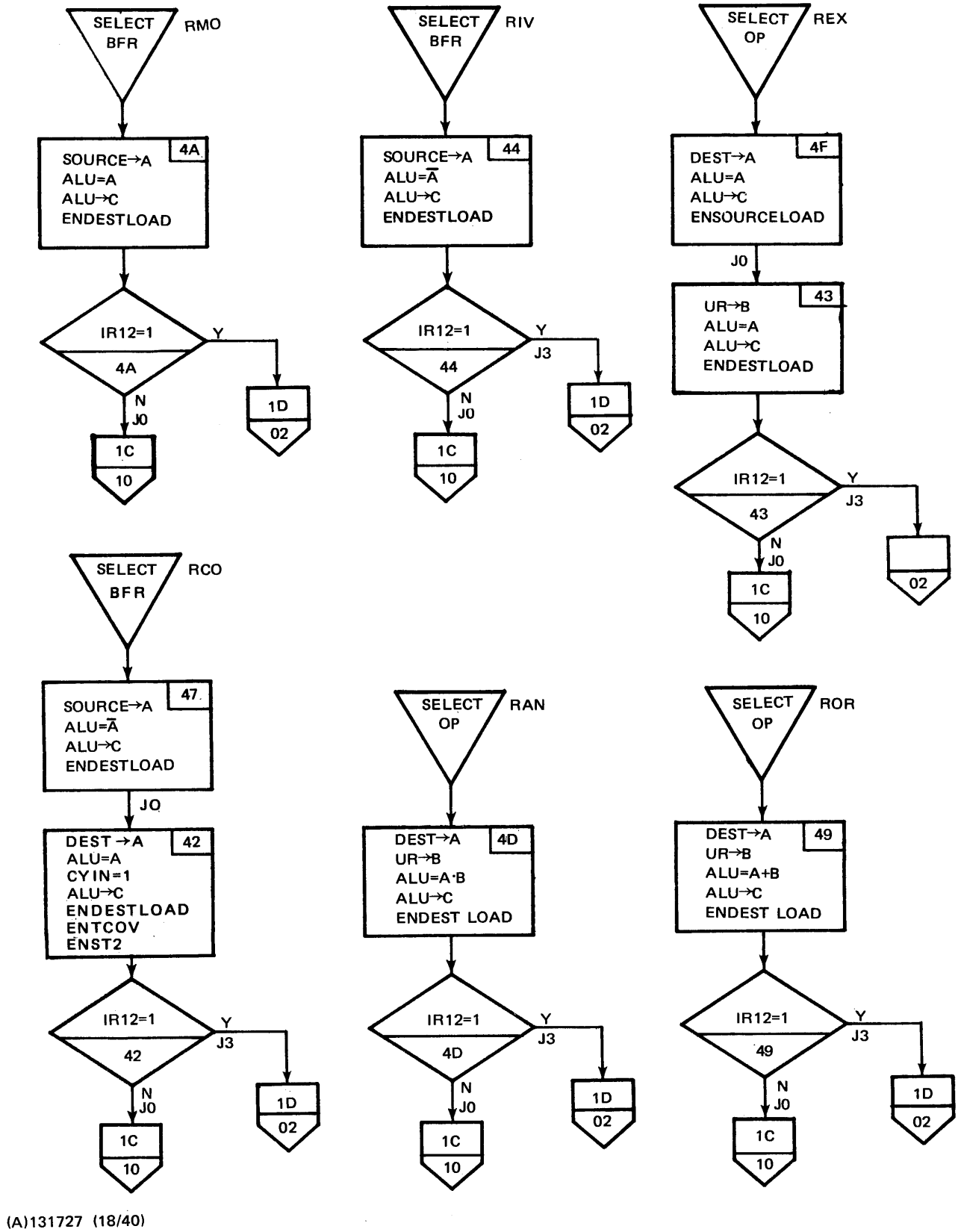
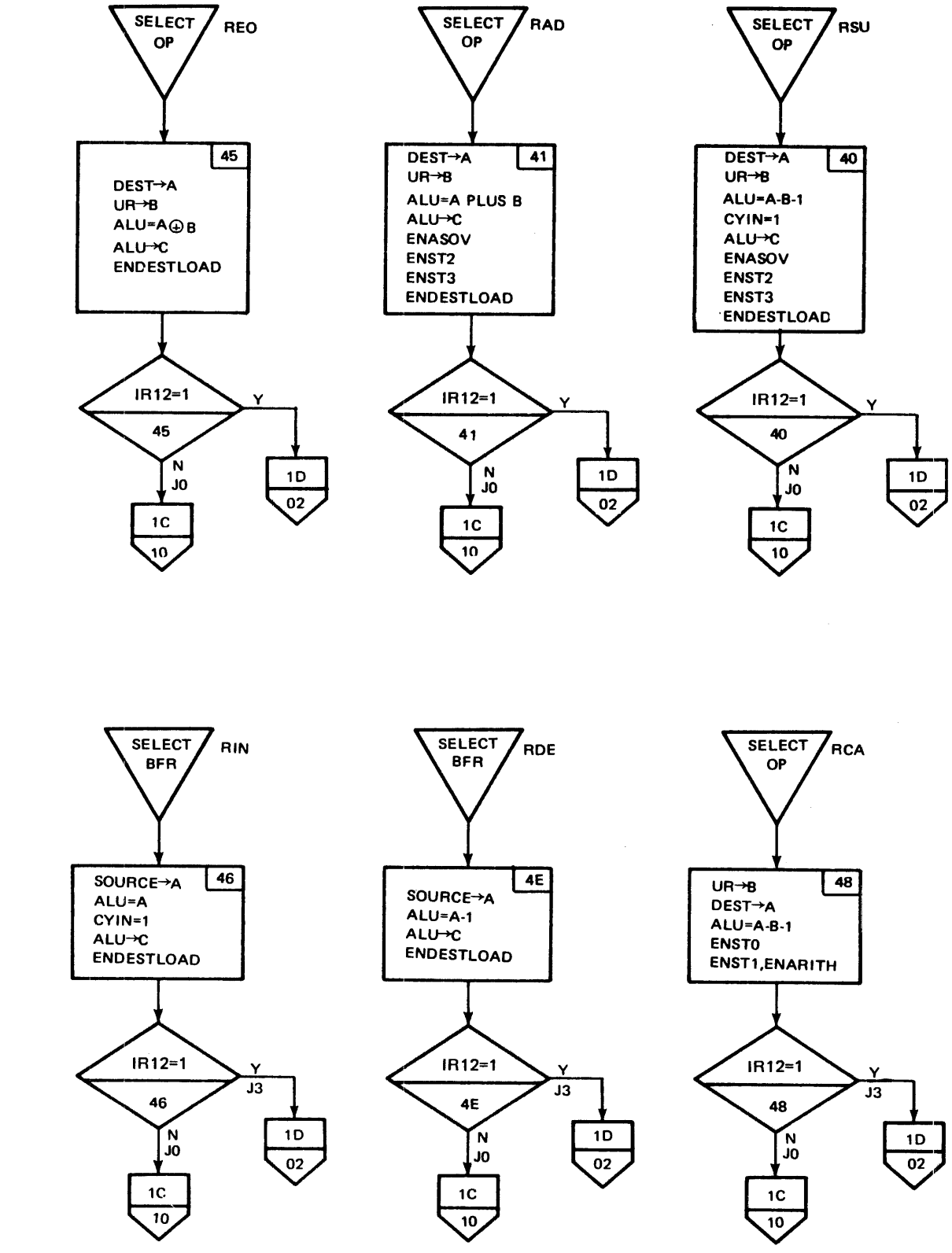


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 17 of 40)



(A)131727 (18/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 18 of 40)



(A)131727 (19/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 19 of 40)

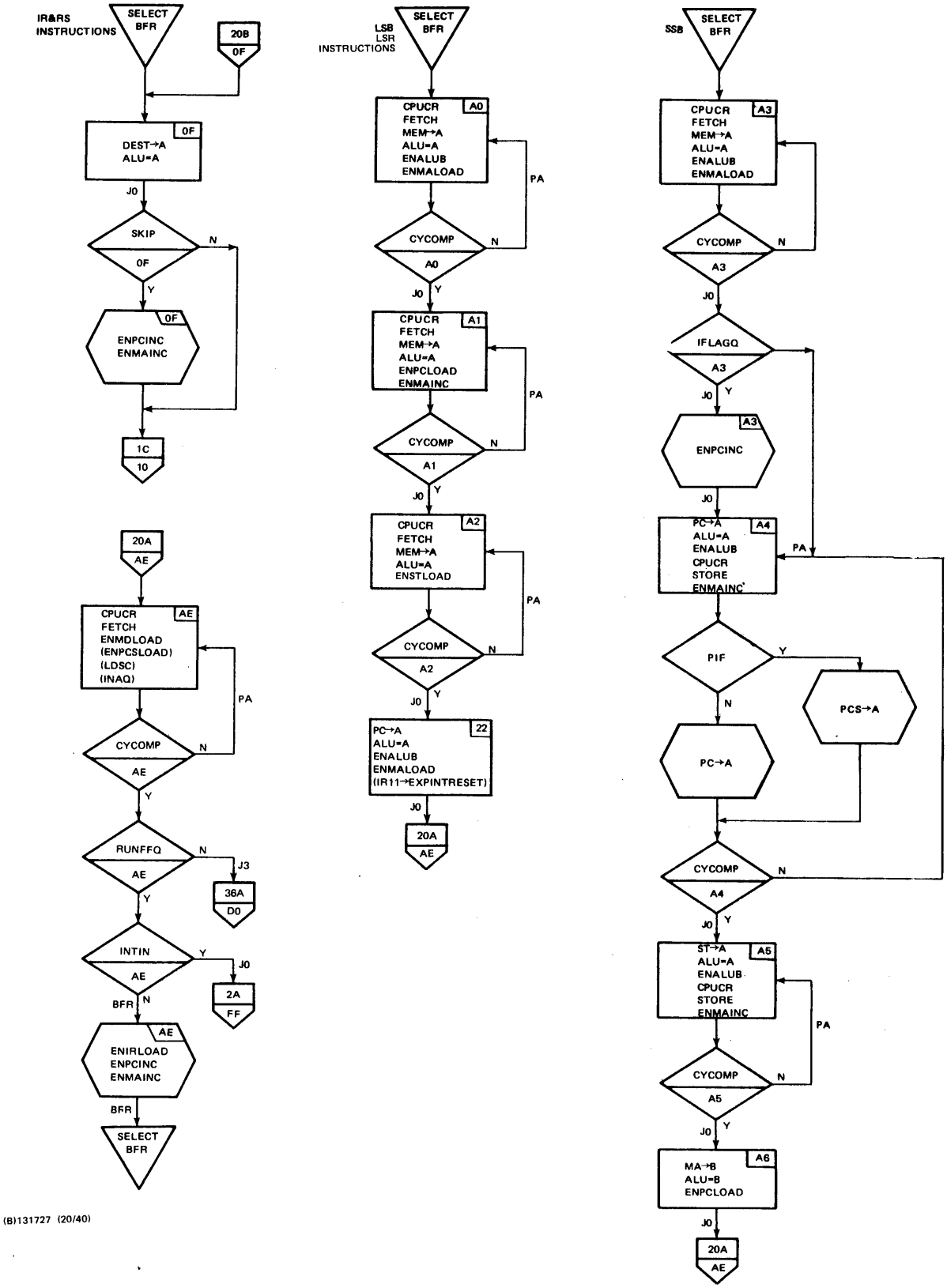
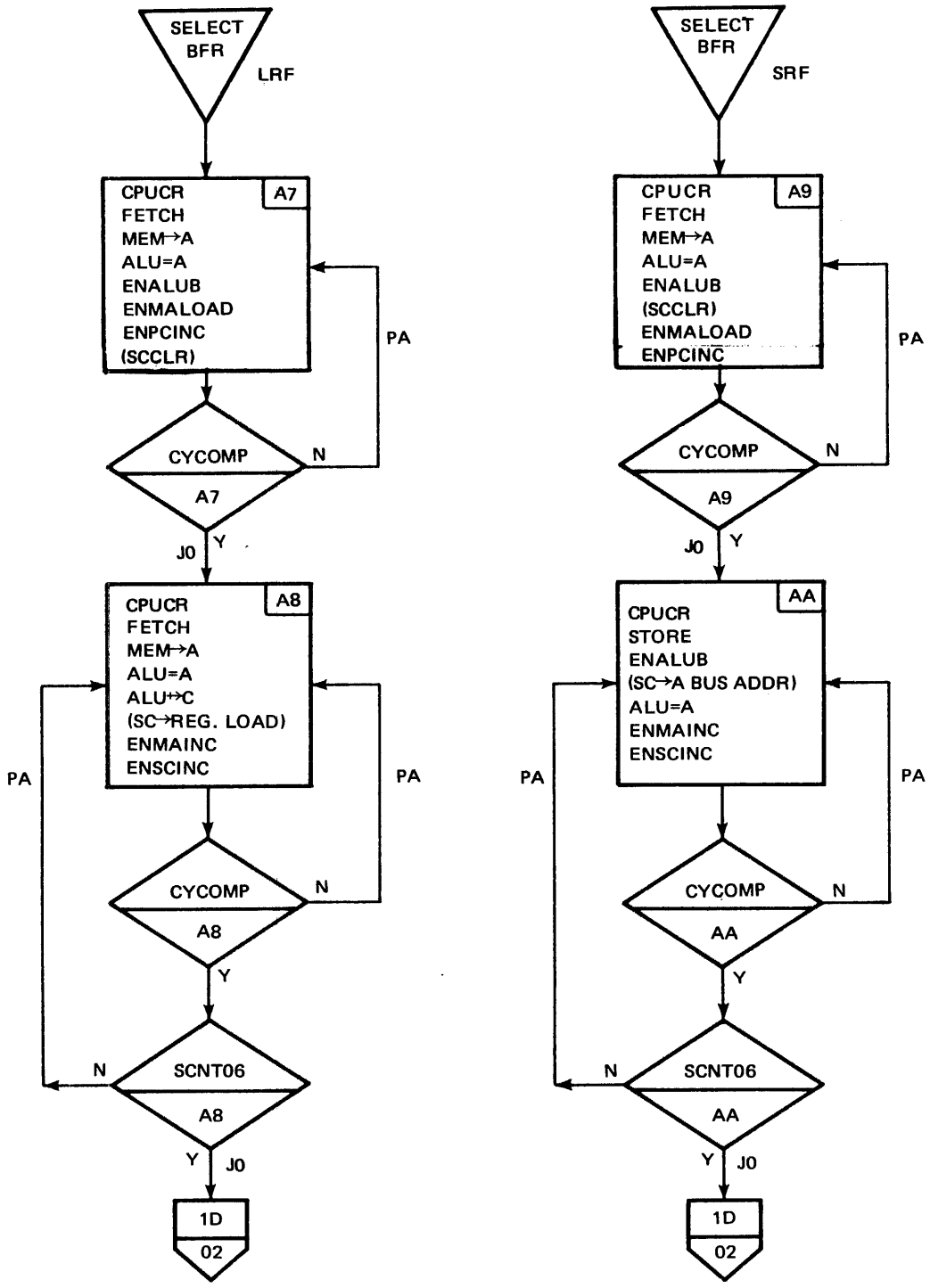
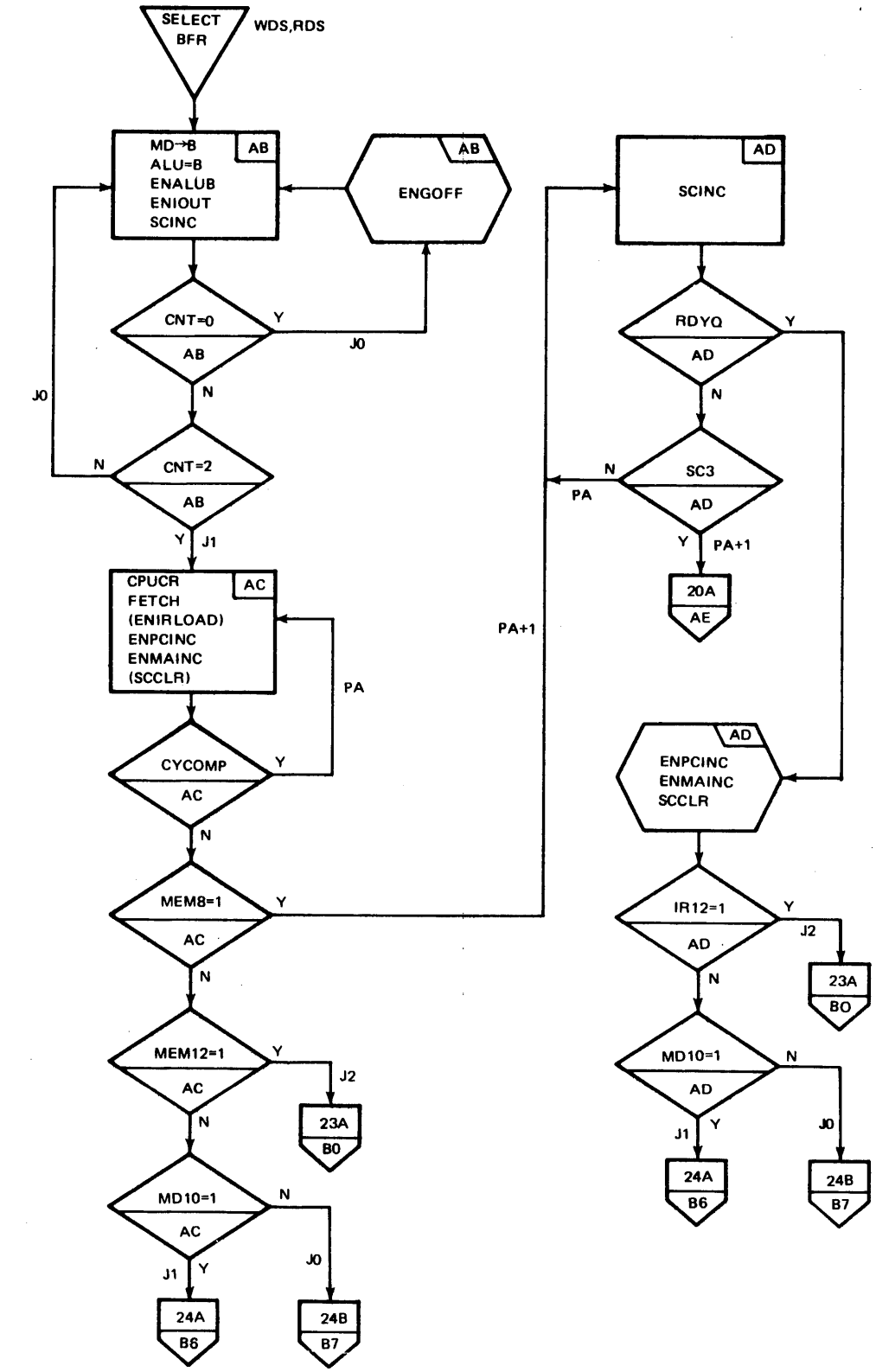


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 20 of 40)



(A)131727 (21/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 21 of 40)



(A)131727 (22/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 22 of 40)

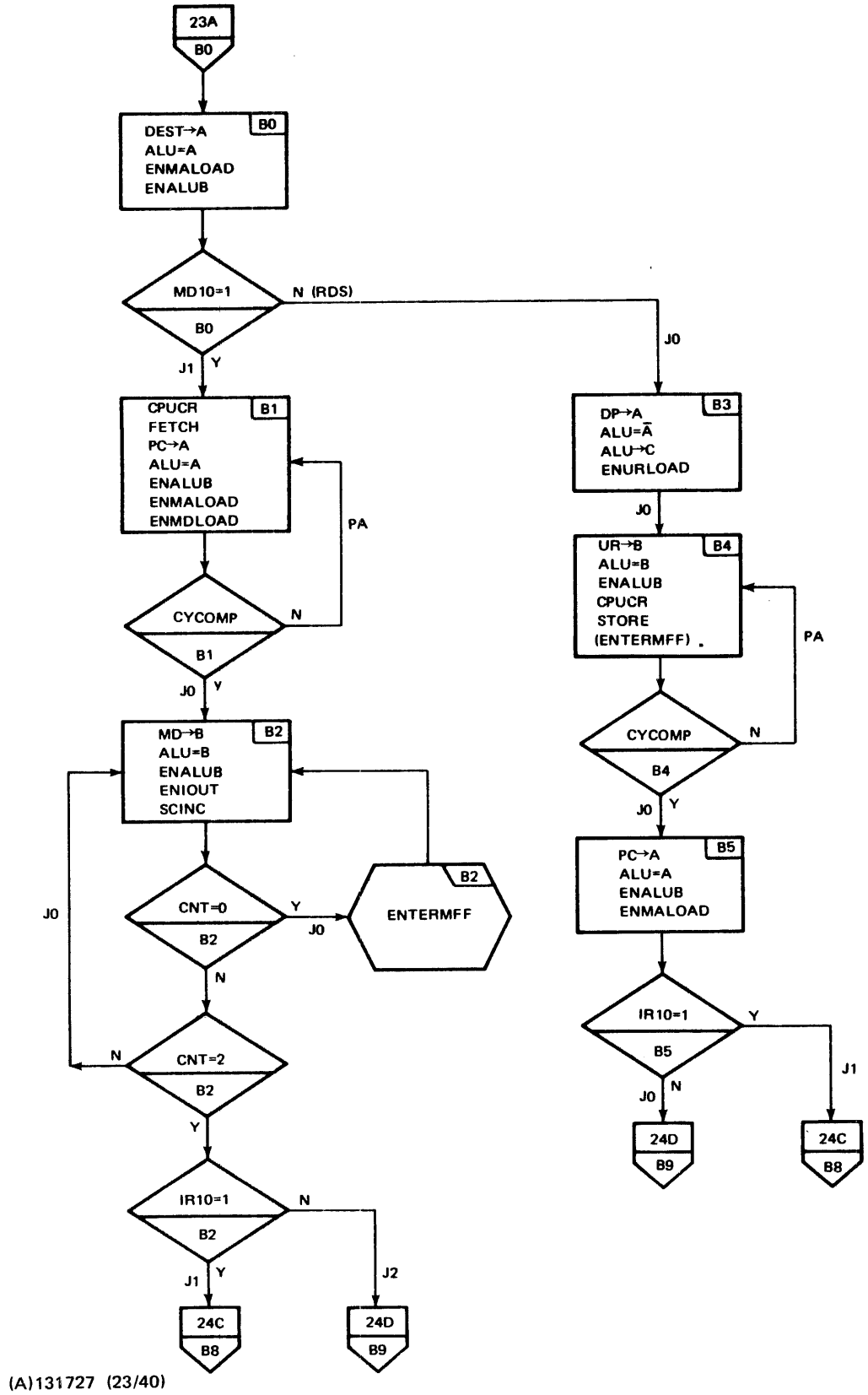
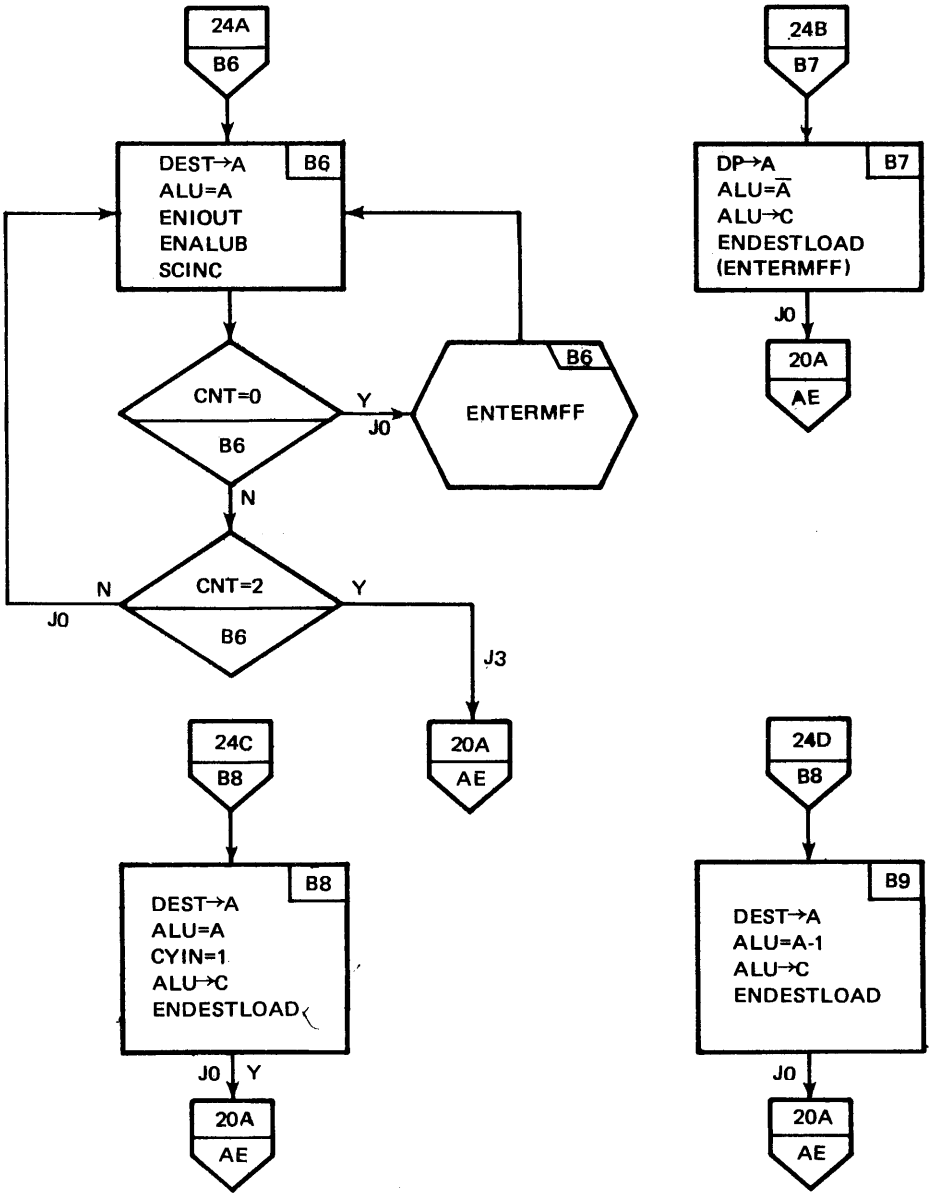
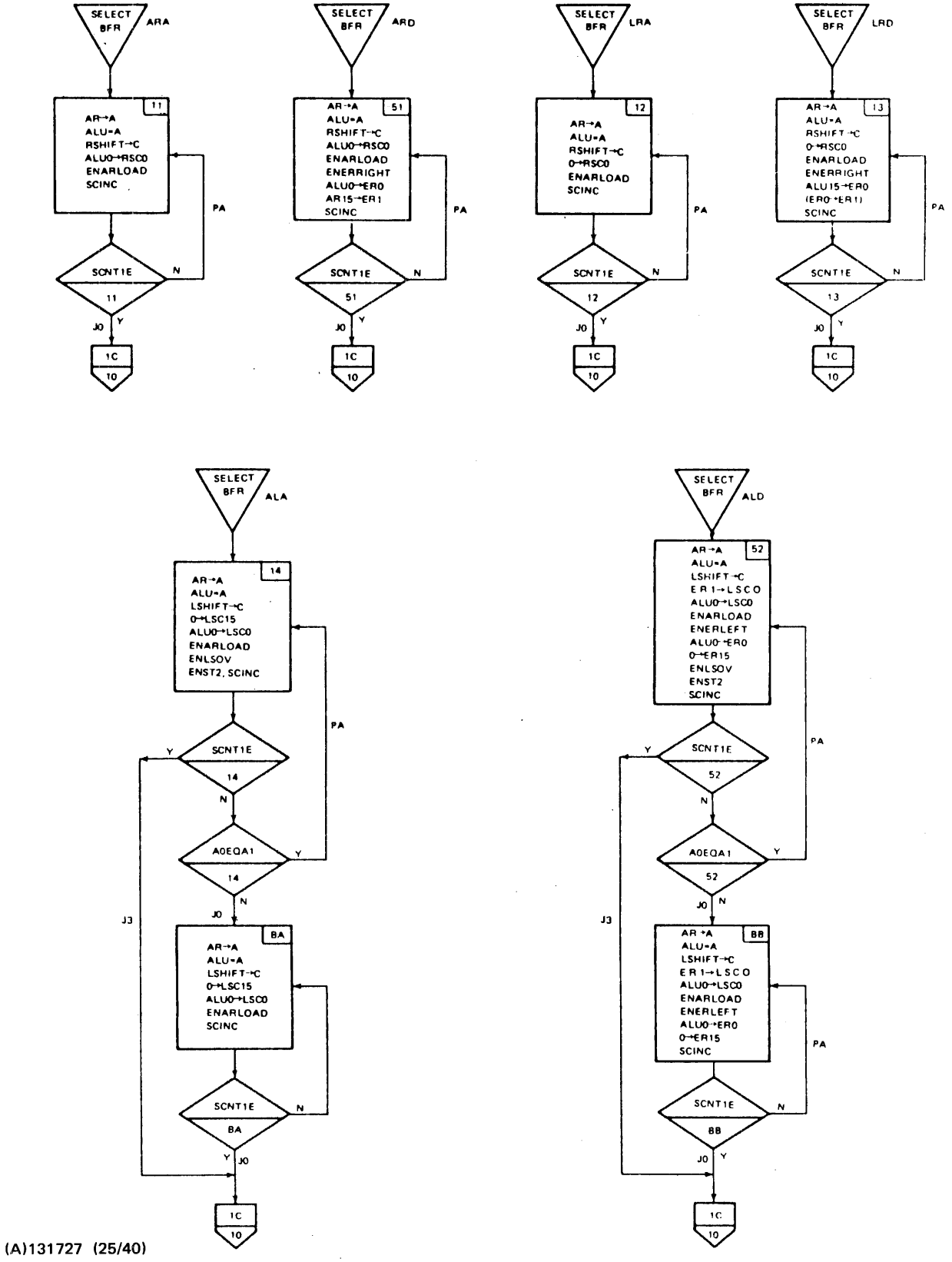


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 23 of 40)



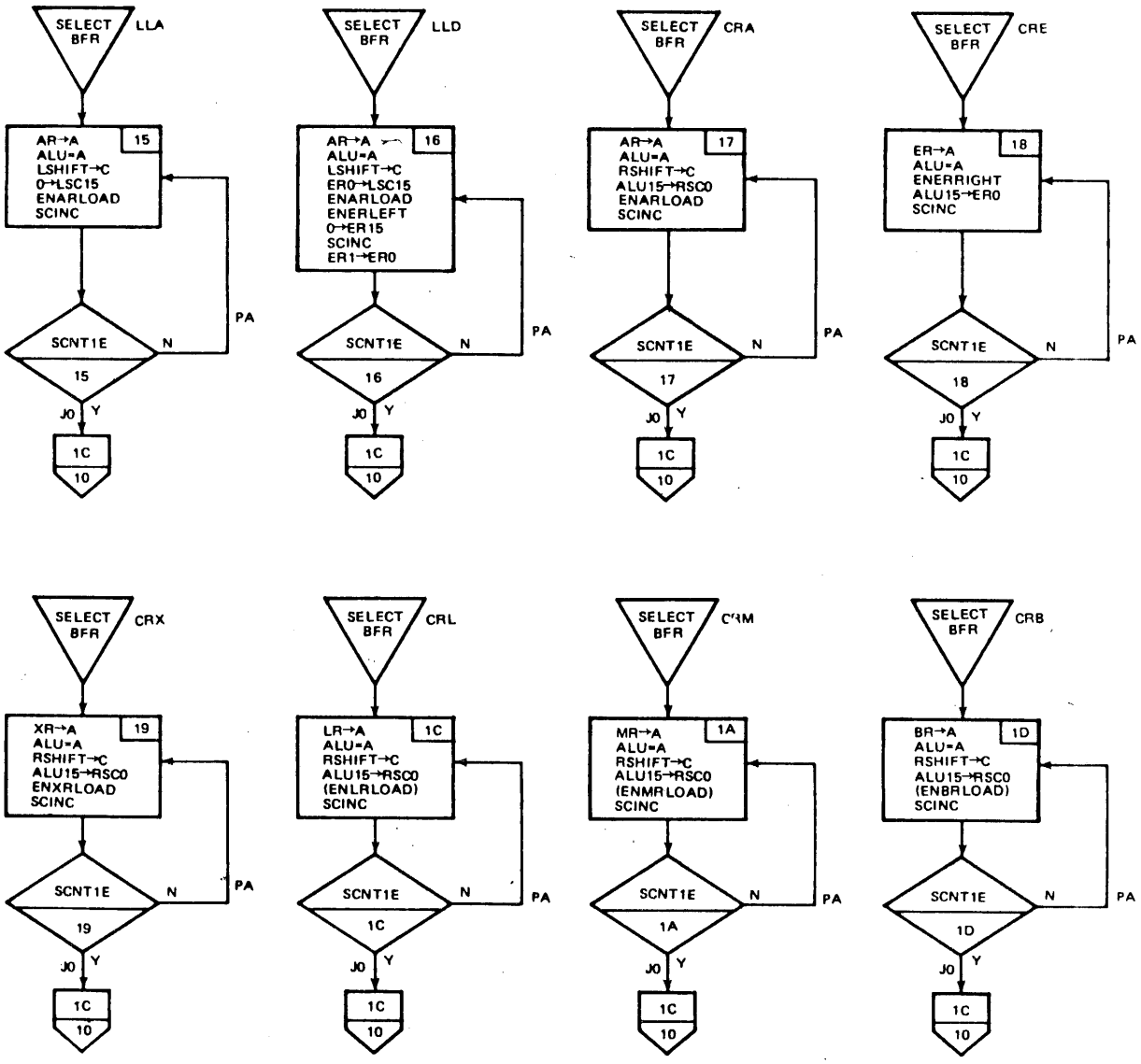
(A)131727 (24/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 24 of 40)



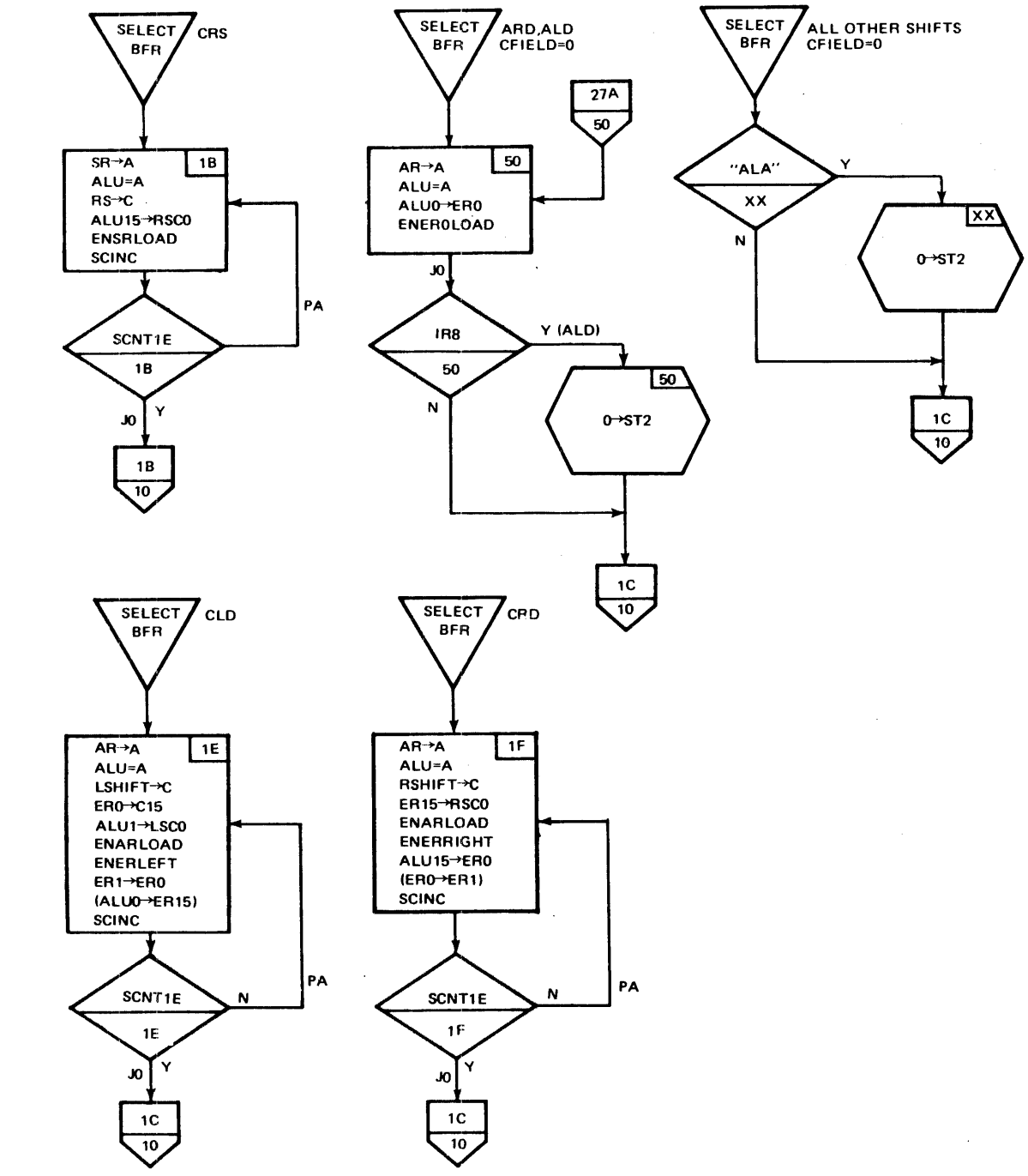
(A)131727 (25/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 25 of 40)



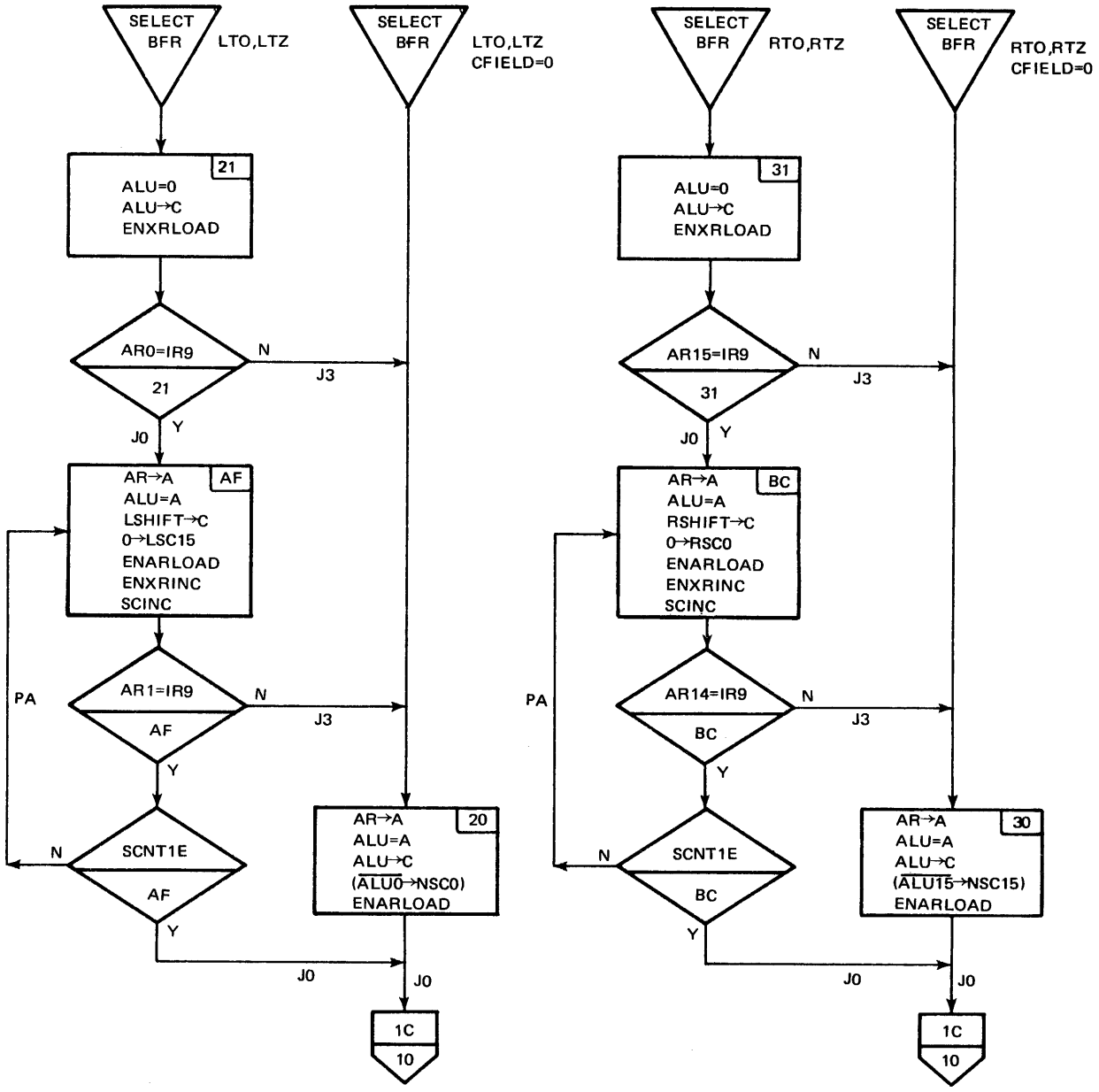
(A)131727 (26 /40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 26 of 40)



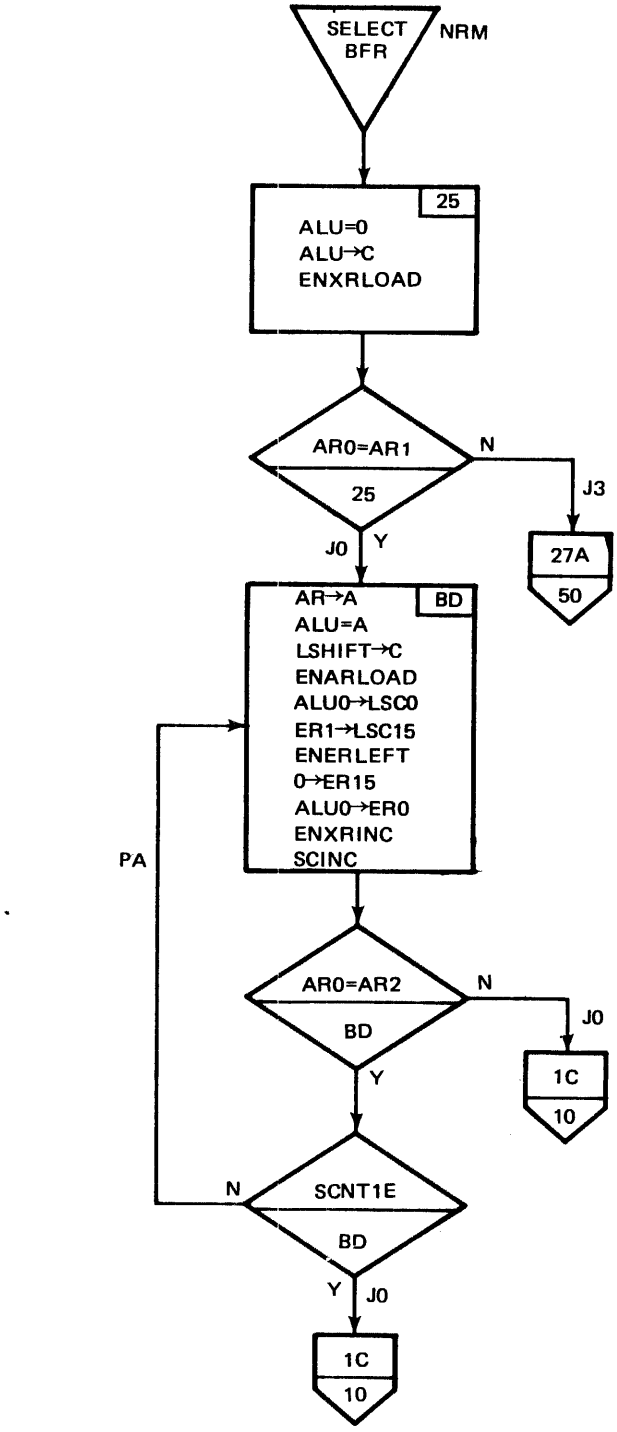
(A)131727 (27/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 27 of 40)



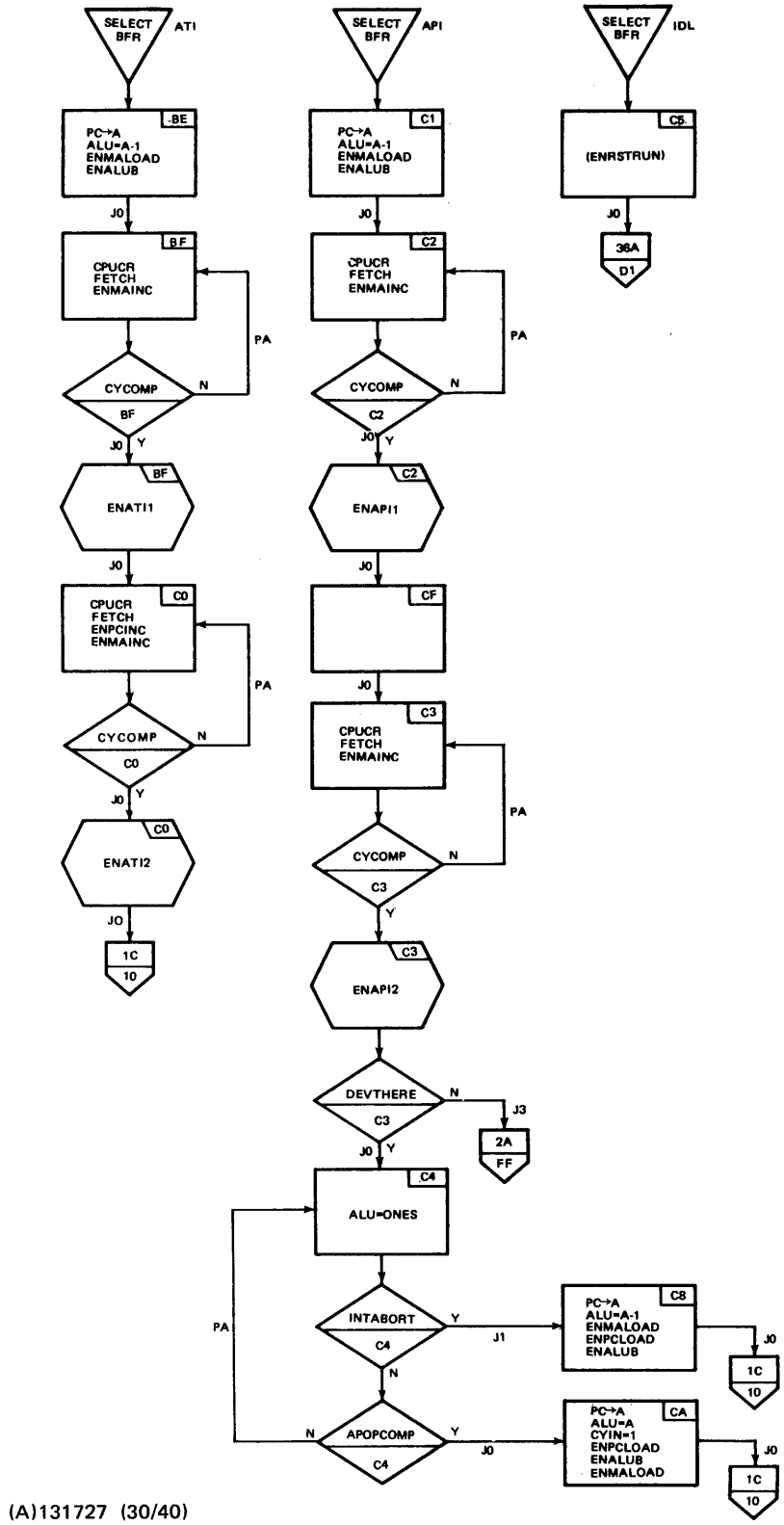
(A)131727 (28/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 28 of 40)



(A)131727 (29/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 29 of 40)



(A)131727 (30/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 30 of 40)

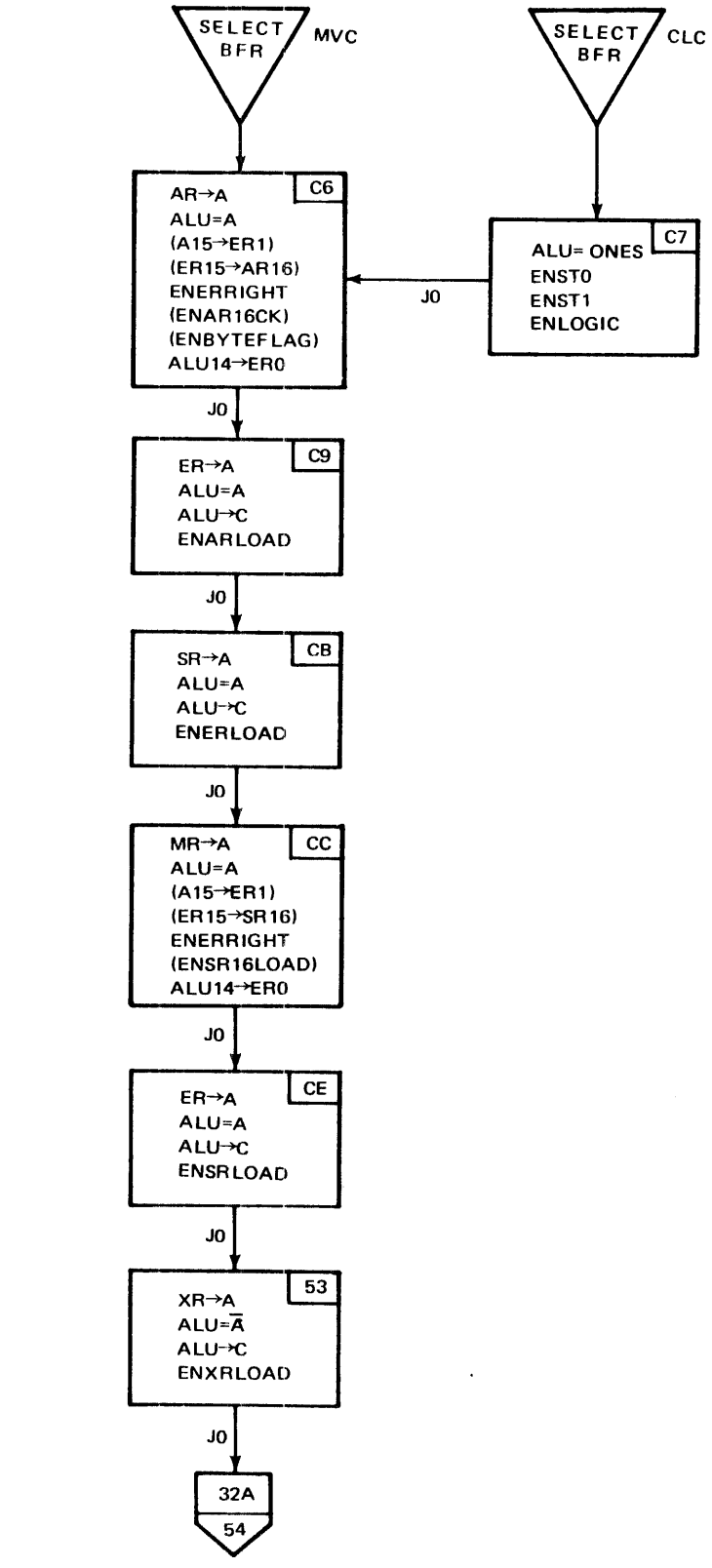


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 31 of 40)

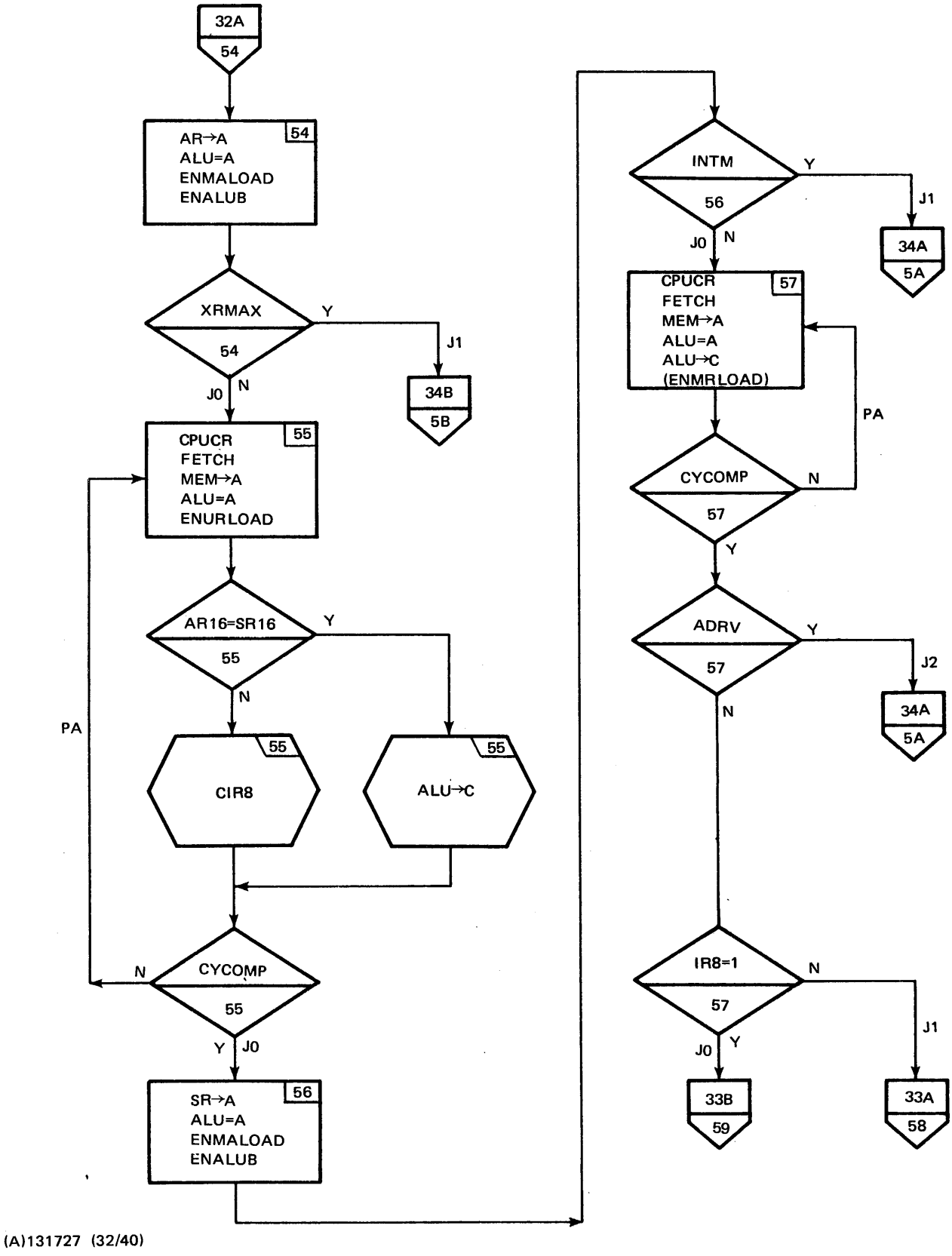


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 32 of 40)

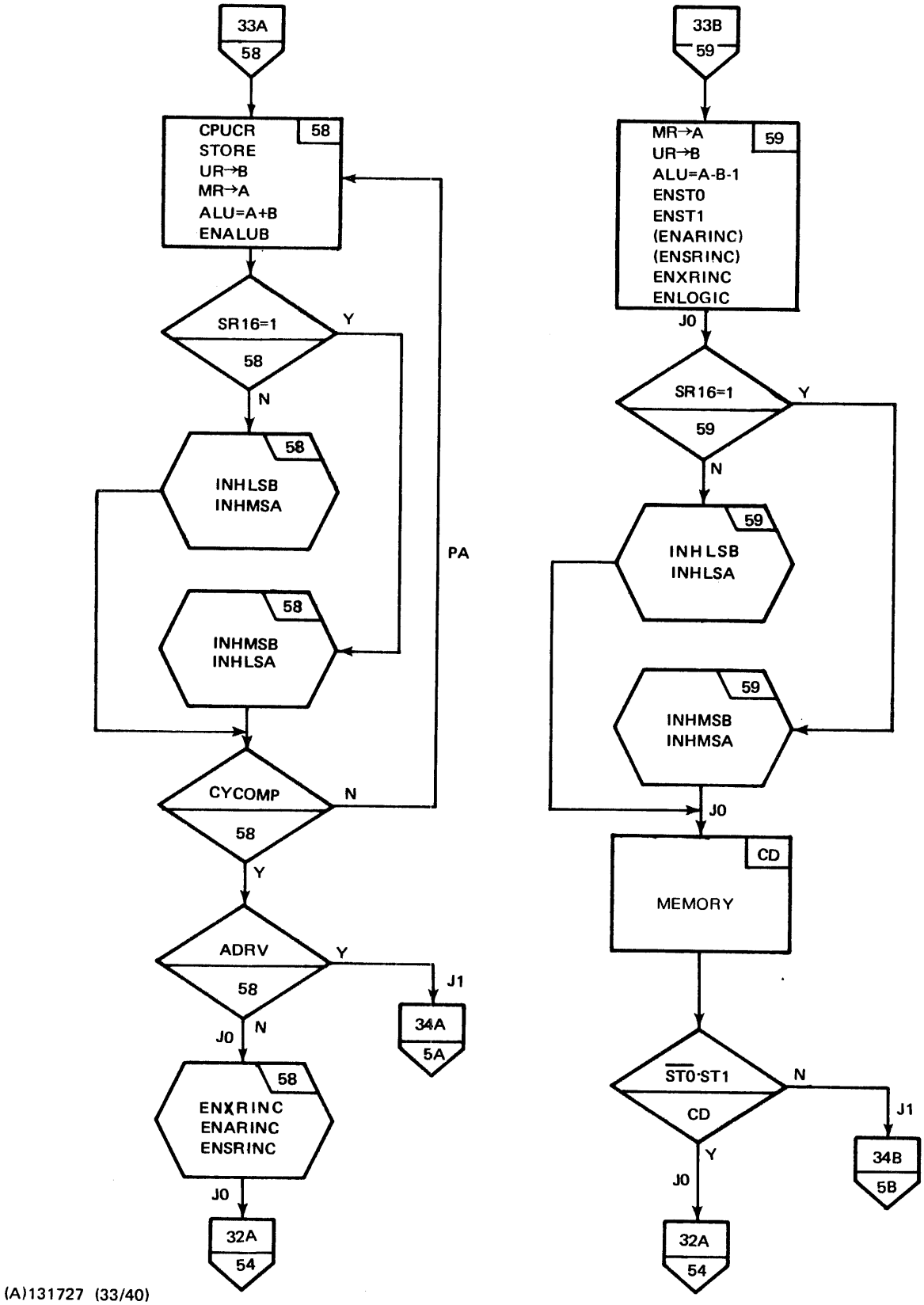
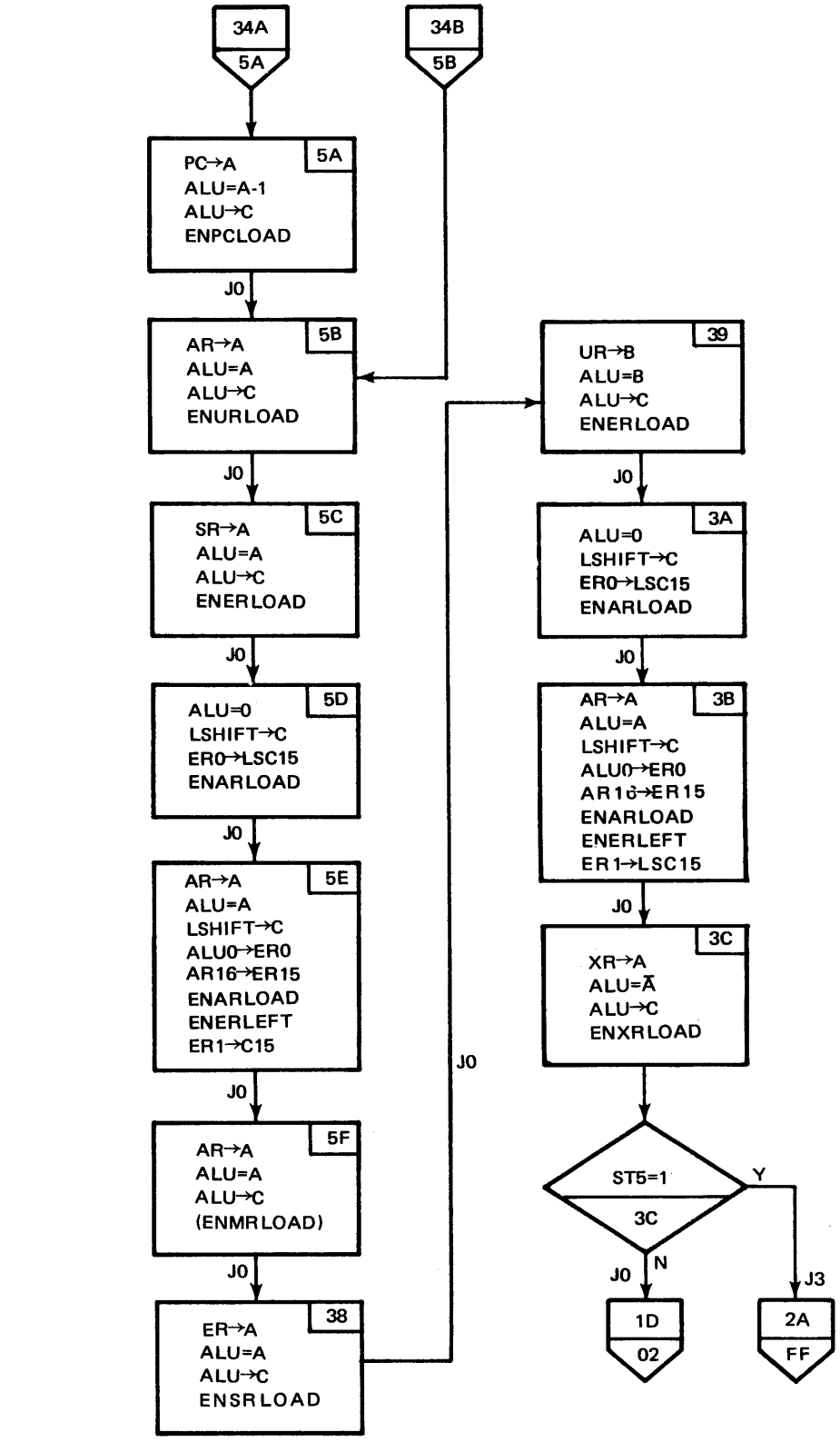
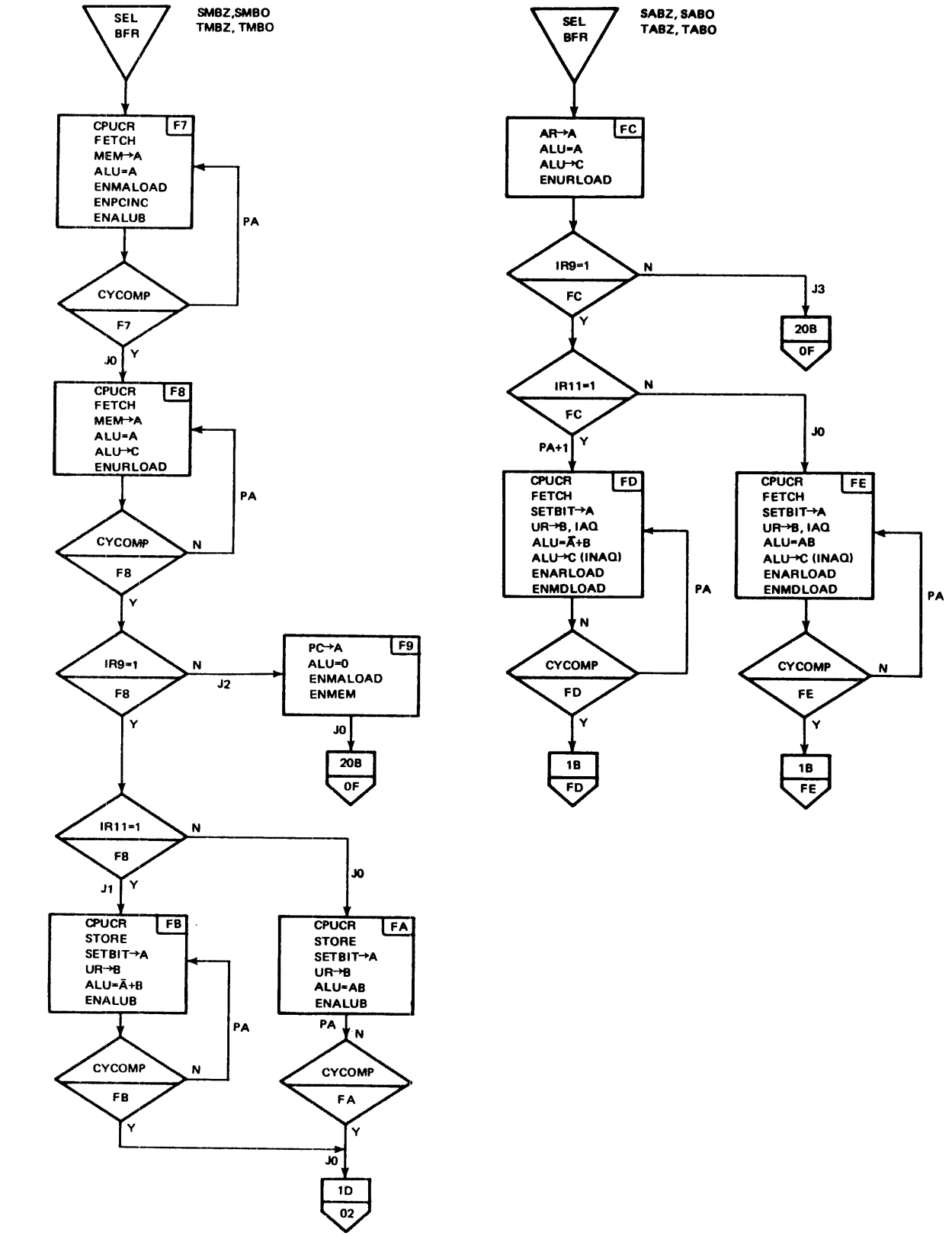


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 33 of 40)



(A)131727 (34/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 34 of 40)



(A)131727 (35/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 35 of 40)

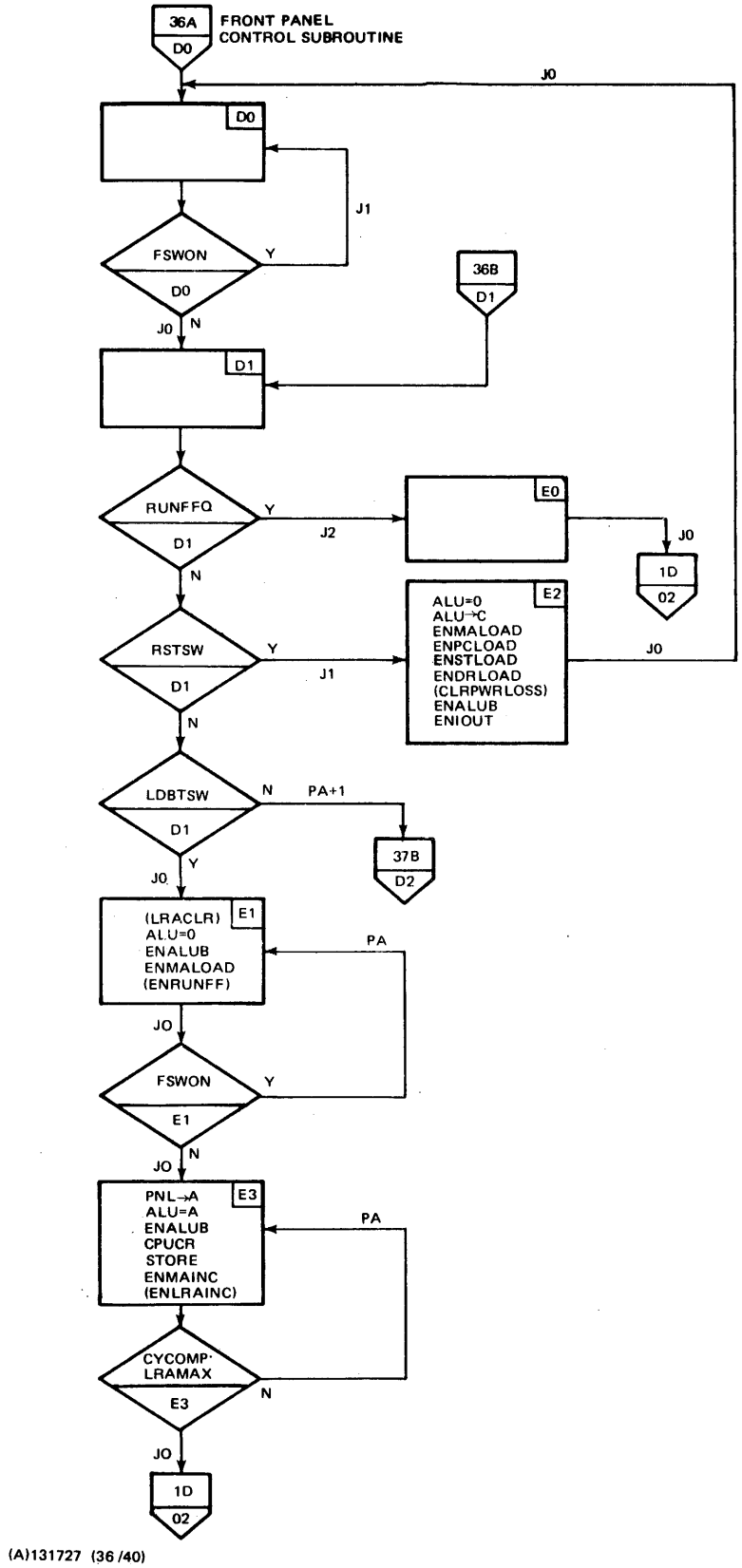
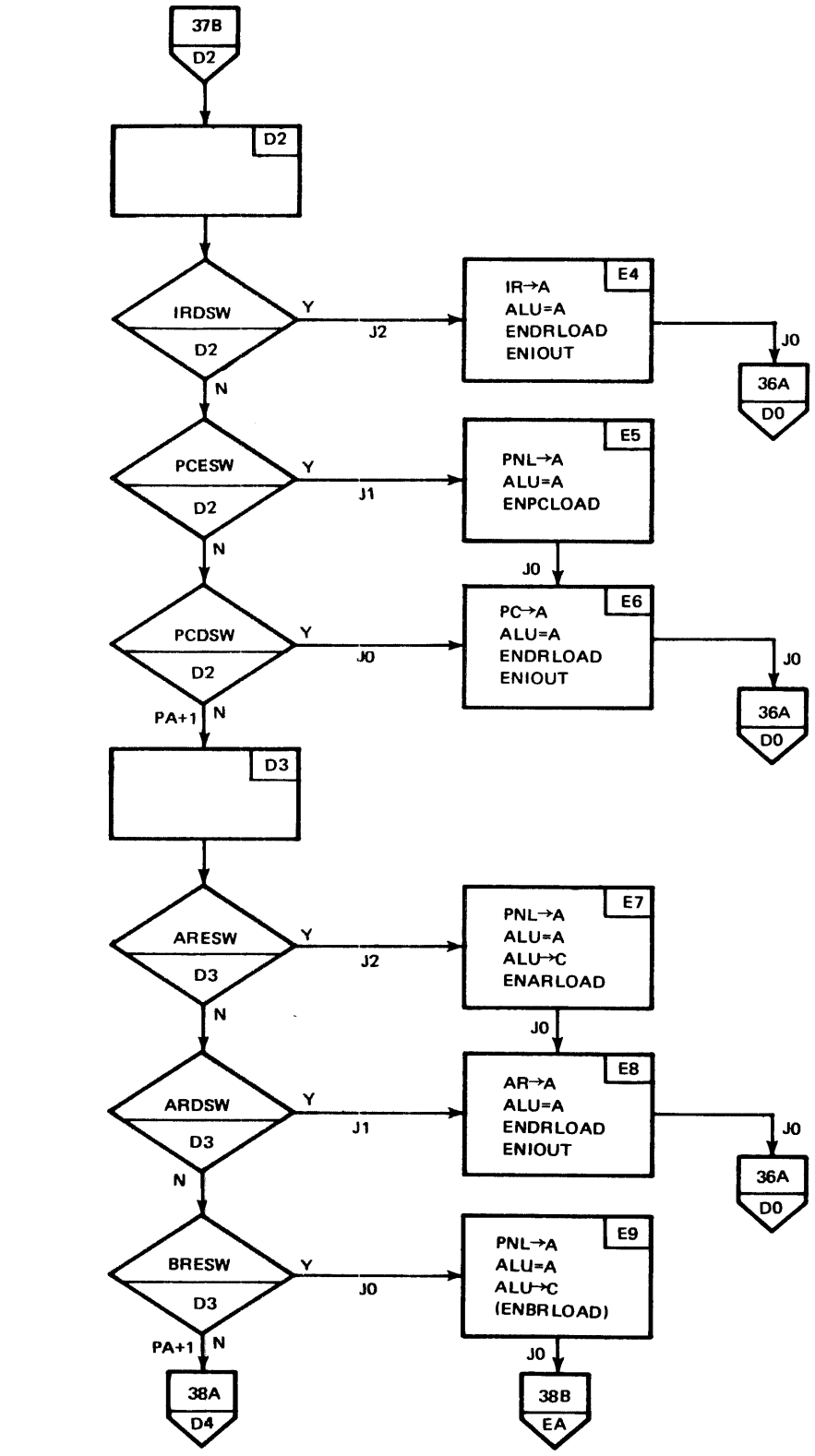
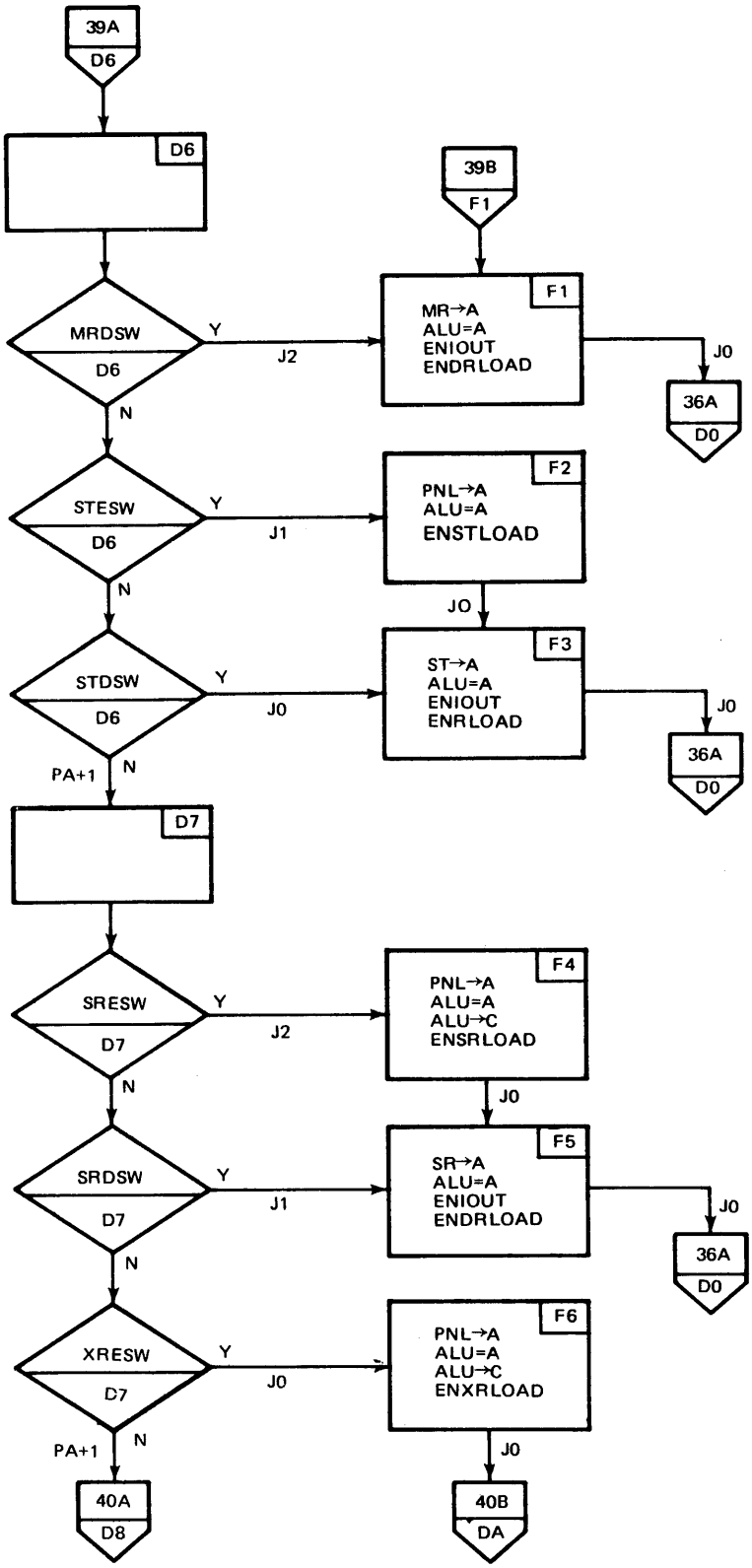


Figure 4-14. Arithmetic Unit Flowcharts (Sheet 36 of 40)



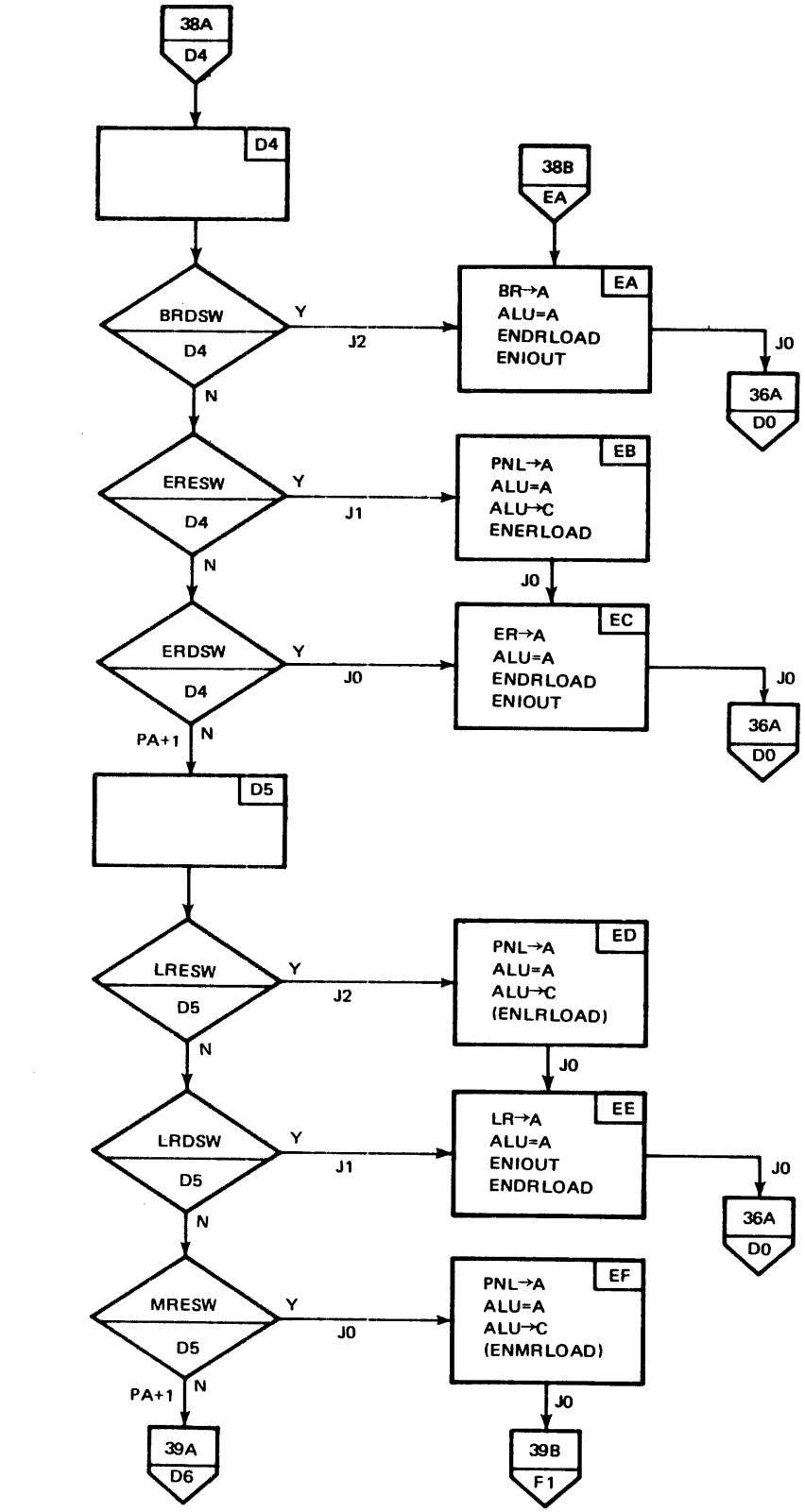
(A)131727 (37/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 37 of 40)



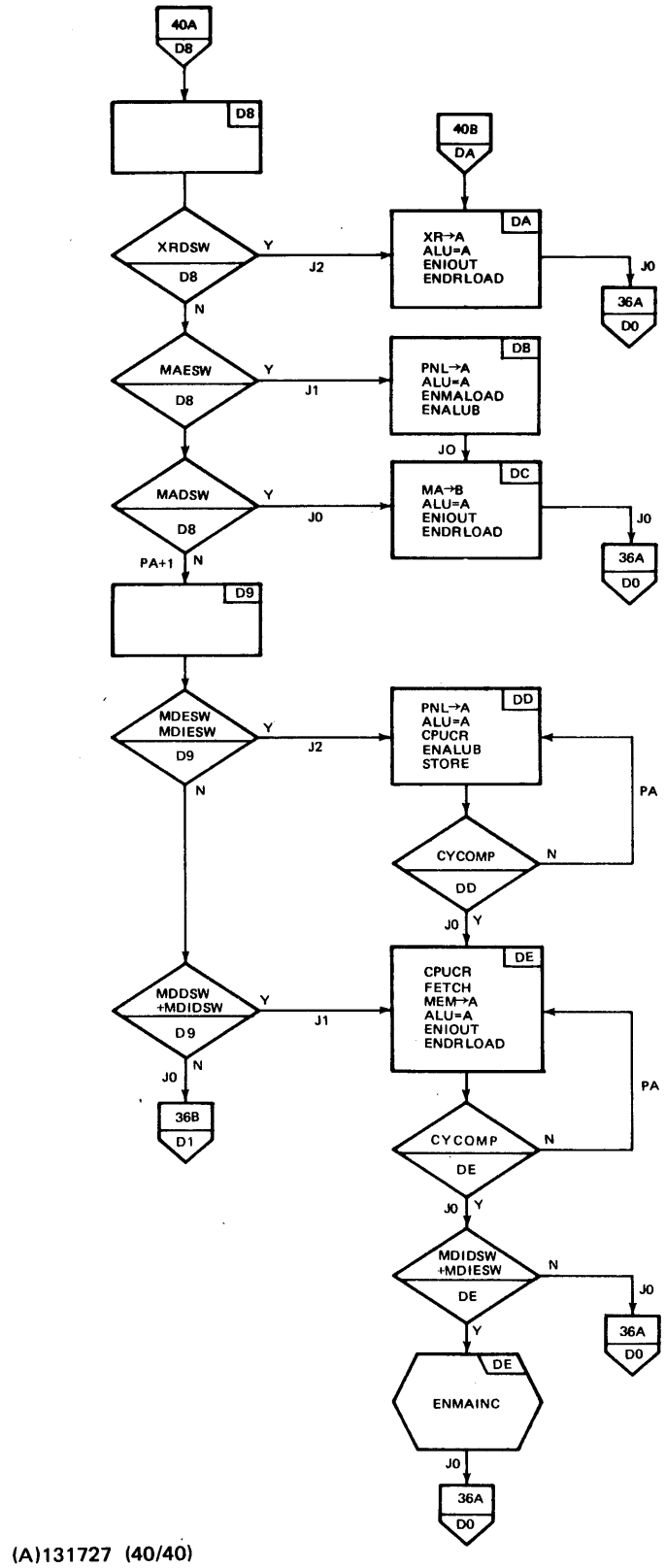
(A)131727 (39/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 38 of 40)



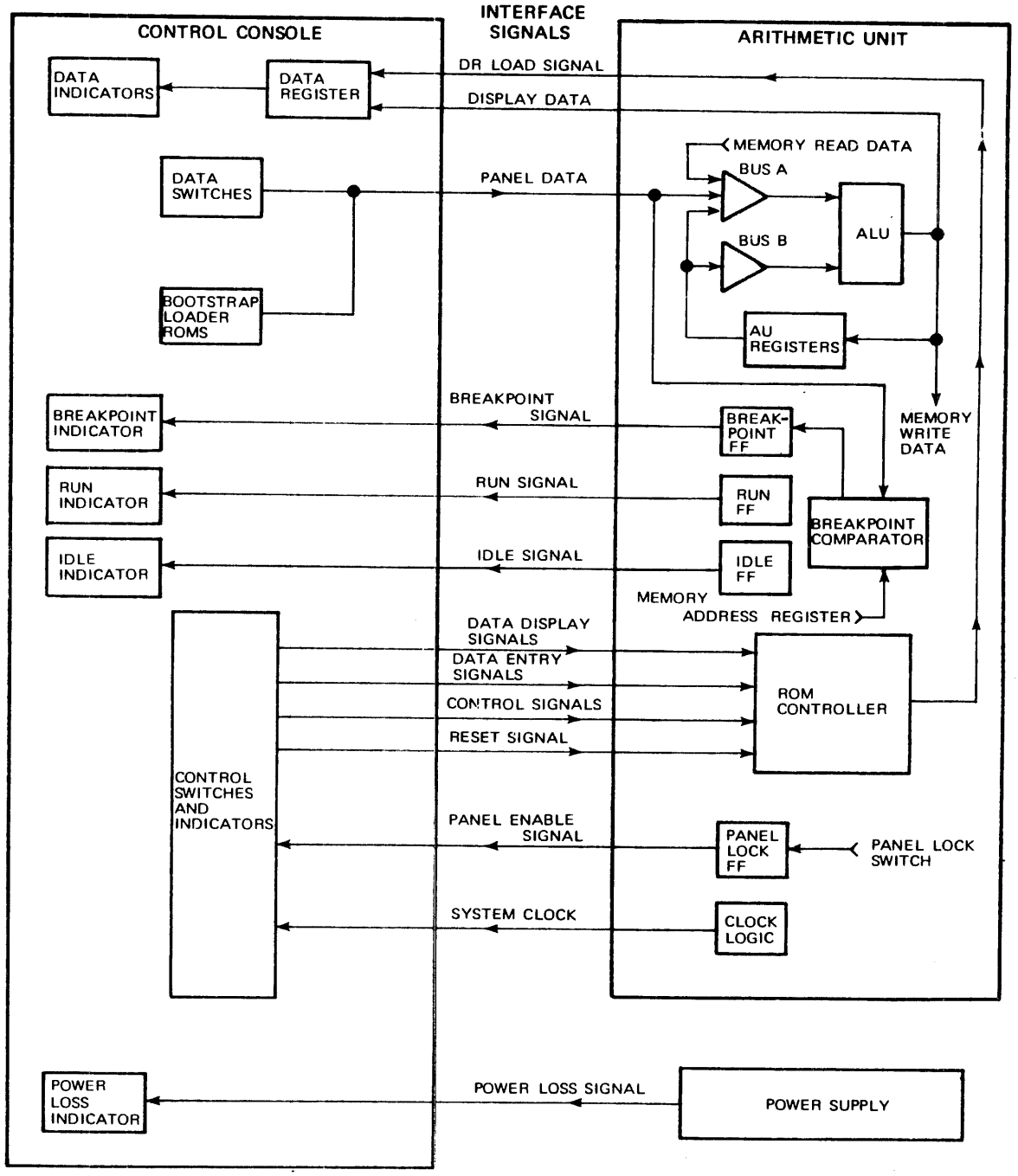
(A)131727 (38/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 39 of 40)



(A)131727 (40/40)

Figure 4-14. Arithmetic Unit Flowcharts (Sheet 40 of 40)



(A)131728

Figure 4-15. Control Console System Interface, Functional Diagram

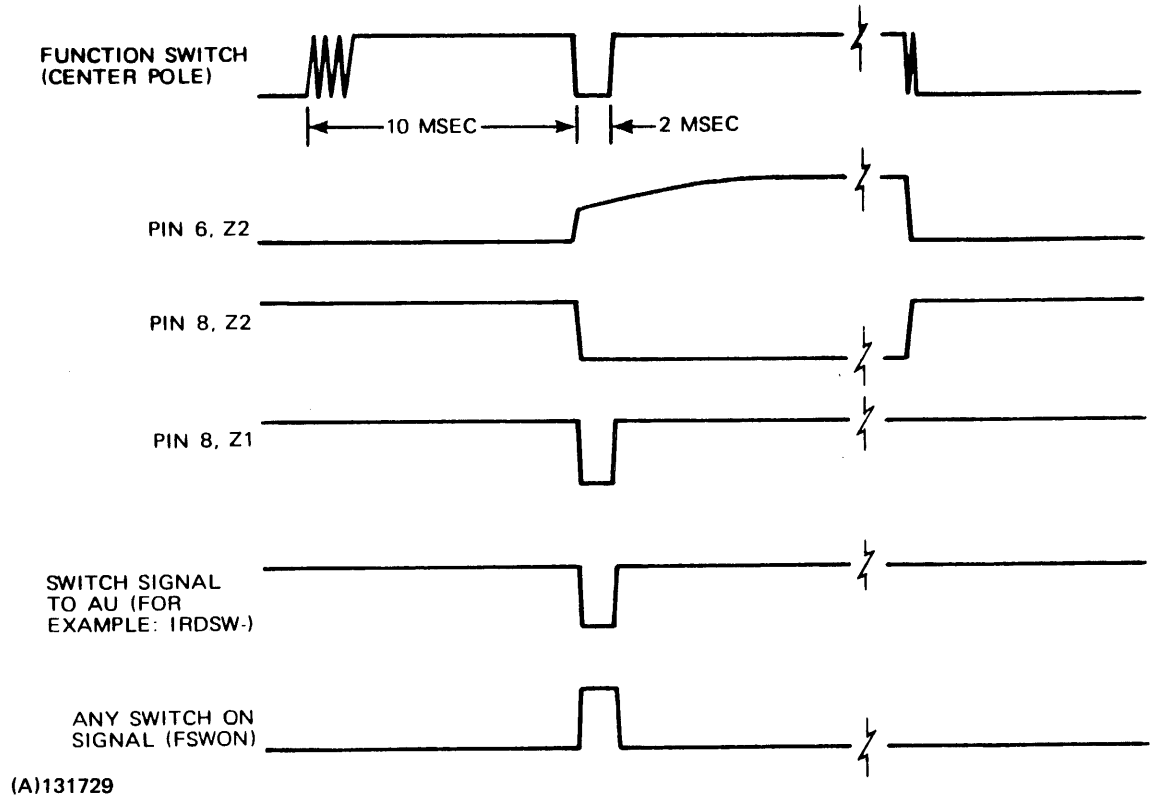


Figure 4-16. Function Switch Timing Diagram

SECTION V

MAINTENANCE OF ARITHMETIC UNIT AND CONTROL CONSOLE

5.1 GENERAL

This section contains trouble analysis procedures for the Arithmetic Unit and Control Console. Only general procedures and guidelines are established in this section as there are numerous procedures that can be implemented to achieve the same end result.

Numerous Texas Instruments documents have been published as an aid for trouble analysis. These documents include all volumes of the Maintenance Manual, the *Model 980 Computer Assembly Language Machine Instruction* (TI Part Number 961961-9730, and the *Model 980 Computer CPU Performance Assurance Tests (PAT).* There is also a Verification and Test Console (test fixture) that is designed for use during trouble analysis. This test fixture is described in a subsequent paragraph of this section. While not essential, this test fixture is extremely useful.

The following procedures include the use of the Verification and Test Console. Additional test equipment should include a multimeter and an oscilloscope. (Teltronix Type 454 or equivalent with 10X probes is recommended.)

5.2 FINDING A FAULT

Faults are separated into two classes:

- Class I faults leave the CPU basically operating, but causes one or more instructions to fail.
- Class II faults leave the console inoperative or keep any instruction from being executed properly.

Class I failures are generally discovered by a programmer when a specific instruction produces an erroneous answer or the loader stops at location XX doing a XXXXXXX or the PAT fails during system checkout. In any case, the failure is expected to occur during the implementation of one specific instruction. After the instruction is identified the first steps are: key this instruction into memory with known operands, single execute, and check the results. An error should be obvious. If the error is not obvious, load and execute the related PAT to detect the presence of an error in this same instruction. Selection, execution, and listings of the PAT are contained in the Programming Guide. If an error is not detected, run the full set of PAT's. If the error remains undetected, obtain additional information from the person that reported the failure. Once a failing instruction is located, the procedure is similar to that for Class II failures.

Class II failures generally result from a bad integrated circuit (IC) in the state transfer logic. One bad flip-flop in the ROM address register is sufficient to reroute the microroutines including the microroutine that services the console.

The AU functional diagram (figure 4-5); AU microsequence flowcharts (figure 4-13); and load, pin, and documentation lists are provided as aids to determine the precise location of a fault. The flowcharts identify functions that occur every period the clock is running. Most failures occur as a deviation from this sequence. Most Class I failures are the result of a bad gate in the test logic. Class II failures also frequently result from a bad gate in the test logic. A bad gate or

flip-flop in the state transfer path also is frequently the cause of Class II failures. In either case, the test fixture should be installed for trouble analysis.

If instructions are executing, set up a short loop while executing the failing instruction and branching to the location of that instruction. The Programmer's Reference Manual provides the information required to construct these instructions.

Start the loop to execute. Attach channel one of the oscilloscope to the RA SYNC test point on the test fixture. It is now possible to determine what states (ROM addresses) the microsequence is entering. This terminal is high when the ROM address agrees with the number set in the ROM address switches. Find the instruction being executed in the flowcharts. Set the charts to the first state. Set all succeeding switches to assure that the path is followed.

At some point, the proper sequence may not be followed. When this occurs, set the switches for the last correct state and determine what caused the incorrect path to be followed. This involves a number of possibilities which are determined by the function that is being accomplished in that state and the decisions that are being made. One possibility is that the gate generating the signal to force a particular decision is bad. To detect a bad gate: locate the signal in the documentation list, sync the oscilloscope on channel one, and examine the inputs and outputs using channel two of the oscilloscope when operating in the alternate mode. This test configuration permits the signals to be observed only during the related state. If the output is the correct function of the inputs, retrace the logic tree while monitoring the circuitry for bad gates. The documentation list is designed as an aid for a test of this type.

If the microsequence appears to be followed properly, consider the nature of the error and examine bit-by-bit the actions performed during each state using RA SYNC and the documentation list to locate the related signals. Observe that data bus and register control signals are correct in each state. The theory of operation section of the maintenance manual and the associated functional diagram identifies those signatures that should be examined.

In some cases (including Class II failures) the dynamic mode of operation is not feasible. Generally, the microsequence is looping in an abnormal path that prevents normal trouble analysis. Alternately, it may be necessary to know the exact sequence of states performed. To accomplish this function, the microsequence must be executed one clock pulse at a time while continuously observing the controller state and operations on the test fixture displays.

To supply single clocks, the console clock jumper E80 and E81 must be disconnected from 80 and moved to E81. With the jumper disconnected the clock switch functions as described in Section IV of this manual.

Assuming a Class I failure has occurred; turn on the clock and construct the instruction in the registers, set PC to the first word address of that instruction, set correct status, place the CLOCK switch to the OFF position, place the MODE switch to RUN, and depress the CLOCK switch (STEP position) twice while depressing the START switch (START position). This switch action sets RUNFF. The test fixture should display the microsequence to be in the console service routine. Stepping the clock repeatedly should remove the controller from the console routine. When removed from the console routine, instruction acquisition and decoding begins.

If a Class II failure has occurred: place the CLOCK switch on the OFF position, turn ac power OFF and then ON, and STEP the CLOCK through the power up sequence until the control path is lost.

5.3 REPAIR

In most cases a fault can be traced to a bad component on a printed-circuit (PC) board. This is generally an integrated circuit. Once it has been determined to replace a component on a PC board, the procedure is simple. The component pin(s) should be cut as close as possible to the case of the integrated circuit, heated individually, and removed from the PC board. Removal of the pin should clear the hole except, in the case of VCC and GROUND, where additional heating and use of a solder remover may be required.

CAUTION

Do not attempt to push solder from a hole. Do not use a soldering iron rated at more than 30 Watts.

5.4 CONSOLE FAULTS

If a fault is traced to a signal originating on the control console, refer to Section IV of this manual for a detailed description of console operation. A schematic of the console is included in the electrical drawings manual for the computer.

5.5 MEMORY, CRU, DMAC, MEMORY CONTROLLER I/O, OR POWER SUPPLY FAULTS

Faults that are traced to the memory or memory controller can be corrected by referring to the Memory, Memory Controller and Direct Memory Access Channel (DMAC) manual for the computer. Similarly, refer to the relevant maintenance manual for failures that occur in other system assemblies.

5.6 VERIFICATION AND TEST CONSOLE OPERATION

The purpose of the Verification and Test Console (TI Part Number 960736-D001) is to display signals and signal functions of the Arithmetic Unit and Memory Controller. The displays on this test fixture are designed for use during trouble analysis. The test set consists of a display panel, a buffer card, and an interconnection cable.

The buffer card plugs into any Input/Output (I/O) card slot in the computer for a source of Vcc and ground. Three cables that are terminated in DIP plugs are permanently connected to the card. Two of the plugs are inserted into test sockets located on the AU board. The designation for each plug is marked on the associated cable. Orientation of each plug is the same as the circuits soldered to the board (with respect to the notched end). The third plug is used on the memory controller.

The interconnection cable is terminated with a 72-pin connector which connects to the top edge of the buffer card.

A front view of the display panel is illustrated in figure 5-1. The test fixture has four sets of LED displays that display the present and next ROM address (NRA), the select signals for the NRA selector, and a 16-bit data display. The set of eight 2-position switches allow the ROM Address (RA) sync state to be selected. The DIP plug for data display can be used at five locations on the memory controller. These plug sockets allow the actual memory address, the PIF upper limit, the PIF lower limit, and a set of miscellaneous memory controller control signals to be displayed on the data display. A pin list for the memory controller identifies the signals available on the test socket.

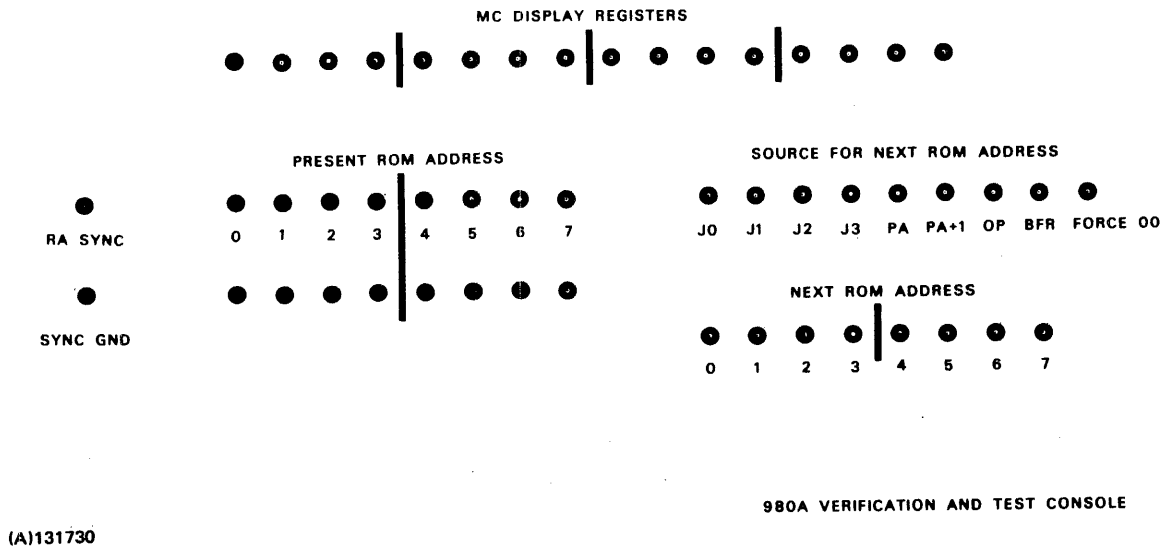


Figure 5-1. Verification and Test Console

ALPHABETICAL INDEX

ALPHABETICAL INDEX

INTRODUCTION

The following index lists key words and concepts from the subject material of the manual together with the area(s) in the manual that supply major coverage of the listed concept. The numbers along the right side of the listing reference the following manual areas:

- Sections - References to Sections of the manual appear as "Section x" with the symbol x representing any numeric quantity.
- Appendixes - References to Appendixes of the manual appear as "Appendix y" with the symbol y representing any capital letter.
- Paragraphs - References to paragraphs of the manual appear as a series of alphanumeric or numeric characters punctuated with decimal points. Only the first character of the string may be a letter; all subsequent characters are numbers. The first character refers to the section or appendix of the manual in which the paragraph is found.
- Tables - References to tables in the manual are represented by the capital letter T followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the table). The second character is followed by a dash (-) and a number:

  Tx-yy

- Figures - References to figures in the manual are represented by the capital letter F followed immediately by another alphanumeric character (representing the section or appendix of the manual containing the figure). The second character is followed by a dash (-) and a number:

  Fx-yy

- Other entries in the Index - References to other entries in the index are preceded by the word "See" followed by the referenced entry.

USER'S RESPONSE SHEET

Manual Title: Model 980 Computer Maintenance Manual
Arithmetic Unit and Control Console (960699-9702)

Manual Date: 1 September 1975 Date of This Letter: _____

User's Name: _____ Telephone: _____

Company: _____ Office/Department: _____

Street Address: _____

City/State/Zip Code: _____

Please list any discrepancy found in this manual by page, paragraph, figure, or table number in the following space. If there are any other suggestions that you wish to make, feel free to include them. Thank you.

CUT ALONG LINE

Location in Manual	Comment/Suggestion
_____	_____
	_____
	_____
	_____
_____	_____
	_____
	_____
	_____
_____	_____
	_____
	_____
	_____

NO POSTAGE NECESSARY IF MAILED IN U.S.A.
FOLD ON TWO LINES (LOCATED ON REVERSE SIDE), TAPE AND MAIL

FOLD

FIRST CLASS
PERMIT NO. 7284
DALLAS, TEXAS

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

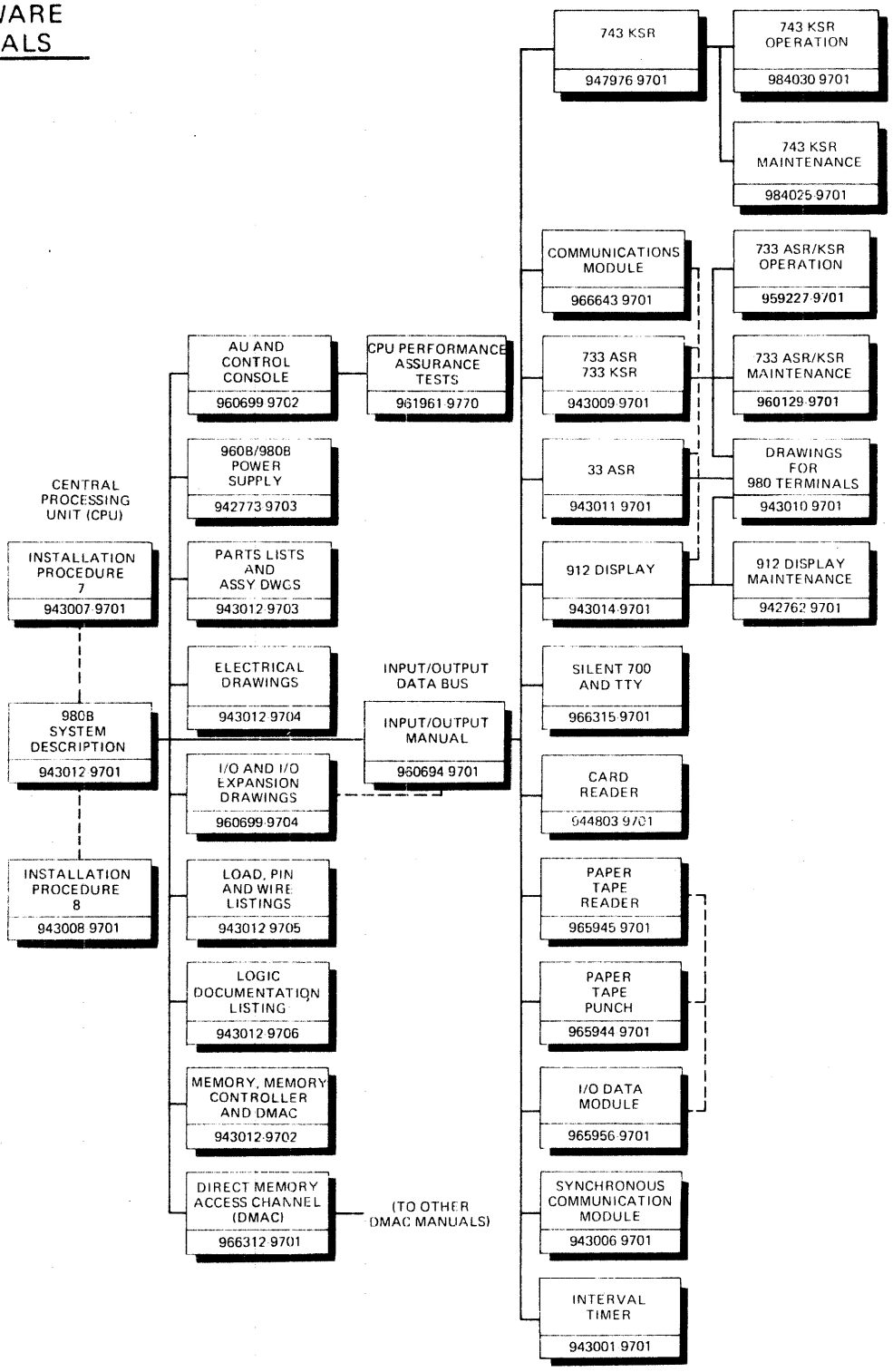
TEXAS INSTRUMENTS INCORPORATED
DIGITAL SYSTEMS DIVISION

P.O. BOX 2909 · AUSTIN, TEXAS 78769

ATTN: TECHNICAL PUBLICATIONS
MS 2146

FOLD

980B COMPUTER SYSTEM HARDWARE MANUALS



TEXAS INSTRUMENTS
INCORPORATED
DIGITAL SYSTEMS DIVISION
POST OFFICE BOX 2909 AUSTIN, TEXAS 78769