

901593A
FIELD ENGINEERING
MAINTENANCE DOCUMENTATION

Price: \$12.50

SIGMA 3 CENTRAL PROCESSOR UNIT (CPU)

MODEL 8101/8102

TECHNICAL MANUAL

PRELIMINARY RELEASE

XEROX

TO THE CUSTOMER ENGINEER:

In this attempt to make your job easier by providing you with good, usable documentation, we must have feedback from you to test its validity.

You can help us in this effort by taking a few minutes to complete the following survey and returning it to us.

① Which of the following terms best describes your job:

Systems Engineer	<input type="checkbox"/>
Diagnostician	<input type="checkbox"/>
Instructor	<input type="checkbox"/>
Customer Engineer/Field Engineer	<input type="checkbox"/>
Trainee	<input type="checkbox"/>

② Do you feel that the material was:

	Yes	No
Easy to read	<input type="checkbox"/>	<input type="checkbox"/>
Well organized	<input type="checkbox"/>	<input type="checkbox"/>
Fully covered	<input type="checkbox"/>	<input type="checkbox"/>
Well illustrated	<input type="checkbox"/>	<input type="checkbox"/>
Clearly explained	<input type="checkbox"/>	<input type="checkbox"/>

③ How did you use this manual:

As an introduction to the subject	<input type="checkbox"/>
For additional knowledge of the subject	<input type="checkbox"/>
Maintenance	<input type="checkbox"/>

④ If you feel that specific changes should be made, check the appropriate block and cite the changes in the comments section:

Suggested change (page ____)	<input type="checkbox"/>
Error (page ____)	<input type="checkbox"/>
Suggested addition (page ____)	<input type="checkbox"/>
Suggested deletion (page ____)	<input type="checkbox"/>

⑤ COMMENTS: (If there is not enough space here for your comments, use the back of this sheet)

Please complete this survey and return to:

MANAGER, MAINTENANCE AND SELF-STUDY
DOCUMENTATION
MAIL STOP C2-16, AIRPORT FREEWAY CENTER
5250 W. CENTURY BOULEVARD
LOS ANGELES, CALIFORNIA 90045

TABLE OF CONTENTS

Section		Page	Section		Page
I	INTRODUCTION		II	FUNCTIONAL DESCRIPTION (Contd)	
	1-1 Scope of Manual	1-2	(Contd)	2-40 DIO Interface (CPU Interface Function)	2-11
	1-2 Content of Manual	1-2		2-46 Adder Function	2-12
	1-3 Related Publications	1-2		2-52 Adder	2-13
	1-4 Prerequisites to Manual Use	1-2		2-66 J-Bus Function	2-18
	1-5 Sigma 3 Central Processor Unit (CPU)	1-2		2-67 J-Bus and Arithmetical Register Group	2-18
	1-6 Physical Description	1-2		2-75 D-Register (Register Function)	2-20
	1-7 CPU Structure	1-3		2-85 Parity Check/Generation Section	2-22
	1-8 Capabilities	1-3		2-88 H-Register (Register Function)	2-23
	1-9 Memory Interface	1-3		2-96 S-Register (Register Function)	2-24
	1-10 Input/Output Interface (I/O)	1-3		2-103 LA-Register (Register Function)	2-25
	1-11 Direct Input/Output Interface (DIO)	1-5		2-109 LG and LM-Registers (Register Function)	2-26
	1-12 CPU Options	1-5		2-110 LM-Register	2-26
	1-13 Power Fail-Safe Option	1-5		2-117 LM-Register	2-27
	1-14 Real-Time Clock Interrupts	1-5		2-121 RA-Register (Register Function)	2-28
	1-15 Additional I/O Channels	1-5		2-131 General and I/O Channel Registers (Register Function)...	2-30
	1-16 Protect Feature	1-6		2-137 FRL-Register (Register Function)	2-32
	1-17 Extended Arithmetic	1-6	III	DETAILED PRINCIPLES OF OPERATION	
	1-18 Integral Interrupts	1-6		3-1 Introduction	3-1
	1-19 External Interface	1-6		3-2 Logic Diagrams	3-1
	1-20 General Specifications for CPU	1-6		3-7 PCP Mode of Operation	3-1
	1-22 Sigma 3 System Power Requirements	1-6		3-8 System Operation Procedure - PCP Control	3-1
II	FUNCTIONAL DESCRIPTION			3-10 Fetching and Storing Data - PCP Control	3-14
	2-1 Central Processor Unit (CPU)	2-1		3-12 Program Control Operation	3-17
	2-9 Functional Description	2-1		3-16 Instruction Preparation Sequence	3-17
	2-10 CPU Data Flow	2-2		3-22 Source of Next Instruction Address	3-18
	2-15 Processor Control Panel (PCP) Function	2-4		3-27 Instruction Execution Sequence	3-30
	2-18 Clock Generation and Control Function	2-6		3-29 Direct Control Instruction	3-30
	2-23 Instruction Decode/Execution Function	2-8		3- Memory Reference Instruction (To be added)	
	2-28 Memory Interface (CPU Interface Function)	2-9		3- FUCPY and SHIFT Instructions (To be added)	
	2-33 I/O Interface (CPU Interface Function)	2-10			

ILLUSTRATIONS

Figure	Title	Page	Figure	Title	Page
1-1	Sigma 3 Central Processor Unit (CPU), Location in 8101/8102 Model Cabinet	1-1	2-17	General and I/O Channel Registers (Fast Memory) Data Flow Diagrams .	2-31
1-2	Relation of the CPU to the Sigma 3 Computer System	1-3	2-18	FRL-Register Data Flow Diagram	2-32
1-3	Sigma 3 CPU Configuration with Options	1-4	2-19	Interrupt Priority Control Function, Block Diagram (To be added)	
2-1	Sigma 3 Central Processor - Functional Block Diagram, Data Flow	2-3	2-20	Memory Protect Control Function, Block Diagram (To be added)	
2-2	Processor Control Panel (PCP) Function, Sheet 1 of 2	2-4	3-1	Processor Control Panel - PCP Control Mode of Operation	3-4
2-2	Processor Control Panel (PCP) Function, Sheet 2 of 2	2-5	3-2	Power Distribution Diagram (2 Sheets)	3-5
2-3	Clock Generation and Control Function, Block Diagram	2-7	3-3	Conditions for Normal Mode Indication	3-7
2-4	Instruction Decode/Execution Control Function, Block Diagram	2-8	3-4	Data Flow During Prep Phase of SIO (Bootstrap)	3-8
2-5	CPU Interface Function, Memory Interface Block Diagram	2-9	3-5	Data Flow During Execution of SIO (Bootstrap)	3-9
2-6	CPU Interface Function, I/O Interface Block Diagram	2-10	3-6	I/O Service Call Data Flow Diagram	3-13
2-7	CPU Interface Function, DIO Interface Block Diagram	2-11	3-7	Source of Next Instruction Address at ENDE	3-19
2-8	Adder Function, Data Flow Diagram	2-12	3-8	Instruction Prep Sequence - Data Flow	3-29
2-8a	Adder Operation - Addition	2-13	3-9	Direct Control Instruction Execution - Data Flow	3-39
2-8b	Adder Operation - Subtract/Compare	2-14	3-10	Memory Reference Instruction Execution - Data Flow (To be added)	
2-8c	Adder Operation - H-Minus-1	2-14	3-11	FUCPY and SHIFT Instructions Execution - Data Flow (To be added)	
2-8d	Adder Operation - Logical AND	2-15	3-12	Logic Diagrams (Sheet 1 through 91)	3-40
2-8e	Adder Operation - Logical Inclusive OR	2-15		PCP Panel and Clock Control -	Sheets 1-13
2-8f	Adder Operation - Logical Exclusive OR	2-16		ENDE Control -	14
2-8g	Adder Operation - D-Register Transfer to J-Bus	2-16		Memory Interface Control and RSTE -	15, 16
2-8h	Adder Operation - H-Register Transfer to J-Bus	2-17		Prep Phase Control -	17, 18
2-8j	Adder Operation - Complement of H to J-Bus	2-17		Instruc Decode Control -	19-22
2-9	J-Bus and Arithmetical Register Group, Data Flow Diagram	2-19		Instruc Phase Control -	23-26
2-10	D-Register Data Flow Diagram	2-21		SW1 and SW2 Control -	27, 28
2-11	Parity Check and Generation, Data Flow Diagram	2-22		Status Indicator Control -	29-31
2-12	H-Register Data Flow Diagram	2-23		Adder Control -	32-40
2-13	S-Register Data Flow Diagram	2-24		J-Bus and Control -	41-43
2-14	LA-Register Data Flow Diagram	2-25		D-Register and Control -	44-49
2-15	LG-and LM-Registers, Data Flow Diagram	2-27		Parity Check/Generation Control -	50, 51
2-16	RA-Register Data Flow Diagram	2-29		H-Register and Control -	52-55
				S-Register and Control -	56-59

ILLUSTRATIONS (Contd)

Figure	Title	Page
3-12	Logic Diagrams (Contd)	
	LA-Register Control -	Sheets 60,61
	RA-Register and Control -	62-66
	General/IO Channel (Fast Mem) Control -	67-69
	LG-Register and Control -	70
	LM-Register and Control -	71
	IO Interface Control -	72-82
	DIO Interface Control -	83-88
	1024 KHZ Oscillator -	88
	Instruction - Address/Operand Stop -	89
	Sync Counter & Auto Reset -	90
	Console Interrupt -	91

(Further Interrupt Control and Memory Protect Logic to be Added)

TABLES

Table	Title	Page
1-1	Sigma 3 CPU Model and Options	1-5
1-2	General Specifications	1-6
1-3	Sigma 3 System Power Requirements	1-6
3-1	System Operation Procedures	3-1
3-2	IO Service Connection Sequence Control	3-10
3-3	Fetching and Storing Data Procedures	3-14
3-4	Instruction Preparation - Clock-Step Procedure	3-18
3-5	Instruction Prep Phase Sequence Control (9 Sheets)	3-20
3-6	Direct Control Instruction Execution Sequence Control (8 Sheets)	3-31
3-7	Memory Reference Instruction Execution Sequence Control (To be added)	
3-8	FUCPY and SHIFT Instruction Execution Sequence Control (To be added)	

RELATED PUBLICATIONS

Publication Title	Document No.
Sigma Computer Systems Interface Design Manual	900973
Sigma 3 Computer Reference Manual	901592
Sigma 3 CPU Model 8101A01/8102A01, Illustrated Parts Breakdown	901763
Sigma 3 Core Memory, Model 8151/8152/8155, Technical Manual	901954
Sigma 3 External IO Processor, Model 8171, Technical Manual	901595
Sigma 3 Central Processor, Model 8101, Assembly No. 153253, Engineering Support Manual	902400
Sigma 3 Core Memory, Model 8151/8152/8155, Engineering Support Manual	902401
Sigma 3 External IO Processor, Model 8171 Engineering Support Manual	902402
Sigma 2/3 Memory Diagnostic Manual	900676
Sigma 2/3 Central Processor Unit, Interrupt Diagnostic Manual	901137
Sigma 2/3 Power Fail Safe Test Diagnostic Manual	901160
Sigma 2/3 Real Time Clock Test Diagnostic Manual	901164
Sigma 2/3 CPU Watchdog Timer Test Diagnostic Manual	901571
Sigma 3 CPU Extended Arithmetic Diagnostic Manual	901589
Sigma 3 Memory Diagnostic Fault Locator, Diagnostic Manual	901604
Sigma 3 CPU Diagnostic Auto Diagnostic Manual	901608
Sigma 3 Multiport Memory Random Exerciser Diagnostic Manual	901615
Sigma 3 External IO Processor Diagnostic Manual	901646
Sigma 3 Internal IO Processor Test, Diagnostic Manual	901659
Power Supply, PT14B Data Package	901655
Power Supply, PT15B Data Package	901656
Power Supply, PT16B Data Package	901657
Power Supply, PT17B Data Package	901658
Installation Manual (Tentative)	901716

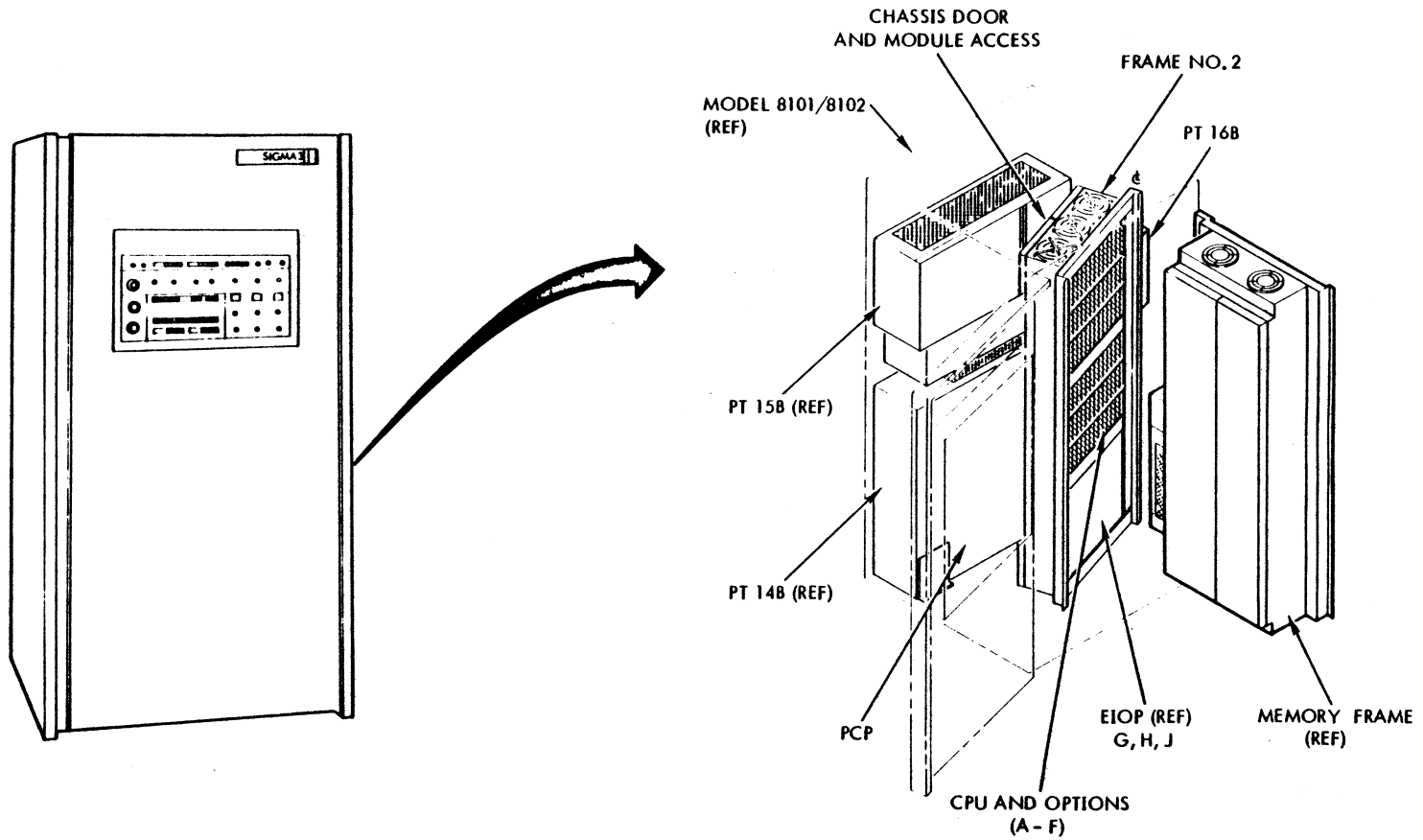


Figure 1-1. Sigma 3 Central Processor Unit (CPU), Location in 8101/8102 Model Cabinet

SECTION I INTRODUCTION

1-1 SCOPE OF MANUAL

This manual contains technical information pertinent to the Sigma 3 Central Processor Unit (CPU) which is one of the main units included as part of the Sigma 3 Computer System. This manual describes the Sigma 3 CPU both physically and functionally as applied in the Sigma 3 Computer system.

This manual also includes information necessary to maintain the central processor in the field of operation.

1-2 CONTENT OF MANUAL

The contents of this manual are sectionalized as follows: Section I contains an introduction to the equipment covered by the manual and to the method and type of coverage.

Section II contains a general description of the CPU operation. The overall mode of operation is divided into several functional groups of logic and each group is analyzed in the order of data transfer to, within, and from the CPU.

Section III provides detailed descriptions of several modes of operations performed by the CPU. These are presented in flow diagrams which are referenced to logic diagrams located in the back of Section III.

1-3 RELATED PUBLICATIONS

A list of related publications, page, preceding Section I, is provided as additional reference material essential to the maintenance personnel to insure a complete awareness of proper installation, operation and maintenance of the central processor unit relative to the Sigma 3 Computer system.

1-4 PREREQUISITES TO MANUAL USE

In the construction of this manual, it is assumed the user is familiar with the functions of transistorized digital circuitry and can interpret MIL-STD-806B logic symbology. It is also assumed the user can interpret logic functions expressed in logic equation form and is familiar with machine language programming.

It is essential the user is capable of using standard test equipment in diagnosing and repairing hardware malfunction associated with digital equipment and has experience in the operation and troubleshooting of magnetic core memories.

1-5 SIGMA 3 CENTRAL PROCESSOR UNIT (CPU)

1-6 PHYSICAL DESCRIPTION

The Sigma 3 CPU is located in the upper 6 chassis located in the center swing frame of the Sigma 3 Computer System cabinet, models 8101/8102 as shown in figure 1-1. This section of the swing frame also provides module locations for the installation of options which are available with the CPU. Refer to table 1-1. Sigma 3 system operation is performed at the Processor Control Panel (PCP) located in the front door of the cabinet. This control panel is included in the CPU configuration.

In some Sigma 3 system configurations, an external I/O processor (EIOP) is substituted for the integral I/O processor (IIOP) option or is in addition to the IIOP. In either case, the EIOP is located in the lower 3 chassis in the center swing frame while the IIOP is located within the area of the CPU. The Sigma 3 system requires one or the other I/O processors, however, both may be added if desired.

The PT16B power supply mounted on the center swing frame provides regulated dc for the center frame logic. The PT16B obtains operating power through a junction box from the PT14B/PT15B power supply combination located in the rear of the same cabinet.

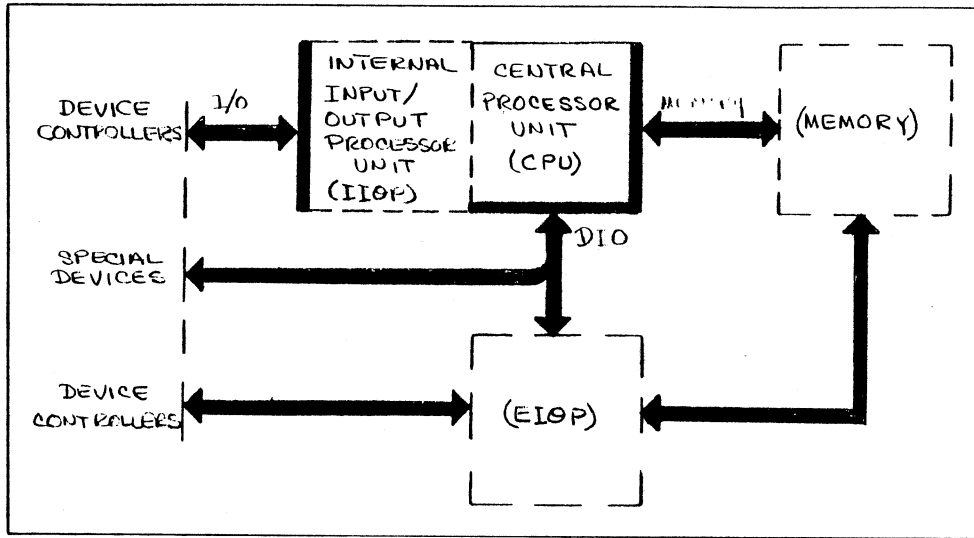


Figure 1-2. Relation of the CPU to the Sigma 3 Computer System

1-7 CPU STRUCTURE

The center swing frame which supports the CPU and its options in the Sigma 3 cabinet provides for 9 module chassis, each chassis with slots for a maximum of 32 modules. Modules plug into a back wiring board for circuit intraconnection. Ribbon cables and connectors are used to carry information between chassis and frames. Coax cables and connectors are used for carrying information outside the cabinet or when it is necessary to minimize noise pickup, or prevent signal distortion and also to simplify the mechanical aspects of cabling. The CPU and the available options are located in chassis A through F. Refer to figure 1-3 for the physical configuration.

1-8 CAPABILITIES

The CPU provides logic to execute instructions, to perform arithmetical and logic operations and to interface with core memory. In addition to programmed operation, the Sigma CPU is manually operated by using switches provided on the processor control panel (PCP), four coaxial cable connectors are provided for this interface, refer to figure 1-3. When the external interface option is installed, the CPU provides direct input/output interface (DIO) with special devices. In the Sigma 3 configuration the EIOP is connected to the DIO interface along with special devices. Refer to figure 1-2.

In addition to the memory and DIO interfaces, the CPU in conjunction with the IIOU option, provides an interface with input/output peripherals through device controllers. The EIOP interfaces with a separate group of peripherals through other device controllers. The EIOP is a high performance I/O processor that has its own logic and memory bus to minimize interference with CPU operation. For detailed information regarding the Sigma 3 EIOP and core memory, refer to Technical Manuals 902402 and 901594, respectively.

1-9 MEMORY INTERFACE

When interfacing with memory, the CPU addresses core, fetches and stores information in core, checks and generates data parity, monitors the power fail-safe signals (when this option is installed) and provides memory interface control signals. The CPU is connected to memory by four ribbon cables from chassis A, slots 32, 31, 30 and 29. Ribbon cables from slots 32A and 31A are to the 32K within the cabinet containing the CPU; slots 30A and 29A contain ribbon cables to a second 32K contained in an additional cabinet. Refer to figure 1-3 for ribbon cable locations.

1-10 INPUT/OUTPUT INTERFACE (IIOU)

When concerned with the I/O interface by utilizing the IIOU option, the CPU addresses a peripheral through a device controller and, with this option installed, the CPU controls the transfer of information between peripherals and memory in increments of one-byte (8 bits). A minimum of one byte or a maximum of 4 bytes (two words) can be transferred with each service call. Figure 1-3 shows the locations of the 4 coaxial cable connectors in slots 30B, 28B, 27C and 25C between the CPU and device/subcontroller to provide the paths for data and control signals.

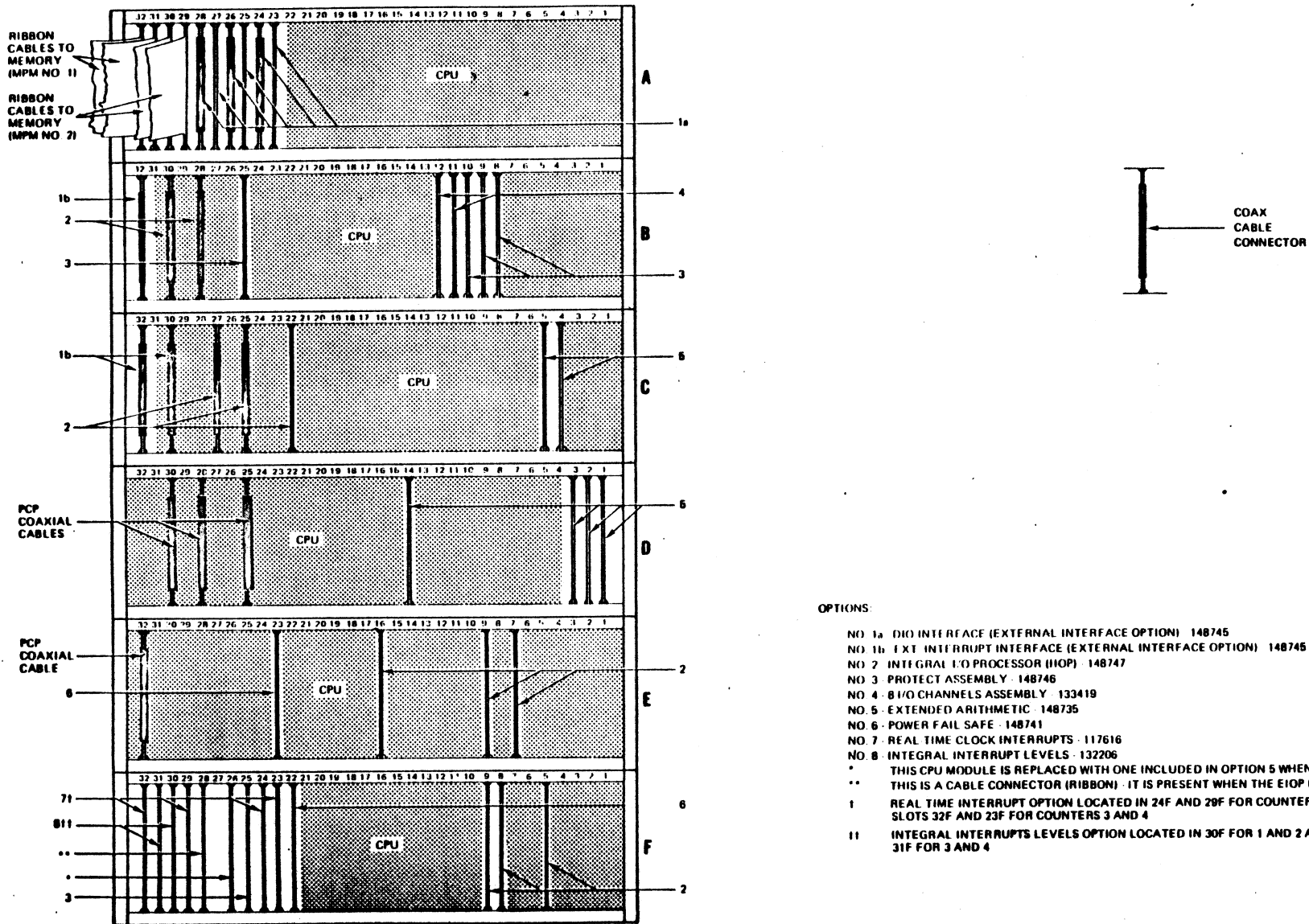


Figure 1-3. Sigma 3 CPU Configuration with Options

Table 1-1. Sigma 3 CPU Model and Options

<u>Model</u>	<u>Specification Nomenclature</u>	<u>Common Name</u>	<u>Assembly No.</u>
8101/8102	Central Processor Unit (CPU)	CPU	148586
8105	Integral IOP Assembly	IIOF	148747
8111	Real-Time Clock Assy	Real-Time Clocks	117616
8113	Power Fail-Safe Assy	Power Monitor Interrupts	148741
8114	Protect Assy	Memory Protect	148746
8119	Extended Arithmetic Feature Assy	Multiply/Divide Option	148735
8122	Interrupt Assy 2 Level	2 Integral Interrupts	132206
8170	External Interface Assy	DIO and External Interrupt Interfaces	148745
8172	Eight I/O Channel Assy	IIOF Channel Expansion	133419

1-11 DIRECT INPUT/OUTPUT INTERFACE (DIO).

When the external interface option is installed (paragraph 1-19), the CPU can interface directly with a device without the use of an I/O channel. This interface is provided for special devices (e.g. an A-D converter) and for communication with EIOP. With this option the CPU uses bi-directional lines to transfer an address and one word of information between the special device or the EIOP and a register within the CPU. This one word of information can then be manipulated by the CPU as directed by a program stored in memory. Figure 1-3 shows the location of the 3 coaxial cables between the CPU and the DIO interface in slots 28A, 26A and 24A. Additional DIO logic is contained in slots 27A, 25A and 23A. A ribbon cable is added in slot 28F and a clock coax cable from 32D when an EIOP is connected to the DIO interface.

1-12 CPU OPTIONS

In addition to the two options required to establish the I/O and DIO Interfaces, there are 7 more options which are available to expand the capabilities of the Sigma 3 CPU basic system. These are generally described in the following para 1-13 through 1-19.

1-13 POWER FAIL-SAFE OPTION

This option consists of two interrupt levels used to enter programmed routines that save and restore volatile information in the event of a power failure. The power-off interrupt level is triggered when the power supply voltage falls below a safe limit; the power-on interrupt level is triggered when power returns to safe limits. This option consists of 2 modules. Refer to figure 1-3 for the location of the modules.

1-14 REAL-TIME CLOCK INTERRUPTS

Part of this option consists of two real-time counters to meet the needs of real-time systems where timing information must be provided. These counters can be used for elapsed time after a given event or the current time of day. The counters can be driven from the facility a-c line frequency, from the 500 hertz, 2K or 8K hertz oscillators within the system, or from an external source. (Note: Counter number 1 is backwired to 500 hertz). The remaining part of this option provides two internal interrupt levels used to enter a programmed routine when a counter reaches zero. This option is contained on two modules. Refer to figure 1-3 for locations.

1-15 ADDITIONAL I/O CHANNELS

This option provides 8 more I/O channels to expand the basic IIOF configuration from 4 channels to 12 channels. This option consists of 2 modules. Refer to figure 1-3 for the location of the modules.

1-16 PROTECT FEATURE

This option provides the logic to protect segments of memory from the possibility of cross-over when the CPU is operating on both a background and foreground program. This option also includes a module containing logic for an interrupt level used to enter a programmed routine if the protected boundaries in memory have been violated. In addition to the memory protect interrupt level, the same module contains a machine fault interrupt level. This optional interrupt level allows the CPU to enter a subroutine if a memory parity error occurs or there is an interface timer runout condition during an external I/O, an integral I/O service call, or a DIO timeout.

1-17 EXTENDED ARITHMETIC

This option provides the additional logic required for executing the multiply, divide and multiple precision mode instructions. If this option is not installed, the multiply/divide exception interrupt level (standard) is triggered whenever an attempt is made to perform a multiply or divide instruction. This option consists of modules. Refer to figure 1-3.

1-18 INTEGRAL INTERRUPTS

This option provides the logic for two interrupt levels (per module) which can be triggered by an external source. A maximum of four levels (2 modules) may be installed. This option is usually installed when a Sigma 3 system configuration does not justify the installation of a full external interrupt system. Refer to figure 1-3 for location.

1-19 EXTERNAL INTERFACE

One portion of this option provides the logic for the DIO Interface described in paragraph 1-11. The remaining portion provides the interface logic required for the installation of an external interrupt system. The external interrupt interface consists of 3 coax cable connectors, refer to figure 1-3 for the slot locations.

1-20 GENERAL SPECIFICATIONS FOR CPU

1-21 General Specifications for the Sigma 3 Central Processor (CPU) are given in table 1-2.

Table 1-2. General Specifications

Logic circuit operating voltage	+8 VDC -8 VDC +4 VDC	} Supplied by PT16B
Frame fan operating power	120v, 60 hertz	
Logic signal levels Coax cable voltage levels	One, 2.5V Min. - 4.4V; zero, 0V-1.5V Max. One, 1.4V Min. - 4.4V; zero, 0V-0.44V Max.	
Word length	16 bits plus parity bit	
Data format	Two 8-bit parts (bytes) byte 0 and byte 1; 16 bit word. Fixed point in two's complement; 15 bit integer and sign bit in zero position for arithmetic operations. For logical operations, 16 bits without sign.	
Code	Binary expressed in hexadecimal notation.	

1-22 SIGMA 3 SYSTEM POWER REQUIREMENTS

1-23 The two main sources of power for the Sigma 3 system are the PT14B and PT15B located in the rear of the cabinet. The ac output from the PT15B is required by both the PT16B on the center swing frame and the PT17B on the memory frame. Refer to table 1-3 for system power requirements.

Table 1-3. Sigma 3 System Power Requirements

		<u>Input</u>	<u>Output</u>
PT14B	AC-DC converter	3Ø, 120/208v, 60Hz.	120V, 60Hz. 60VDC
PT15B	DC-AC converter	60VDC *120V, 60Hz.	109V(aver.)1800 Hz.
PT16B	DC regulator	109V(aver.)1800Hz. *120V, 60Hz	+8 (0-55A-Max) -8 (0-12A-Max) +4 (0-110A-Max)
PT17B	DC regulator	109V(aver.)1800Hz.	+24VDC +17 to +25VDC
*Fan power		*120V, 60Hz.	120V, 60HZ.

SECTION II
FUNCTIONAL DESCRIPTION

2-1 CENTRAL PROCESSOR UNIT (CPU)

The Central Processor Unit (CPU) provides the logic circuits to perform all functions carried out by the arithmetical and control sections of the Sigma 3 Computer System. The arithmetical section performs all arithmetical calculations and some logical operations. The control section acts as the master dispatcher and clock of the computer. Information words are selected in sequence from a stored program in memory under the direction of the control section which coordinates all operations, times the operations, and controls the flow of information through the CPU.

2-2. To communicate with the outside world, the CPU is provided with an input/output processor (IOP) interface option which becomes an integral part of the CPU inasmuch as it shares some of the CPU logic. This option allows for CPU/memory connection to on-line peripheral devices through their associated controllers. Another option allows for direct input/output communication with a special device, for example; an analog-to-digital converter. This interface (DIO) is also used for CPU communication with an external I/O processor (EIOP) also an option to the CPU.

2-3. Refer to figure 2-1. The overall performance of the CPU and its options is accomplished by logic circuits which can be grouped into nine separate functions. The arithmetical section includes the register, J-bus, and the adder functions. For the transfer or computation of data, this section operates in data-flow loops of register, to adder, to J-bus, to register, or, register to J-bus, to register, to adder, to J-bus. The control section is covered by the CPU interface, the clock generation and control, the instruction decode/execution control and the processor control panel (PCP) functions.

2-4. The CPU interface function is covered in paragraph 2-1 and 2-2. The clock generation and control function provides the circuits to generate a 6 Mhz master clock and a phase-sync which divides the master clock into clock phase A and clock phase B, each phase generated every other cycle of the master clock. The master clock and the sync for phase B clock are output to external units to generate their internal clocks (memory or EIOP). Phase A is used to generate clocks used within the CPU. These clocks are started and stopped by the setting of switches on the PCP, during program execution, or program interrupt.

2-5. The instruction decode/execution control function provides a register to receive the operation and address computation codes of the instruction, and the logic to decode this information. The interpretation of these codes provides the control for instruction execution. This function also uses the internal clocks to generate timing phases to synchronize each step of the instruction execution with the basic CPU clock.

2-6. Data flow to the PCP panel is for display purposes, data flow from the PCP panel is to the J-bus for input to registers and perhaps on to memory. This data transfer takes place when the CPU is idle. The logic circuits and switches grouped in the PCP control function control this flow of data. Switches on the PCP are provided for Sigma 3 System control.

2-7. The CPU also includes logic circuits (when the appropriate options are available) which provides for processing internal and external program interrupts, and memory protect interrupts. These functions are also directed by the control section of the CPU. The priority of the incoming interrupt is established by the interrupt priority control function. When an interrupt arrives, the information lines from this function will carry the address in memory, dedicated to the routine servicing the interrupt. These lines can also carry the state of the interrupt levels as read by an instruction requesting this information to be loaded into a general CPU register. The state of interrupt levels can also be altered in accordance with the contents of a register, by an instruction requesting this operation.

2-8. The memory protect control function includes logic circuits and a stack of general registers which supply the decode logic and memory protect registers assigned to each 4K of memory. Data flow to this function are the first 8 bits (0 through 7) contained in the memory address register at the beginning of each memory cycle. This information is decoded, to first select the register which contains information on a protected area of memory, and then determine if the address of the instruction or operand infringes upon a particular part of a protected area. These lines can also contain a protected-area-information word to be loaded into a particular memory protect register in accordance with an instruction requesting this operation.

2-9 FUNCTIONAL DESCRIPTION

All data in the CPU is transferred in a parallel configuration within, and to the memory, IOP, and DIO interface cables. The register function of the CPU provides all logic circuits which hold, for transfer, useful information in a parallel form. The registers, which are a group of ac flip-flops or buffered latches, hold from 6 to 16 bits of data. The J-bus holds data in a 16-bit parallel line configuration using a combination of the exclusive OR (for adder output to the J-lines) and the AND/OR logic for all of the other J-bus line data input.

2-10 CPU DATA FLOW

Refer to figure 2-1. Data flow through the CPU begins with the control section initiating the operation to fetch an instruction from memory. This is performed by placing the address of the instruction in the memory address register and out on the memory address lines provided by the memory interface control logic. A memory cycle is initiated and the instruction is returned on the read-data lines (16). The instruction is then transferred to the memory/arithmetical section interface register, the D-register. At the same time, the first half of the instruction, which contains the operation and address modification codes, is also placed in the control section where this information is interpreted by the logic performing the instruction decode/execution function.

2-11. As the control section interprets the first portion of the instruction (bits 0-7) control signals are generated which direct, and time, the flow of data in the second portion of the instruction (displacement field, bits 8-15). This data is directed through the arithmetical section for computation of the effective address of the operand called for by the instruction. When the address is obtained, and the instruction is a memory reference type, the address is placed on the memory address lines and a memory cycle is again initiated. The control section continues to execute the instruction as follows: If the instruction is to store information contained in a CPU general register, data is transferred to the D-register for input on the write-data lines (16) to the address on the memory address lines. If the instruction is to perform an arithmetical computation with the contents, or transfer the contents of the address to a general register, the information is placed on the read-data lines and transferred to the D-register. From here on the data is manipulated according to the operation and the result either winds up in memory or is loaded into a general register in the CPU.

2-12. If the instruction is other than a memory reference type instruction, a memory cycle is inhibited. The entire instruction (16 bits) is interrogated by the control section and the instruction is executed according to the code bits contained in the instruction. For example, if the instruction is an I/O instruction, the area of the displacement field carries the I/O function indicator bits. These are output to a register which interfaces with the I/O and are transmitted to a device controller. The device will respond with its status on the function response lines (8). When an I/O service call is issued, the device address is transferred to the I/O data lines (8) from a register in the I/O interface logic. Also, I/O data from or to memory is placed on these I/O data lines.

2-13. If the instruction is to the DIO interface, the special device address (16) or the I/O function bits for the EIOP are placed on the DIO address lines. If data is sent to a special device, the data is on the DIO data lines (16). If the data is to be sent from the addressed special device, the data is returned on the DIO data lines. If the EIOP is responding to an I/O function, the response (status) is on the DIO data lines. The address of the device at the EIOP is also placed on the data lines at the same time the function bits are placed on the DIO address lines.

2-14. The register function of the CPU also includes the parity check/generation control circuits. This control is associated only with the memory/arithmetical section interface register, the D-register. Parity is checked and generated on data in the D-register, each memory cycle.

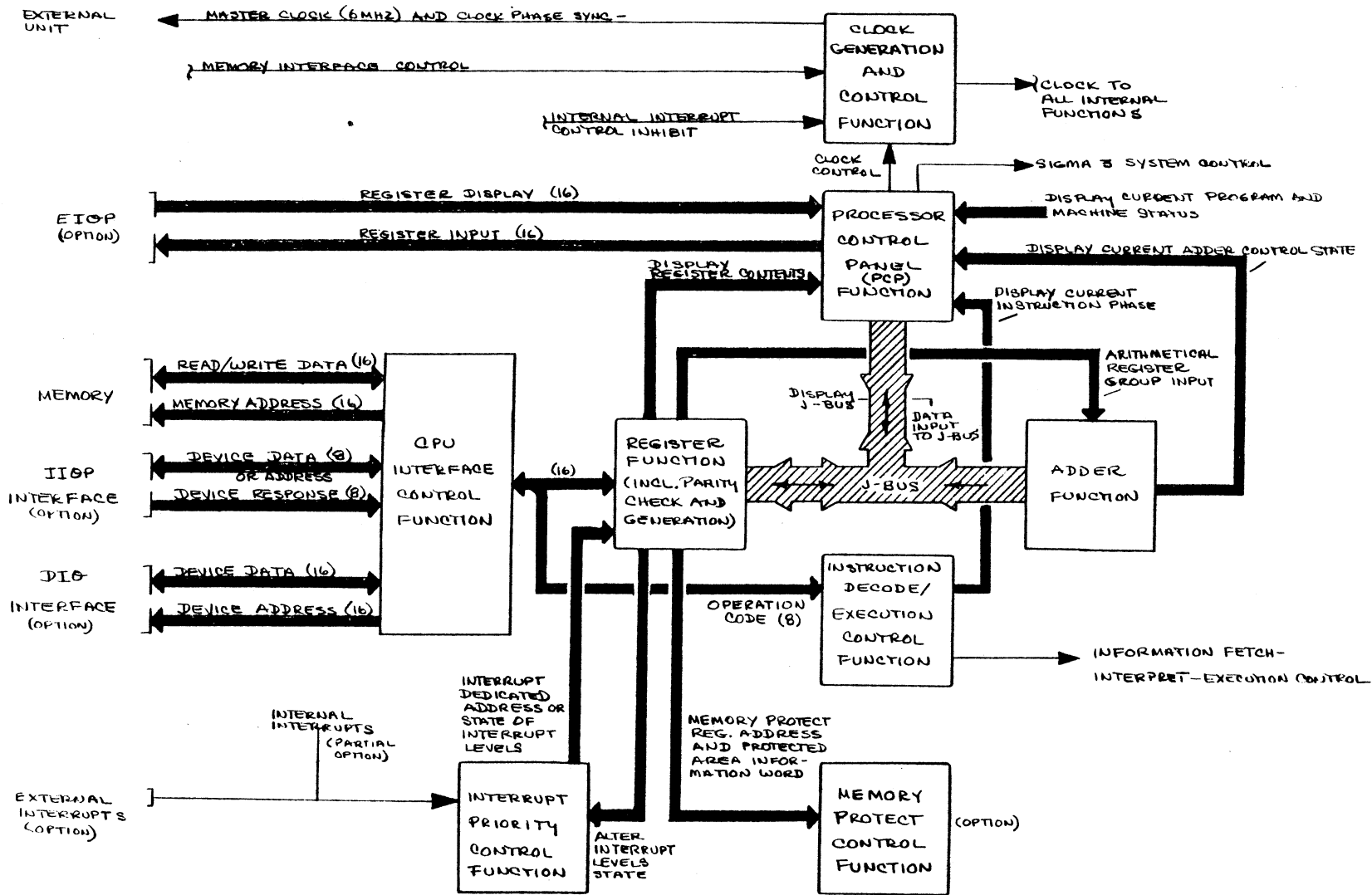


Figure 2-1. Sigma 3 Central Processor - Functional Block Diagram, Data Flow

2-15 PROCESSOR CONTROL PANEL (PCP) FUNCTION

The PCP function of the CPU is part of the Sigma 3 control section. This function is performed by the switches and indicators on the processor control panel (refer to figure 2-2, sheet 2) and the logic circuits which drive the indicator lamps and gate the logic levels established by the setting of the switches. These gated logic levels are transferred as data to registers or to activate the indicator lamp driver circuits or control circuits in other areas of the CPU.

2-16. Refer to figure 2-2, sheet 1. The circuits used to perform the PCP function can be divided into two groups; the control and data-enter switch group, and the data display control group. The first group provides the switches for activating the Sigma 3 System and loading and running a program, performing system reset, or allowing operator control of an interrupt routine. Other switches in this group provide logic levels which allow for manual control of the CPU clock, loading or displaying CPU or EIOP registers, addressing and storing in memory and stepping through a program or an instruction. Switch settings (of a group used specifically during fault isolation procedures) effect control circuits (while the CPU is running) which will cause an automatic stop or system reset under selected conditions.

2-17. The display group consists of all indicator lamps and the associated drive circuits. The drive circuits are selected by switches on the PCP. All data for display is input to the indicator drive circuits which are gated by the logic levels set by the select switches. Specific program status indicator drive circuits (10) also part of the display group, are gated by controls generated while executing an instruction and not by setting select switches. Figure 2-2, sheet 1 shows the type of input to the display group of the PCP function.

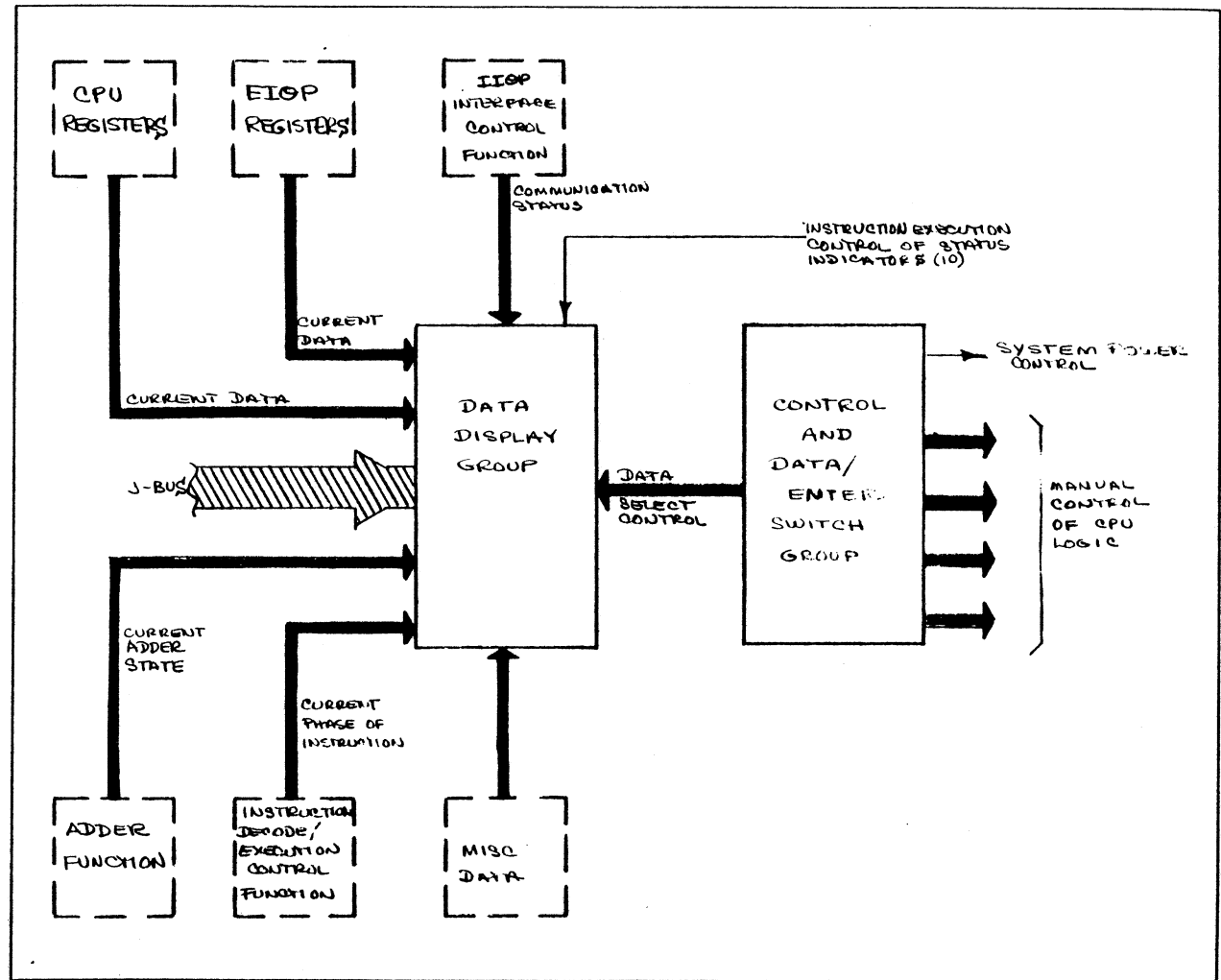


Figure 2-2. Processor Control Panel (PCP) Function (sheet 1 of 2)

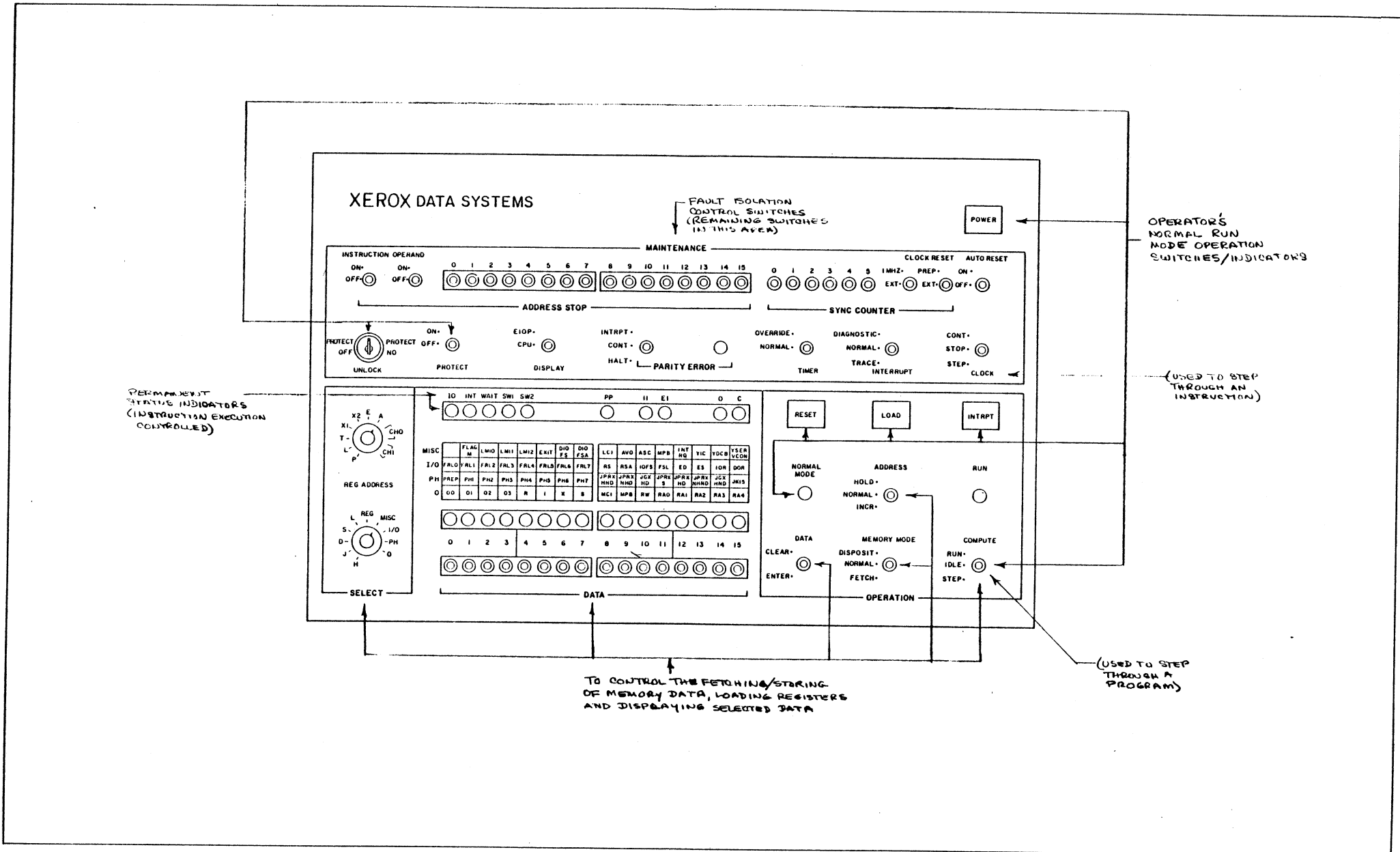


Figure 2-2. Processor Control Panel (PCP) Function set 2 of 2)

2-18 CLOCK GENERATION AND CONTROL FUNCTION

The clock generation and control function of the CPU is part of the Sigma 3 control section. In addition to the POWER switch on the PCP, the starting and stopping of computer operation is controlled by the logic circuits included in the clock generation and control function. Since every data transfer or adder computation must occur precisely on time, it is necessary to synchronize the sequence of operations, therefore a master clock and timing pulses are provided.

2-19. Refer to figure 2-3. A crystal or LC controlled oscillator in the CPU develops stable sine waves of 6 Mhz. Wave-shaping circuits contained on the oscillator module provide rectangular clock pulses from the output of the oscillator and another circuit divides this master clock in a 2:1 ratio and develops a 3 Mhz rectangular pulse. These pulses, output from the master clock module, are input to a phase sync generator that provides logic levels to sync the master clock every other pulse and produce a clock phase A and phase B.

2-20. The master clock and the not-sync A level is provided to memory and to the EIOP for internal clock generation. The master clock and sync A are input to an internal clock driver circuit to provide clock for CPU fast memory and for resetting (erase) buffered latches. This clock pulse is extended (widened) by using a delayed fast memory clock pulse as a control input to the flip-flop clock drive circuit. This extended clock pulse assures better control of data transfer operations. Among the flip-flops controlled, are those in the instruction decode/execution control logic which generate the timing pulses to control the sequence of instruction execution.

2-21. Finally, not-sync A is input to a clock driver circuit that provides clocks to the interrupt priority level control circuits. These clocks are always present except during instruction prep phase and when controlled by the interrupt service entry routine inhibit.

2-22. Clock drive circuits which output clocks to fast memory and all other register and logic circuits in the CPU are started or stopped by a clock latch circuit. Clocks are permitted to RUN (normal operation) or clocks can be controlled in single or short bursts when performing manual control of the CPU circuits. Clocks can be stopped automatically if selected to do so by a PCP maintenance control switch, and clocks can be inhibited by memory interface control to synchronize memory/CPU communication. The lamp driver circuit to the RUN indicator on the PCP is controlled by the output of the clock latch circuit. The RUN lamp is on only when the latch has been lifted from the CPU clock driver circuits.

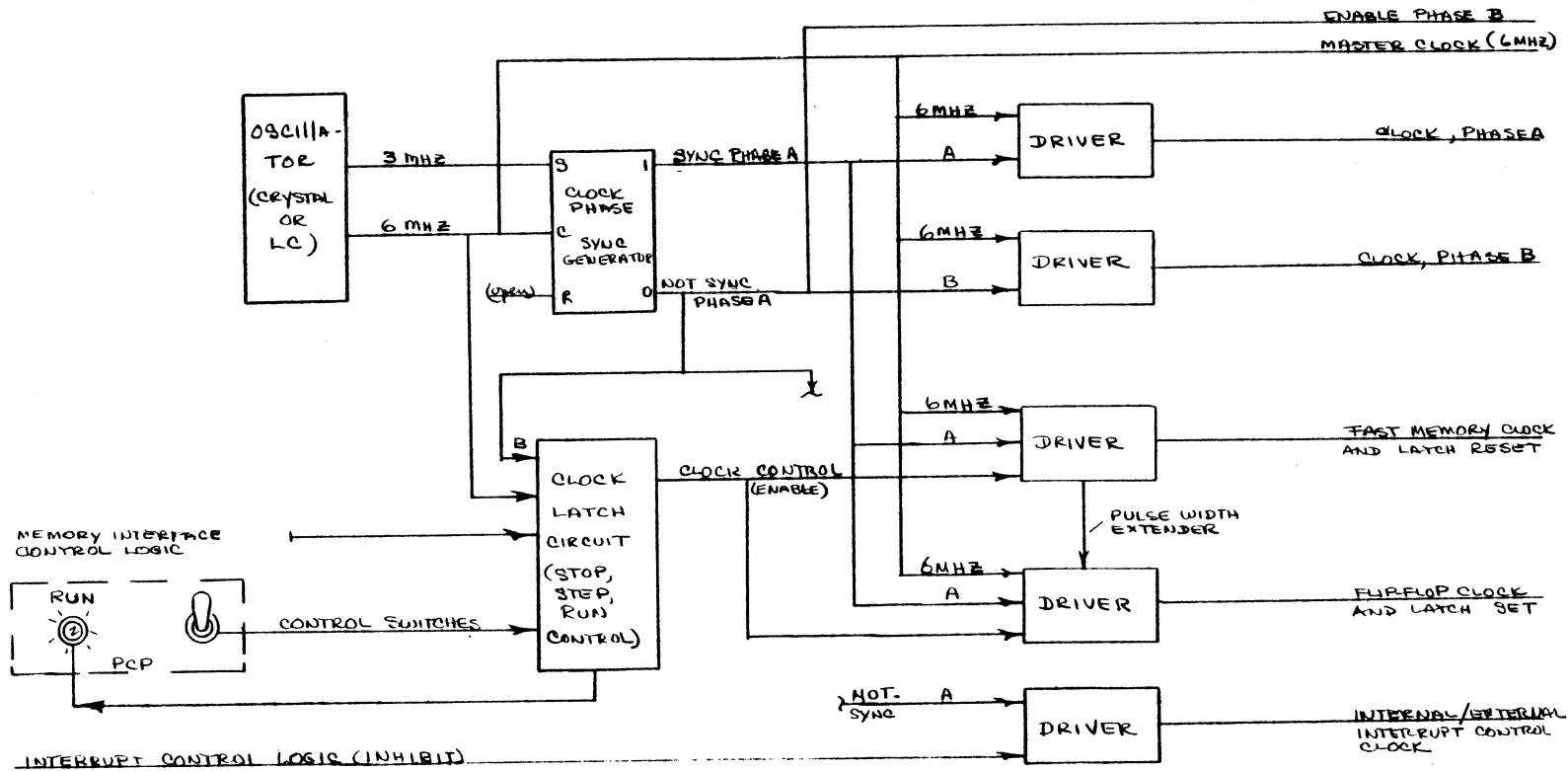


Figure 2-3. Clock Generation and Control Function Block Diagram

2-23 INSTRUCTION DECODE/EXECUTION CONTROL FUNCTION

This function is performed by logic circuits which carry out the fetch-interpret-execution control sequence of CPU operation. These circuits perform the major role in the control section of the Sigma 3 system.

2-24. This function (refer to figure 2-4) is performed by the O-register, and the decode and operation phase control groups of logic circuits. These circuits provide the timing (phase sequencing) for all data transfer and arithmetical computation performed by the CPU. The phase sequencing circuits are synchronized by the CPU clocks.

2-25. The last phase in the execution of an instruction initiates the fetching of the next instruction from memory, this is the beginning of the preparation phases. The fetch operation brings a 16-bit instruction word from memory for transfer to the adder, via the memory/arithmetic section interface register, the D-register. This operation is for effective address computation. At the same time, byte zero of the instruction is also placed in the O-register. This byte of information is the operation code (bits 0-3) and the address modification code RIXS (bits 4-7).

2-26. The O-register latches on to this information for the entire execution sequence. With the information held in the O-register, the decode group of logic interprets the operation and addressing modification to be performed, and develops controls which (during the preparation phases) command the operand address (effective address) computation sequence through the adder. The operand address may be the displacement field (byte 1) with zeros extended into byte 0, if the RIXS bits equal zero. If the RIXS bits do not equal zero, the displacement field becomes one of the operands in the address computation sequence. A maximum of seven phases of operation are required for preparation.

2-27. After the effective address has been determined, the last phase in the preparation sequence initiates the first of instruction execution. Like the preparation sequence, there are seven phases of execution. If a memory cycle is required during the execution phase and a memory fetch is called for, the contents of the effective address is transferred to the adder via the D-register. The transfer of byte zero to the O-register is inhibited by instruction control. The O-register has a single purpose and that is to latch the OP and RIXS codes during all sequences of prep and execute.

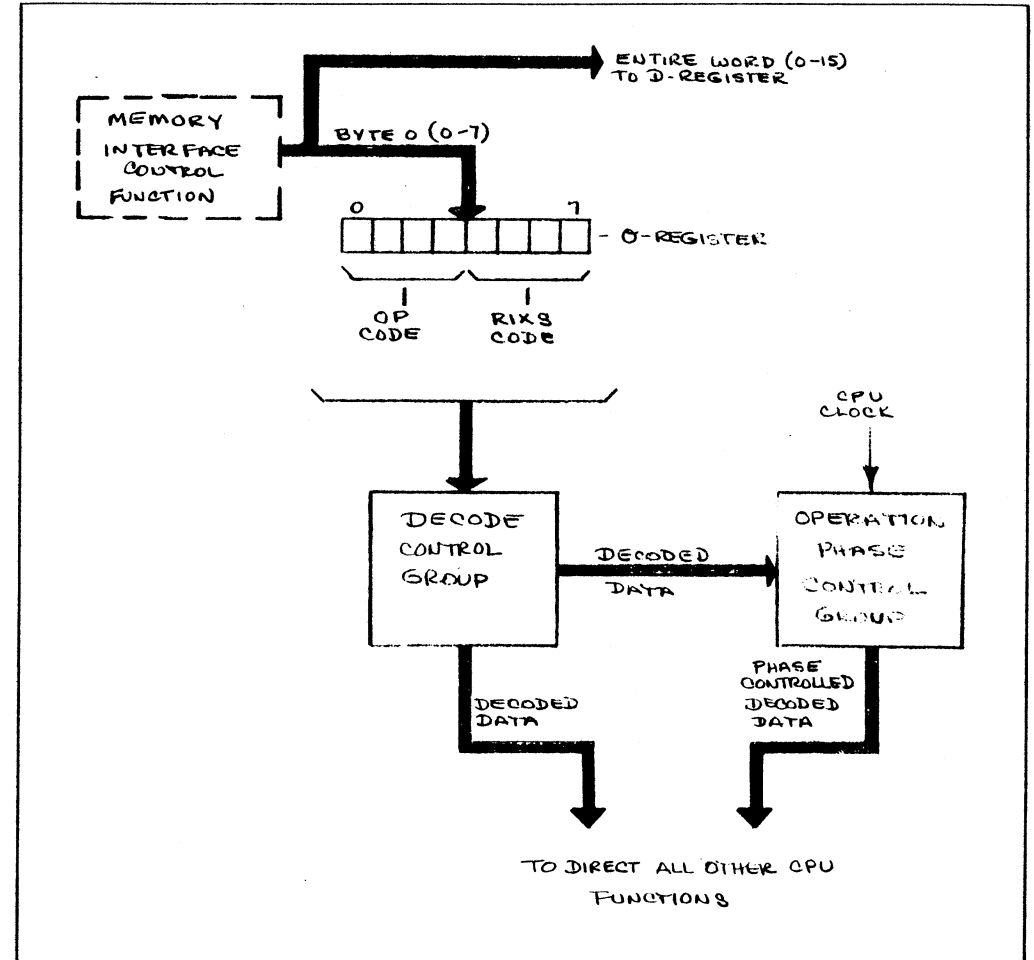


Figure 2-4. Instruction Decode/Execution Control Function, Block Diagram

2-28 MEMORY INTERFACE - (CPU INTERFACE FUNCTION)

The CPU interfaces with memory port No. 1 of Memory Port Multiple (MPM) No. 1 and No. 2 via two ribbon cables. (Refer to figure 205). These ribbon cables carry 16 bits of data, a parity bit, and interface control signals to and from memory. One cable also carries 16 address bits from the LA-register for addressing core during a memory read/write cycle.

2-29. One ribbon cable carries power monitor signals to the internal interrupt logic in the CPU to initiate a power on/off interrupt routine. These monitor signals are generated in the PT15B power supply, delivered to a module in the memory frame and transferred to the CPU via the memory-CPU ribbon cable.

2-30. Data is transferred at the memory interface by the memory port control logic. This logic (governed by I/O interface and instruction decode/execution control logic, in addition to switches on the PCP) generates the controls necessary to transfer data from memory to the D-register for processing by the CPU, and from the processor to core memory for storage. This logic also controls the transfer of the parity bit from the parity check and generation logic in the CPU. During the preparation phases of processing, the D-register control logic provides for simultaneous transfer of byte zero (bits 0-7 of a word from memory) to the O-register for operation decode and address computation control decode.

2-31. The control signals generated by the memory port control logic are also used by other control functions in the CPU. Two control signals which indicate the start of a memory cycle and the mode of operation (read or write) are supplied to indicator circuits controlling lamps on the PCP when these signals are selected for display. Also, the CPU clock is stopped if necessary then restarted by a memory port control signal during a memory cycle in order to synchronize the data transfer.

2-32. When the system is reset by a control button on the PCP, the signal generated is carried via one of the ribbon cables to the reset logic in memory to inhibit memory access and initialize all timing and logic in the MPM section of memory. This memory reset logic also provides the CPU with signals to reset internal logic. In addition, a signal generated by the power monitor logic also provides the same system reset.

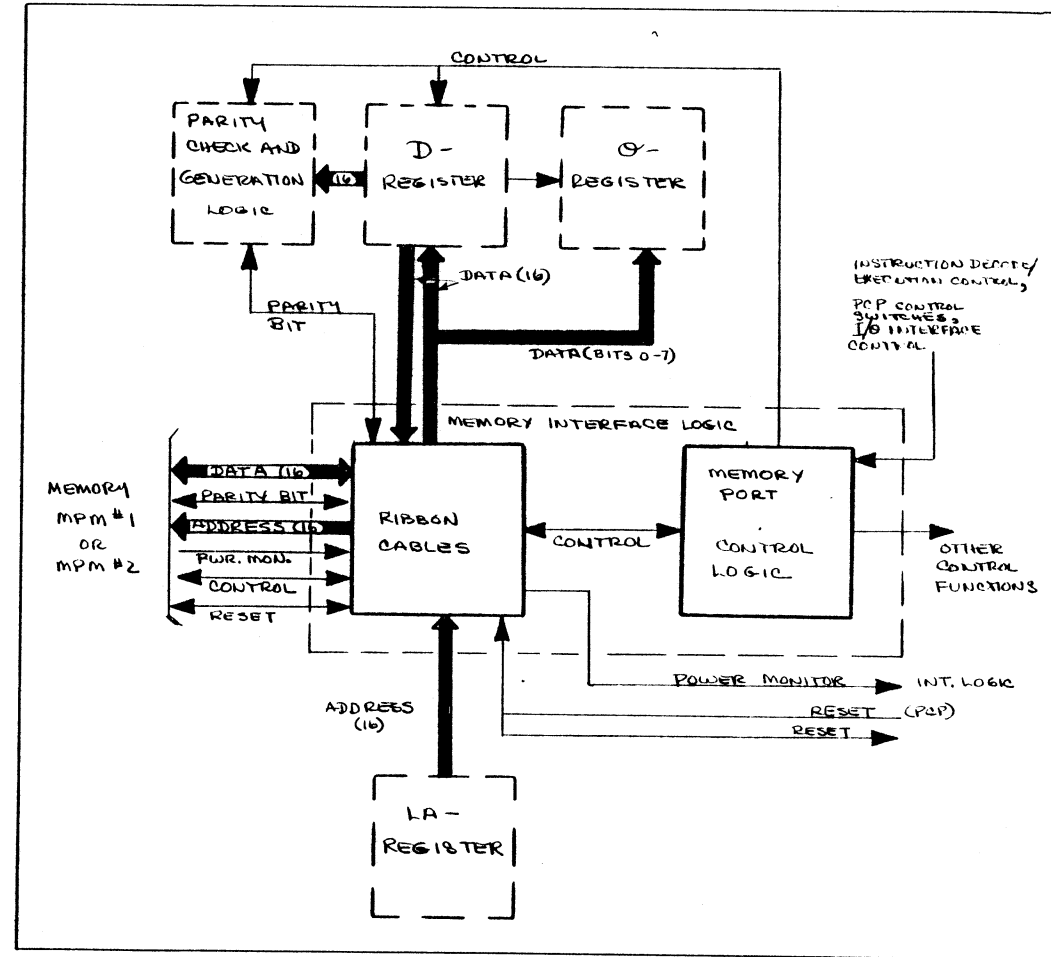


Figure 2-5. CPU Interface Function, Memory Interface Block Diagram

2-33 IIOP INTERFACE - (CPU INTERFACE FUNCTION)

The CPU interfaces with a device controller and its subcontroller through the Integral Input/Output Processor (IIOP) when this option is installed in the system. The IIOP provides the I/O interface logic. (Refer to figure 2-6).

2-34. All data to and from a device is transferred to the CPU via coaxial cables and driver/receiver modules. One cable transfers 8 bits of data, (bidirectionally) a parity bit and control signals. Another carries individual I/O function lines, control signals and a 1024Khz clock generated in the CPU for use by a device. A third cable transfers 8 function response (FR) lines which carry device status or address information to the FRL-register for further manipulation by logic in the CPU. The last cable carries the available-input-output priority signals to and from the subcontrollers. This cable is connected to the lowest priority device subcontroller (end of the priority chain) and the available-indicator from the device is returned through this cable.

2-35. All data is transferred under the control of the I/O phase and control logic which is part of the I/O interface logic. All data is transferred in single byte (8 bits) configuration from and to both single-byte and multiple-byte devices. The parity bit, transferred from the parity check and generation logic is also under control of this control logic.

2-36. The D-register (part of the CPU arithmetical group) is also the memory interface register for the I/O device. The D-register is used during the storage of data in memory from an input device or the reading out of data from memory to an output device during a service call. When reading from memory, data is transferred from the D-register to the IOL-register (I/O interface register) to await transfer to the controller. When storing device data in memory, the data transfer is to the D-register to await transfer to memory.

2-37. During the early phases of executing an I/O instruction (Read Direct), the IOL-register holds the device address from the D-register. This is output to the subcontroller on the I/O data lines. Also, during the same instruction, the LG-register of the CPU holds the effective address (I/O functions) for output on the I/O function lines to the subcontroller.

2-38. An I/O device is allowed a specific time period in which to respond to the I/O processor during the execution of an I/O instruction or a service call. A register in the CPU is incremented (each clock) and phase advance is inhibited until the device responds. If the device does not respond within 10.4 microseconds (32 clocks) an interface timer-runout indicator bit will effect a CPU WAIT condition. The setting of the TIMER switch on the PCP panel however, can override this condition until the expected response is present.

2-39. Control signals from the I/O phase and control logic are sent to other CPU functions to control the data processing sequence. In turn the I/O phase sequence is established by the state of the CPU adder control terms which are applied to the recognition logic within the I/O control logic. During an I/O service call the adder is decrementing, incrementing, or transferring the contents of a register and the state of the adder terms during these clocked operations are used to establish the sequence of I/O phases.

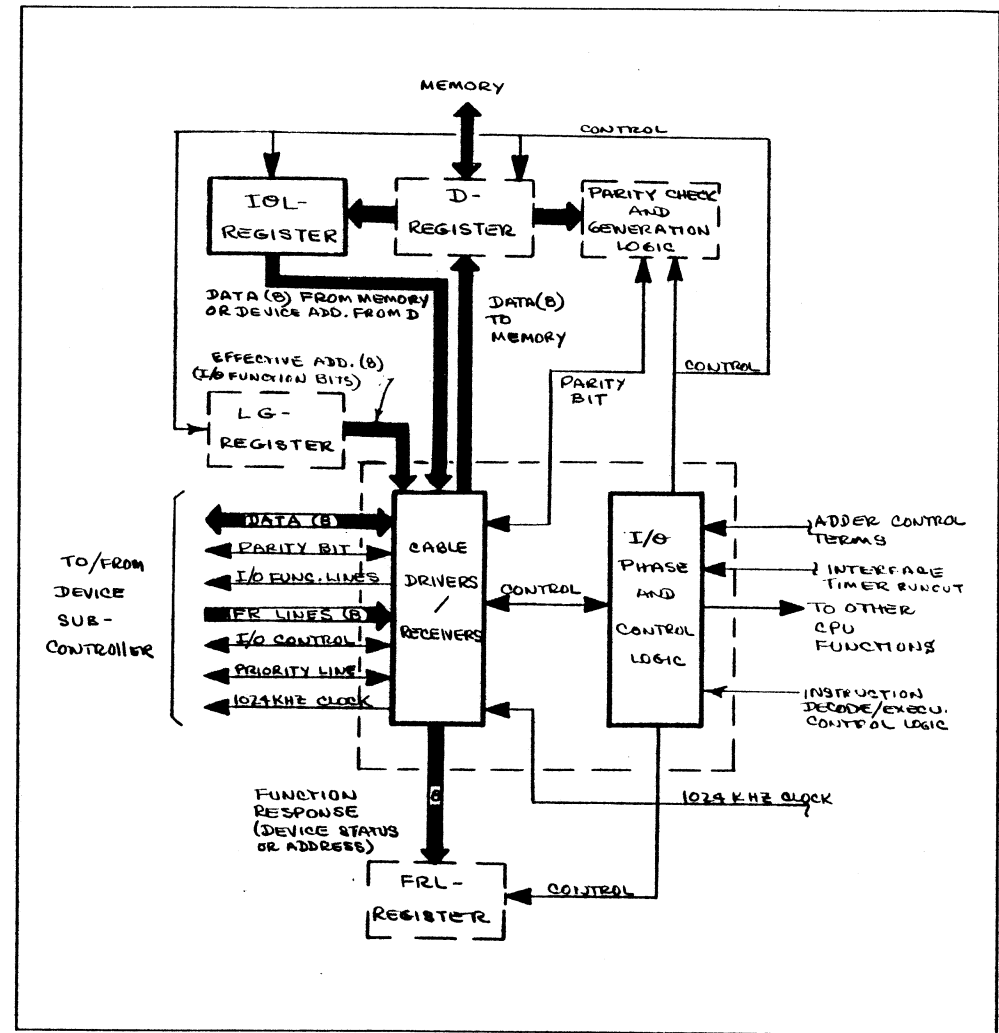


Figure 2-6. CPU Interface Function, IIOP Interface Block Diagram

2-40 DIO INTERFACE - (CPU INTERFACE FUNCTION)

If the DIO Interface option is installed in the system, the CPU interfaces directly with a special device and/or the External I/O Processor (EIOP). The DIO interface logic consists of coaxial cables with driver/receiver modules, a data register, an address register and the necessary control logic. (Refer to figure 2-7).

2-41. The transfer of data at the DIO interface is directed by the DIO control logic. Decoded direct control instructions and the CPU clock command the DIO control logic. Sixteen data and address lines, as well as interface control signals are distributed along the three coaxial cables. A 1024Khz clock generated in the CPU is also transmitted via one of these coaxial cables to the DIO interface for use by a device.

2-42. The DIO register will latch 16 bits of data from the J-bus (part of the arithmetical group of the CPU). However, during certain direct control instructions, bits 0 through 7 are zero and only bits 8 through 15 contain information. The data lines loaded from the DIO data register, function bidirectionally. Depending on the mode of a direct control instruction (read/write) the data lines will carry 16 bits of data in from a special device or the EIOP, or out to a special device or the EIOP. During the execution of a direct control I/O instruction, however, data lines 8 through 15 (byte 1) will contain data going out to a device, while lines 0 through 7 (byte 0) are left open to receive a response from the addressed device. The device response (status) is transferred to the D-register via the I-register to be processed by the CPU. Transfer of DIO data to the I- and D-registers is controlled by the D-register control logic.

2-43. The byte 1 data path to the D-register is for the transfer of data (device address) returned by the device when acknowledging an I/O interrupt. During all other direct control instructions the two bytes of input data (one byte through the I-register) are transferred simultaneously to the D-register for processing.

2-44. The address lines, loaded from the DIO address register will contain the effective address of the direct control instruction transferred from the H-register in the CPU. These lines carry the mode and function bits of the direct control instruction.

2-45. The EIOP is allowed a specific time period in which to respond to an I/O instruction. A register in the CPU is incremented (each clock) and phase advance is inhibited until the EIOP responds. If response is not within 42 microseconds (128 clocks) an interface timer-runout bit will effect a CPU WAIT condition. The setting of the TIMER switch on the PCP panel however, can override this condition until the expected response is present.

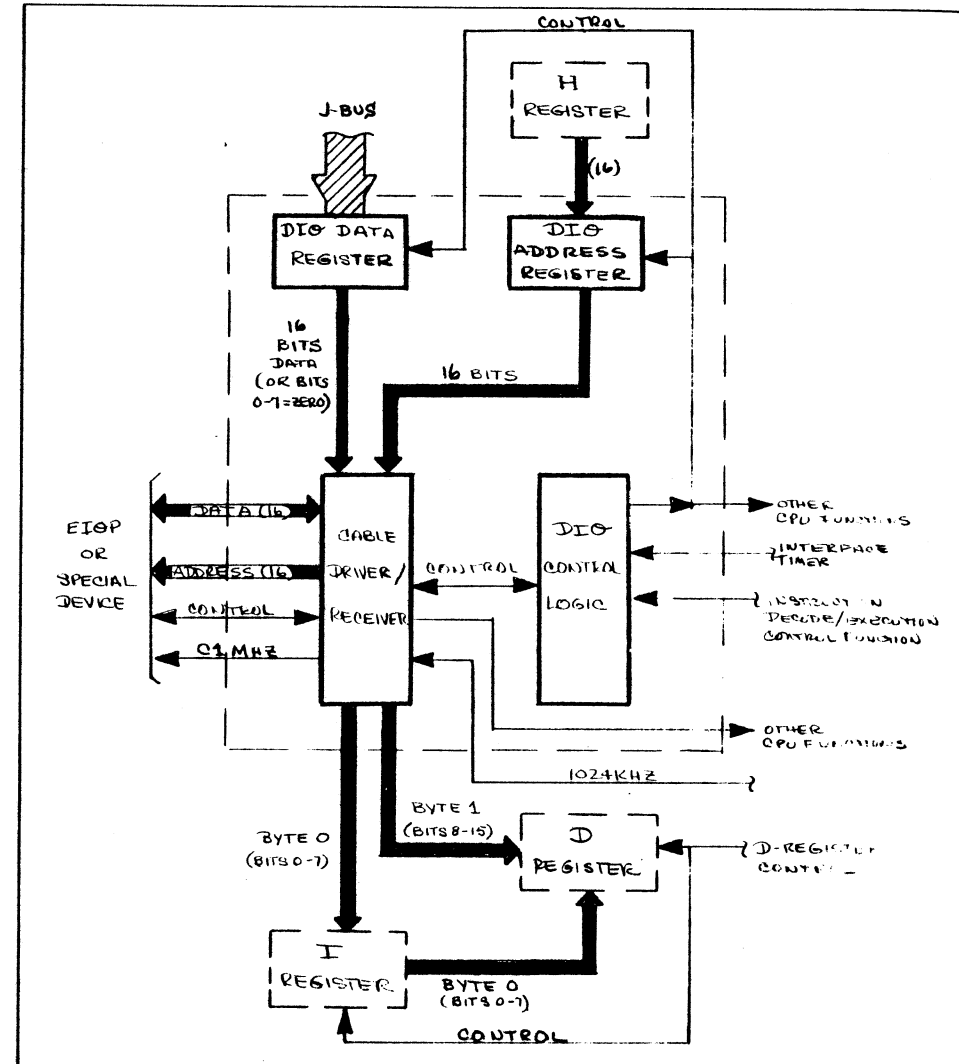


Figure 2-7. CPU Interface Function, DIO Interface Block Diagram

2-46 ADDER FUNCTION

The adder performs the arithmetical function in the J-bus and arithmetical group (refer to figure 2-8 and 2-9). The adder uses the input from the D- and H-registers and performs an add, subtract, and if the extended arithmetic option is installed, a divide or multiply operation. The adder also controls the transfer of the contents of the D-, H-, and S-registers, unaltered, to the J-bus. The adder will also increment the contents of the D, H, and S-registers during the transfer operation by adding one to the contents. These operations are under command of the adder control logic. This logic is directed by the instruction decode/execution control, I/O phase and control, memory interface control and internal interrupt control logic. Adder operation is inhibited when the CPU is in an idle state (i.e., COMPUTE switch is not in the RUN position).

2-47. The result of the adder operation is placed on the J-bus for transfer to a specific register depending on the phase of the instruction executed. Although the arithmetical function includes the add, subtract, divide and multiply operations, all data computations are performed by addition or a series of additions.

2-48. During data computation, the adder receives the two operands from the D-, and H-registers. The data in D has been fetched from memory or was on the J-bus as a partial computation. Depending on the phase or type of instruction the data (from memory) may be an instruction, the effective word which can be a value to increment, or the addend, subtrahend, divisor or multiplicand of an arithmetical operation. From the J-bus it can be the partial results of an effective address computation, or if an extended arithmetic operation is involved, it can be the remainder of a divide operation for transfer back to the J-bus without alteration.

2-49. The value in the H-register has been transferred from a general register in fast memory, this occurs for all instructions except an extended arithmetic operation. This value was loaded during a previous instruction. Depending on the operation, the contents of H is the augend or minuend for all instructions except for an extended arithmetic operation, during which time the contents of H is the difference (divide) or the partial product (multiply).

2-50. If the operation is a multiply or divide, the S-register is included in the arithmetical group of registers. The divide process is carried out by repeated subtractions (using 2^5 complement) and shifting the dividend to the left. This operation stays in the adder-J-bus-H-register loop. The difference on the J-bus and the dividend in the S-register are simultaneously shifted to the left, bit by bit. As a dividend bit (2^{15}) is shifted out of the S-register to the 2^0 position of H, a quotient bit is shifted into the 2^0 position in the S-register. The multiply process is carried out by repeated additions and shifting the partial product to the right. This operation, like the divide, stays in the adder-J-bus-H-register loop. The partial product on the J-bus and the multiplier in the S-register are simultaneously shifted to the right, bit by bit. As a multiplier bit is shifted out of the 2^0 position of the S-register, a product bit from the J-bus (2^0) is shifted into the 2^{15} position of the S-register. The quotient or the product in the S-register is finally placed on the J-bus (via the adder) after an incrementing alteration, if necessary.

2-51. In order to know if the result of an arithmetical operation is valid, the sign of the operands and the result are interrogated to determine if there is an overflow and/or carry. If the 16-bit magnitude in the D-register is equal to or less than the 16-bit magnitude of the H-register, there is a carry. An overflow occurs if there is an error in adding or subtracting algebraically, i.e.; an overflow occurs after an add, if the signs of the operands were positive (0) and the result is negative (1) or the signs of the operands were negative and the result positive. Also, an overflow occurs after a subtract, if the sign of the minuend in H was positive and the subtrahend in D was negative and the result is negative, or the sign of H was negative and the sign of D positive and the result is positive. The presence of a carry or an overflow results in setting associated flip-flops. The state of these flip-flops is indicated by lamps on the PCP. The state of these flip-flops is also used to determine the direction of the instruction execution or program execution.

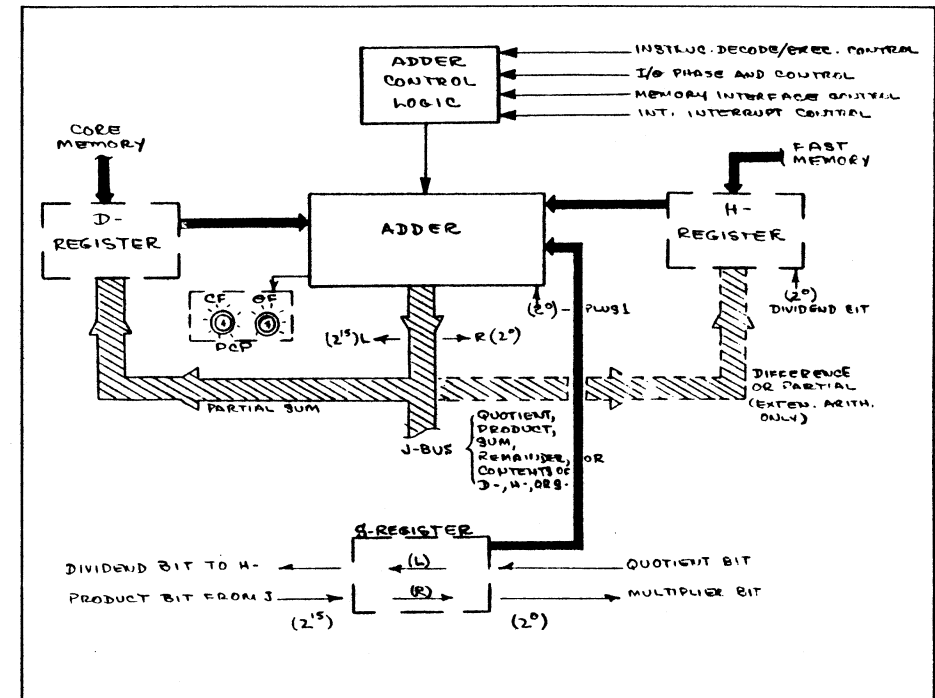


Figure 2-8. Adder Function, Data Flow Diagram

2-52 ADDER

The adder consists of 16 full adder circuits connected in parallel, therefore sums and carries are formed simultaneously. Refer to figure 2-8a through 2-8j. The carry bits are transferred immediately to the next higher order column. If new carries result from adding in the lower order carries, these are again propagated to the next higher order column. The entire sum is completed within one clock and the results are placed on the J-bus.

2-53. During add operations the adder control terms operate on each column of data in the adder and performs the equivalent to a logical AND function to generate and an OR to propagate a carry bit. If the operation is other than an addition, the control logic manipulates the adder input in order that the operation is finally handled as an addition.

2-54. An adder operation is performed in three steps. The generate and propagate steps and the voting step to establish the results. The generate step produces a bit for each column, if the bit is a one it immediately effects a carry into the next higher order-column. The propagate step produces a bit for each column, if the bit is a one it effects a carry into the next higher order column if there is a lower order-column carry. In this manner the carries ripple in the direction of the most significant bit position (2^{15}). The sum of the column is the result of the vote made in the third step. The sum is equal to either the column propagate-bit or the carry-into bit.

2-55. A detailed presentation of adder operation during an ADD, SUBTRACT or COMPARE and subtracting one from the contents of the H-register (H-minus-1) is covered in figures 2-8a, 2-8b, and 2-8c, respectively.

2-56. In figure 2-8a, the carry out (CF) of the MSB is normal as the carry indicates the sign of the result which is negative. The carry would extend ones to infinity if the signs of the operands were extended. The true sum is X'1E7AB' or X'F...FE7AB' with the sign extended. There is no overflow indication (OF) as the result is algebraically correct.

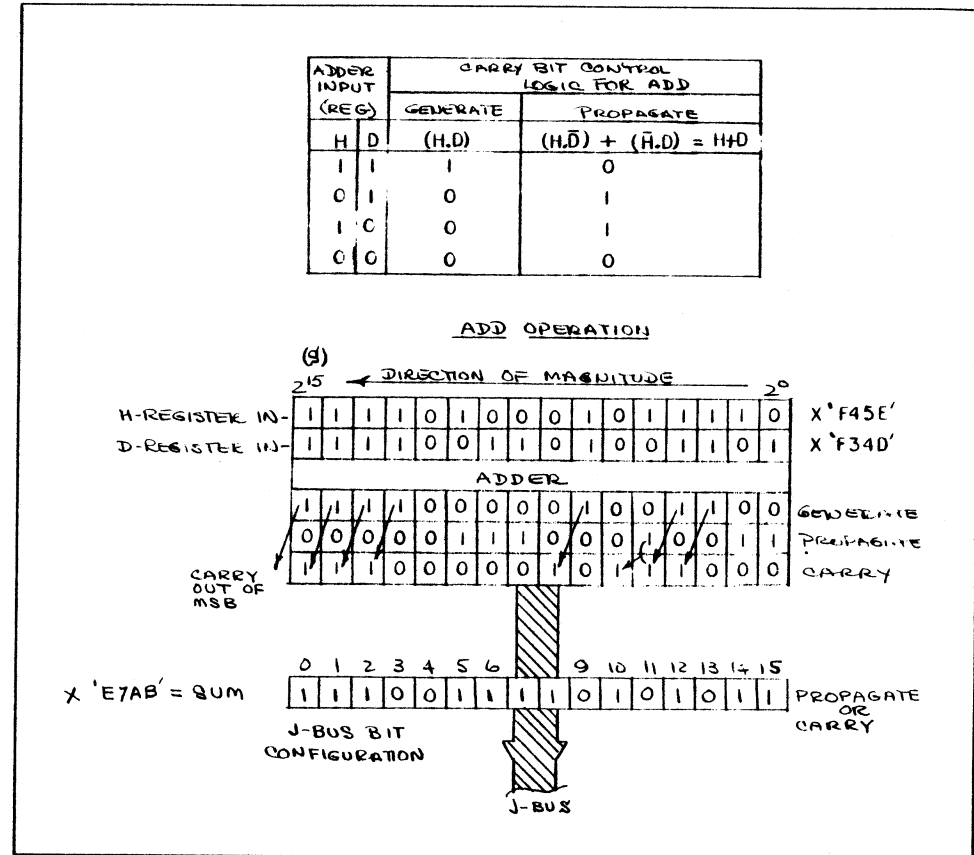


Figure 2-8a. Adder Operation - Addition

ADDER INPUT (REG)		CARRY BIT CONTROL LOGIC FOR SUB OR COMPARE	
		GENERATE	PROPAGATE
H	D	$(H.\bar{D})$	$(H.D) + (\bar{H}.\bar{D})$
1	1	0	1
0	1	0	0
1	0	1	0
0	0	0	1

SUBTRACT OR COMPARE OPERATION

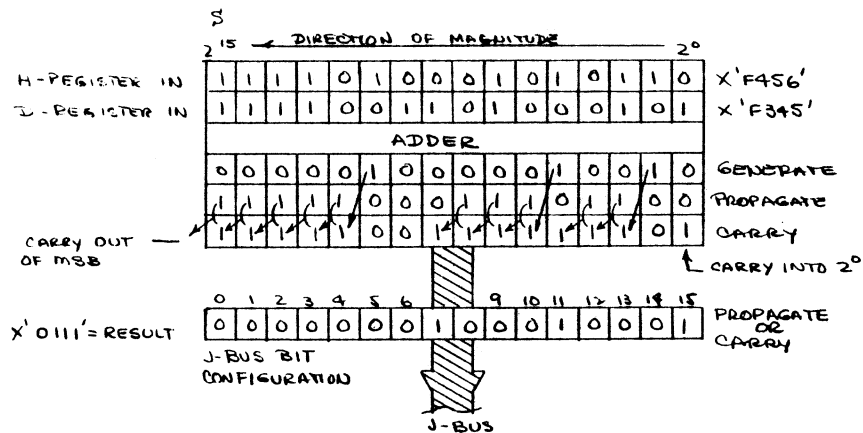


Figure 2-8b. Adder Operation - Subtract/Compare

2-57. The control terms for the subtract/compare operation perform the complement on the contents of the D-register as the terms complete an add operation. At the same time a carry bit is forced into the 2^0 position of the adder to establish the 2^2 complement of D. In this example, the carry out (CF) of the MSB of the adder is normal as the true sign of the result is positive. The carry would extend zeros to infinity if the signs of the operands were extended. There is no overflow indication (OF) as the result is algebraically correct.

ADDER INPUT (REG)		CARRY BIT CONTROL LOGIC FOR H-MINUS-1	
		GENERATE	PROPAGATE
H	D	$(H.D) + (H.\bar{D})$	$(\bar{H}.\bar{D}) + (\bar{H}.D)$
1	1	1	0
0	1	0	1
1	0	1	0
0	0	0	1

H - MINUS - 1 OPERATION

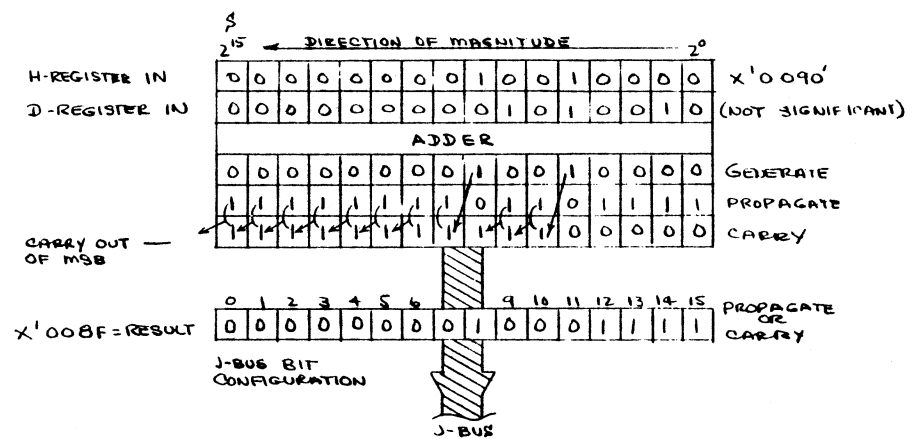


Figure 2-8c. Adder Operation - H - Minus - 1

2-58. In this operation, H-minus-1, the control logic alters the contents of the D-register to equal all ones (the 2^2 complement of one) and then performs an add operation. The carry out (CF) of the MSB is normal as the sign of the result is positive. The carry would extend to infinity if the signs of the operands were extended. There is no overflow (OF) as the result is algebraically correct.

2-59. Logical operations are performed by the adder when carries (adding operation) are involved. The results of the logical operations are equal to the state of the propagate bits. The adder control terms perform only the propagate step in logical operations.

2-60. In the logical AND, only the control terms to perform an AND function are used on each column to form the propagate bit. For an inclusive OR, the control terms for an AND and an OR are used, and for an exclusive OR, only the terms for OR function are used. Figures 2-8d, 2-8e, and 2-8f are detailed presentations of adder operation for logical AND, OR and EXCLUSIVE OR functions.

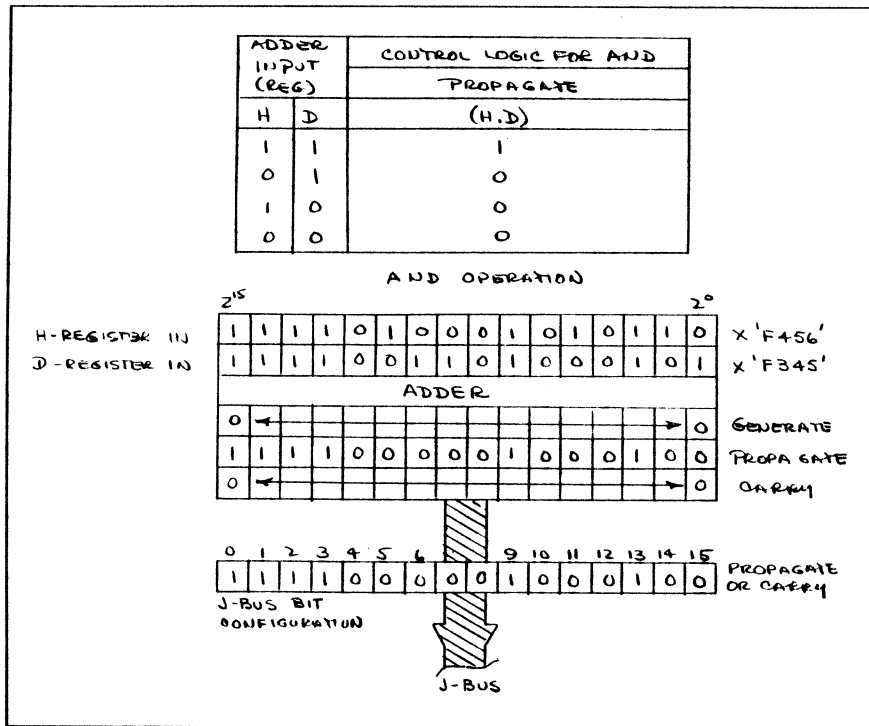


Figure 2-8d. Adder Operation - Logical And

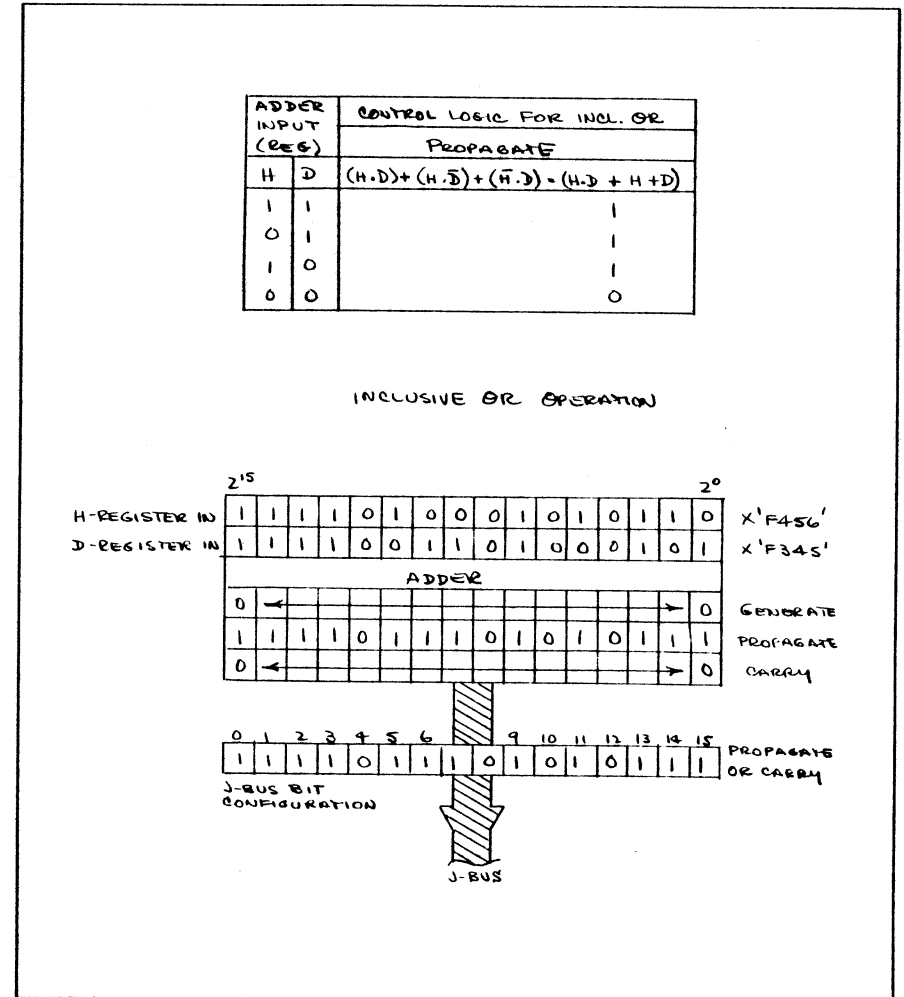


Figure 2-8e. Adder Operation - Logical Inclusive Or

ADDER INPUT (REG.)		CONTROL LOGIC FOR EXCLUS. OR
		PROPAGATE
H	D	$(H \cdot \bar{D}) + (\bar{H} \cdot D) = H \oplus D$
1	1	0
0	1	1
1	0	1
0	0	0

EXCLUSIVE OR OPERATION

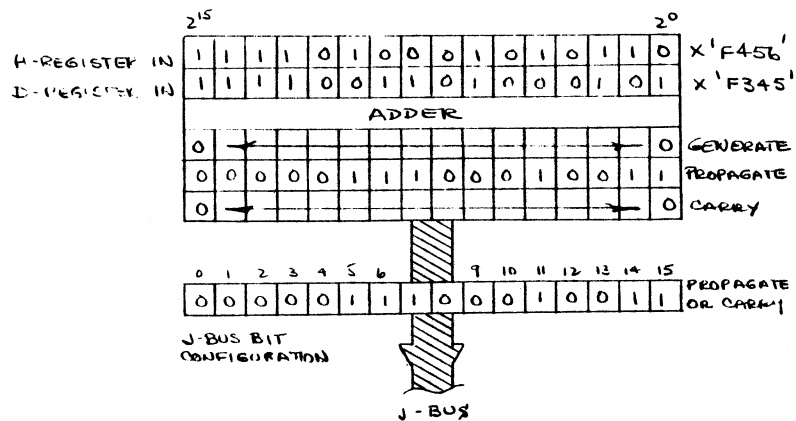


Figure 2-8f. Adder Operation - Exclusive Or

2-61. The D-register, H-register and S-register transfer directly to the J-bus through the adder. These transfers are also performed by the logical function controlling the state of the propagate bit of each column. The D-register contents are transferred when the adder control logic alters the data input from the H-register to equal all zeros and then performs a logical OR on each column. The result is the transfer of the contents of the D-register to the J-bus.

2-62. The H-register is transferred when the adder control logic alters the data input from the D-register to all zeros and performs a logical OR on each column. The result is the contents of the H-register on the J-bus.

2-63. The contents of the S-register are a direct input to the propagate bit location in the adder and when the adder control logic directs this transfer, the contents of S are on the J-bus.

2-64. During the transfer operation for the contents of D-, H-, or S-registers, if it is required to increment the contents, a carry bit is forced into the LSB location in the adder and a one is added before the contents appear on the J-bus. Figures 2-8g and 2-8h are detailed presentations of a D-to-J and an H-to-J transfer.

ADDER INPUT		CONTROL LOGIC FOR D-TO-J
		PROPAGATE
H	D	$(H \cdot \bar{D}) + (H \cdot D)$
1	1	1
0	1	1
1	0	0
0	0	0

TRANSFER INPUT FROM D TO J-BUS

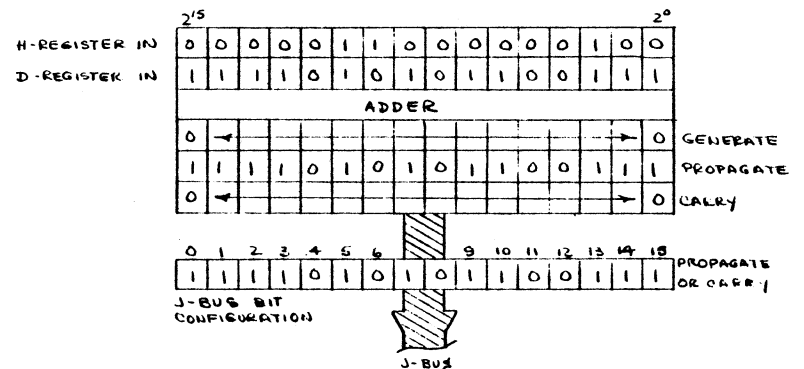


Figure 2-8g. Adder Operation - D-Register Transfer to J-Bus

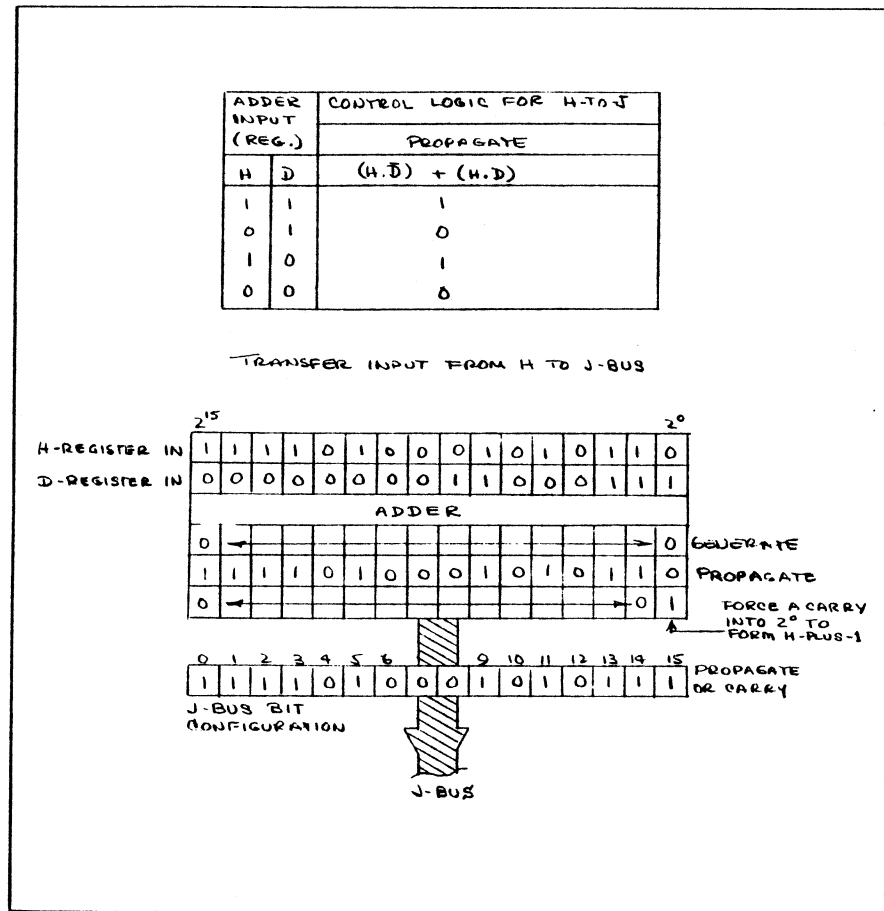


Figure 2-8h. Adder Operation - H-Register Transfer to J-Bus

2-65. During the execution of a specific instruction, it is necessary to transfer the complement of the contents of the H-register to the J-bus. To do this the adder control logic alters the input from the D-register to equal all ones and performs the equivalent of an OR function on each column. Figure 2-8j represents this operation.

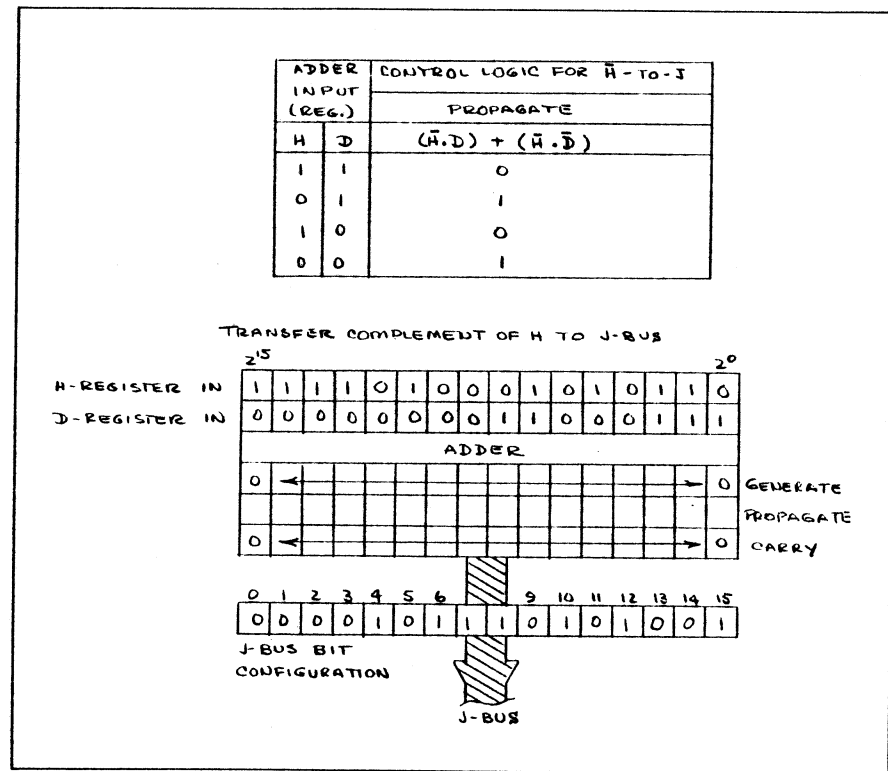


Figure 2-8j. Adder Operation - Complement of H to J-Bus

2-66 J-BUS FUNCTION

The J-Bus is the main information artery in the CPU. (Refer to figure 2-9). Information is placed on the J-bus from the adder, the registers in fast memory and, when the CPU is idle, from switches on the processor control panel (PCP). The J-bus in turn feeds information to those registers which form the arithmetical group, viz.; the D-register, S-register, LA-register, H-register, and registers in fast memory. The J-bus also feeds information to the status bit lamp drivers to light lamps on the PCP.

2-67. J-Bus and Arithmetical Register Group. Figure 2-5 is a functional flow diagram depicting the data flow in the J-bus and arithmetical register group. Input to the J-bus from the adder is the result of a partial or full arithmetical operation. In addition, the input to the J-bus can be the contents of the D-, H-, or S-registers. These registers transfer to the J-bus through the adder. This is accomplished by adder control terms which issue the contents of the D-, H-, or S-registers, without change, to the J-bus. Input to the J-bus from a register in fast memory, or the data switches on the PCP, is commanded by the J-bus control logic. This logic is directed by instruction decode/execution control, I/O phase and control and PCP control switches.

2-68. During the PREP phases (effective address computation) of an instruction, the D-register interfaces with core memory, and data on the J-bus operates in a loop from the D-register (through the adder) to the LA-register (to address memory) and the H-register, or back to the D-register. When the path is back to the D-register, the H-register picks up the contents of a general register (depending on the required computation) and feeds the contents to the adder. The contents of the general register is then added to the contents of the D-register. This operation is repeated until the sum is equal to the effective address of the instruction, at which time (end of preparation) the J-bus feeds the effective address to the LA-register and H-register.

2-69. The J-bus and arithmetical group performs in the same manner during the instruction execution phases as in the PREP phases. During arithmetical computation one operand is previously stored in fast memory and the other (from memory) is in the D-register. However, during the final phase the result is loaded in fast memory or in the D-register for storage in core memory. Refer to the adder function and figure 2-8.

2-70. The path to and from the adder is also used when incrementing the contents of the D-, and S-registers and when incrementing or decrementing the contents of the H-register. When incrementing the contents an adder preset term is generated to control the adding of a one to bit position 15 to perform the required D-plus-1, S-plus-1, or H-plus-1 operation. When decrementing the contents of the H-register (during an I/O service call) an adder preset term is generated to control the required H-minus-1 operation.

2-71. When performing data manipulation between any two registers in fast memory, the path is from a register in fast memory onto the J-bus to the D-register, through the adder and back to a register in fast memory.

2-72. The J-bus is not used between fast memory and the S-register. The major portion of data flow between a register in fast memory and the S-register is during an extended arithmetic operation.

2-73. During the execution of an instruction requiring a register shift operation, information on the J-bus is shifted right or left one bit before it is placed in the H-register; the overflow bit (zero) is lost during a left shift operation, however, the overflow bit (15) is shifted to the zero position in the S-register during a right shift.

2-74. When the CPU is service connected through the IIOP, the D-register serves as the data interface between a device controller and input to core memory. During this mode of operation, the J-bus and arithmetical register group is used only during decrementing the byte count in the H-register and incrementing the word address in the S-register which was transferred from a register in fast memory. When the CPU is connected through the DIO interface the D-register serves as the data interface between the EIOP or special device and input to a general register in the CPU. The data-in flow is from the D-register, through the adder to the J-bus and on to a register in fast memory. The data-out flow is from a register in fast memory on to the J-bus to the D-register through the adder to the J-bus and out to the DIO data register.

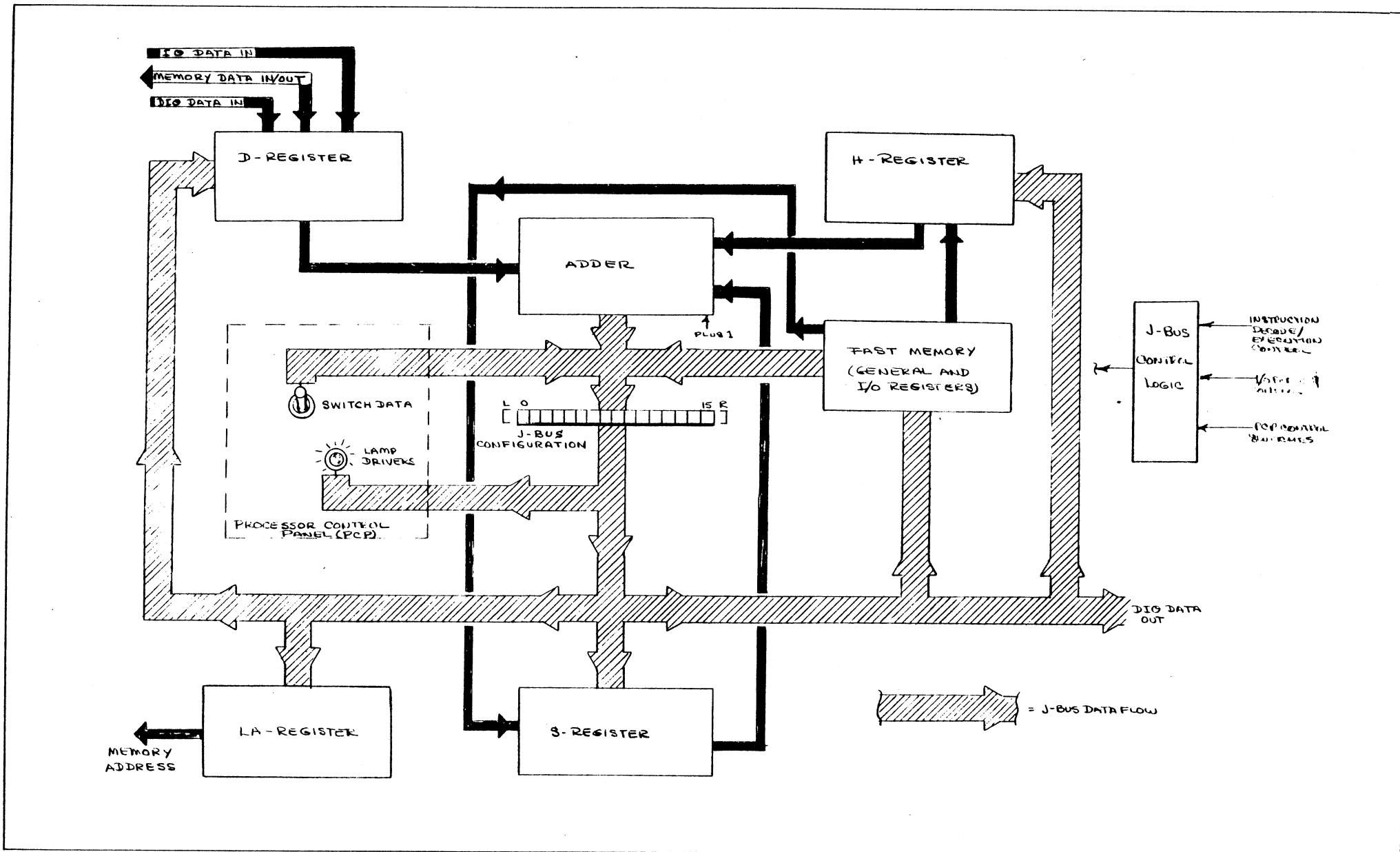


Figure 2-9. J-Bus and Arithmetical Register Group, Data Flow Diagram

2-75 D-REGISTER - (REGISTER FUNCTION)

The D-register is the most utilized register in the CPU. (Refer to figure 2-10). A large portion of D-register operation is performed when the register functions in the arithmetical register group as direct input to the adder. This register also serves as a core memory interface register to hold one word (16 bits) to be stored in memory or one word fetched from memory. When the Integral Input/Output Processor (IIOP) is installed, the D-register provides a data path between memory and a peripheral device. When the Direct Input/Output (DIO) interface option is installed, the D-register provides a communication path between the External Input/Output Processor (EIOP) and/or a special device and a CPU general register. The transfer of data to the D-register is governed by the D-register control logic which is commanded by the I/O and memory interface control logic, the instruction decode/execution control logic and PCP control switches.

2-76. When the register is performing as part of the arithmetical register group, the transfer of data is from memory (an instruction or operand) to the adder for computation or transfer, unaltered to the J-bus. When transferring DIO data into a general register, the path is through the adder (unaltered) to the J-bus.

2-77. When operating relative to the I/O interface (IIOP), single bytes of data (8 bits) from a peripheral device are input to the D-register and after a full word (2 bytes) are transferred, the word is transferred to memory. When connected to an output device a full word is transferred to the D-register from memory and then transferred to the peripheral via the IOL-register (part of the I/O interface logic) one byte at a time. Refer to I/O interface logic, and figure 2-6.

2-78. The peripheral at the I/O interface responds to an I/O instruction on the function response lines (8) to the D-register via the FRL-register. On other occasions the FRL lines to the D-register represent the least significant byte (byte 1) of internal interrupt status. The FRL-register also transfers to the D-register the sum of the shifts taken during a normalizing operation.

2-79. When the CPU is in communication with the EIOP or a special device via the DIO interface, the least significant byte (byte 1) of the word is issued directly to the D-register, the most significant byte (byte 0) is issued via the I-register. This allows for the simultaneous use of the bidirectional DIO data lines (one byte out and the other byte in, refer to DIO interface logic, and figure 2-4). When not used for single-byte bidirectional transmission, the DIO data lines carry one word of data to the D-register via the combination of the D- and I-register paths. On all other occasions the contents of the I-register represent the most significant byte (0) of internal interrupt status. On these occasions the FRL-register simultaneously contains the least significant byte (1) of internal interrupt status.

2-80. During instruction preparation phases of an instruction that does not require address computation, as a register-copy or set-multiple-precision mode direct control instruction, bits 8 through 15 in the D-register are transferred to the LG-register. These bits are interrogated to determine types of operations to be performed to execute the instructions. During the execution of a direct-read I/O instruction, the LG-register is the buffer for the I/O function lines to the device subcontroller.

During the execution of a direct write (interrupt mode) instruction the D-register contents points to the interrupt levels to be operated upon. At this time the contents of D are transferred to the Internal interrupt level logic.

2-81. When the CPU is in idle (memory cycle inhibited) and an instruction has been entered in D from the switches on the PCP panel (via the J-bus) byte zero of the instruction (operation code and RIXS bits) is transferred to the O-register for decode. The transfer takes place when the CPU is set from the idle to the COMPUTE state. Also when the CPU is in the idle state, the contents of the D-register, when selected by PCP switches, can be brought to the data display lamps on the PCP.

2-82. At one time, only during the preparation of a register copy instruction, the three least significant bits in the D-register (effective address) are issued to the general register address-register (RA). These bits are the address of the source register of the copy instruction.

2-83. When the CPU enters an interrupt subroutine the program status indicator bits (Program Protect, PP, External Interrupt, EI, overflow, OF and carry, CF, indicators) as part of a word, are input to the D-register for storage in memory. After the execution of the subroutine the status-bit word is fetched from memory to the D-registers and restored, via the J-bus, to the program status indicators. The status bits are also transferred to the D-register during a read direct control instruction in order to load them (via the adder and J-bus) into the accumulator (fast memory, general register number 7).

2-84. During a memory cycle (read/write) the data in D is also present in the parity check/generation section. Refer to paragraph 2-85 and figure 2-11 for functional description of the parity check/generation logic operation.

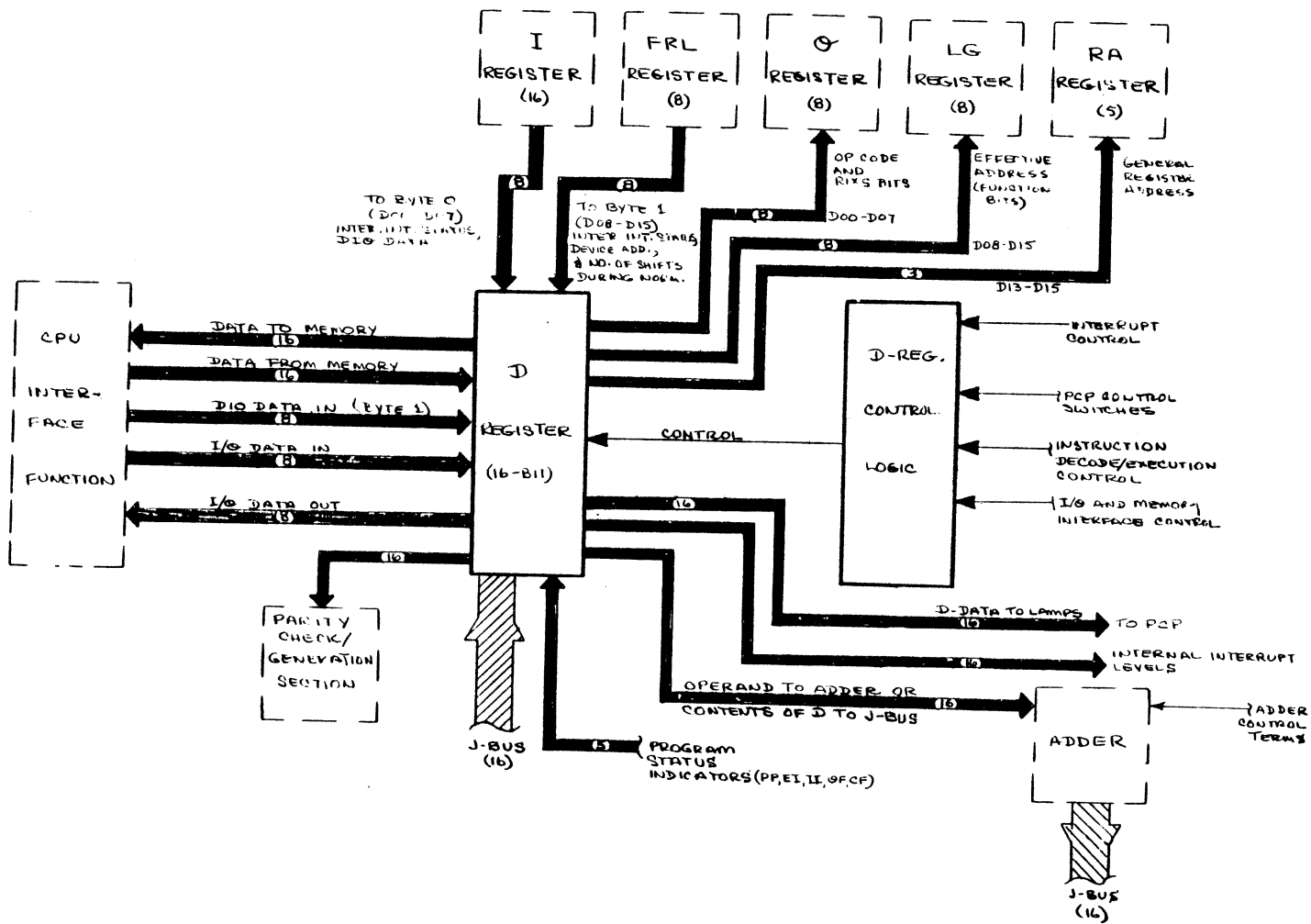


Figure 2-10. D-Register Data Flow Diagram

2-85. Parity Check/Generation Section. (Refer to figure 2-11). To avoid errors due to pickup of illegitimate extra ones (pulses) or dropping a one during the transfer of a word or byte of a word, an odd parity check/generation system is used to check the validity of the information transferred. This system screens a word or byte (counts the number of ones) and determines the value of an extra bit (parity bit) to be added to the word or byte which will cause the total number of ones to remain or become an odd number.

2-86. Data in the D-register is checked for odd parity each memory cycle. The parity bit (extra bit) will accompany the data from memory or an I/O device. The parity check circuits count the number of ones in the word or byte to determine if the value of the parity bit sent with the data is valid, if not, a parity error is indicated. Data to be transferred to memory or an I/O device is also screened and a parity bit is generated by the parity generation circuits to establish an odd number, indicating valid data.

2-87. When the parity check circuit has determined that the state of the parity bit sent with the data is not valid, a memory parity error (MPE) indicator is generated. This indicator is used to establish certain control for CPU operation:

1. The MPE indicator drives the parity error lamp on the PCP panel and sets the condition to drive the WAIT circuit and lamp when the PARITY ERROR switch (on the PCP) is set to INTERPT (interrupt) and an I/O service call is not in execution. Also when the PARITY ERROR switch is set to either INTERPT or HALT the MPE indicator is used to inhibit the CPU clock.
2. The MPE indicator points to a machine fault and this allows the CPU to enter an interrupt routine if the machine-fault interrupt option is installed.
3. During an I/O service call the MPE indicator is transmitted to the error flag position in the IO control doubleword for future information.
4. During an I/O order-out service call the MPE indicator is applied to a bit position in the D-register which designates an IOP HALT terminal order which is sent to the device controller at the termination of the service call.

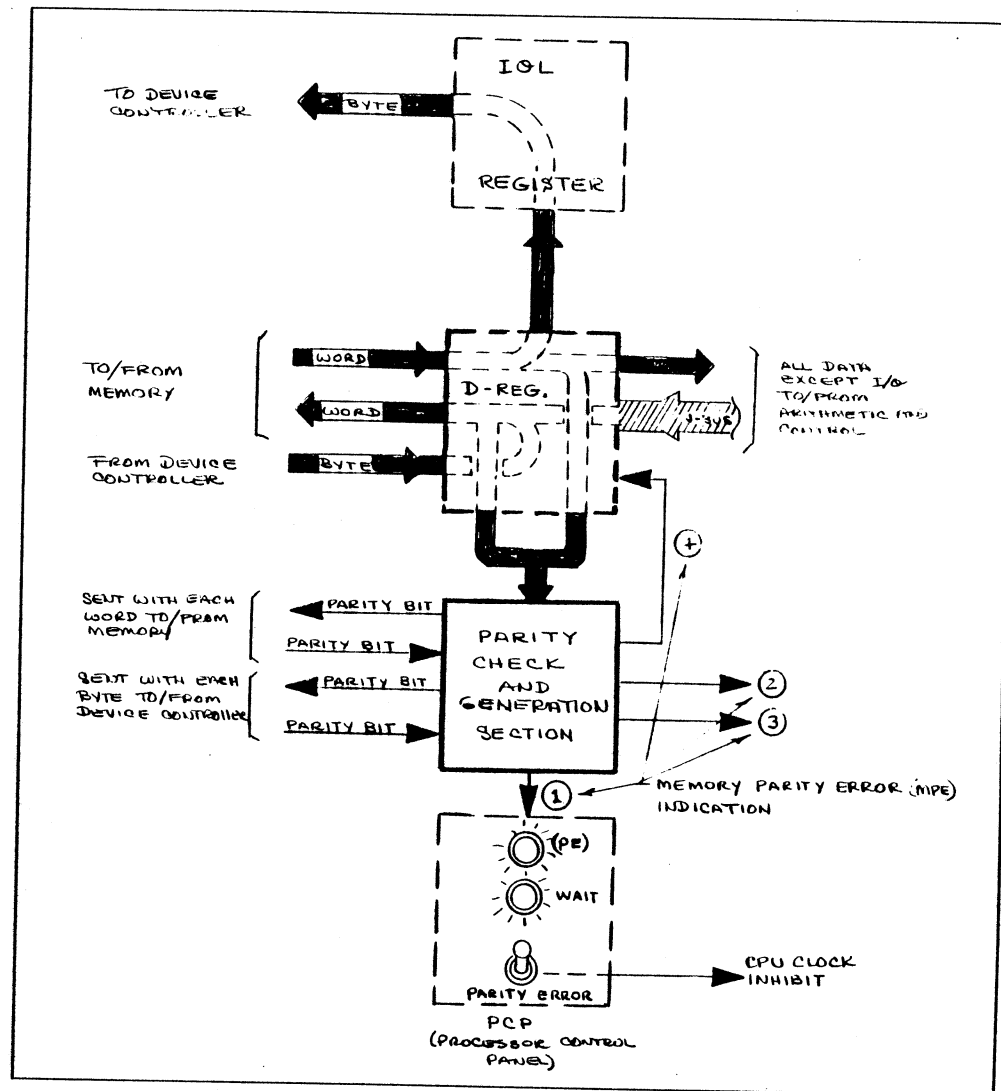


Figure 2-11. Parity Check and Generation, Data Flow Diagram

2-88. H-REGISTER - (REGISTER FUNCTION)

A large portion of H-register operation is performed when the register functions in the arithmetical register group as direct input to the adder. The register operation is also prominent during the execution of an instruction requiring a shift operation.

2-89. During an I/O service call, this register holds and manipulates (through the adder) the decrementing of the I/O table byte count. In addition, the H-register is used as a counter to establish elapsed time at the I/O and DIO interfaces during the execution of a read/write direct I/O or DIO instruction.

2-90. Data transfer to the H-register is under the command of the H-register control logic. (Refer to figure 2-12). This logic is directed by terms from instruction decode/execution control logic, I/O phase and control logic, and PCP control switches.

2-91. Data flow to the H-register is from the J-bus, or from a general or I/O register in fast memory. Data flow from the H-register, other than to the adder, occurs during the execution of a read/write direct control instruction at the DIO interface. At this time the H-register holds the mode and function bits (effective address) for transfer to the DIO address register.

2-92. When the H-register is performing as part of the arithmetical register group, the data in H is input to the adder for computation and out on the J-bus for transfer. Data flow from the H-register, not for computation, is through the adder, unaltered, and out on the J-bus for transfer.

2-93. When the H-register is operating during a shift operation, input from the J-bus is shifted right or left before entering the H-register. If the shift is to the left, bit zero of the S-register is transferred to bit 15 of the H. If the shift is right cyclic, bit 15 of the S-register is transferred to bit zero of H.

2-94. The H-register is used as a counter (interface timer) while waiting for a device response at the I/O or DIO interfaces. When bit H10 is high (decimal value of 32 clocks or 10.4 msec.) time has runout at the I/O interface for device response. When bit H08 is high (decimal value of 128 clocks or 42 msec.) time has runout at the DIO interface.

2-95. When the CPU is in the idle state, the H-register can be loaded, via the J-bus, with data in the configuration set by 16 data-bit switches on the PCP. Also, when the CPU is idle, the contents of the H-register (if selected) is displayed by the PCP lamps.

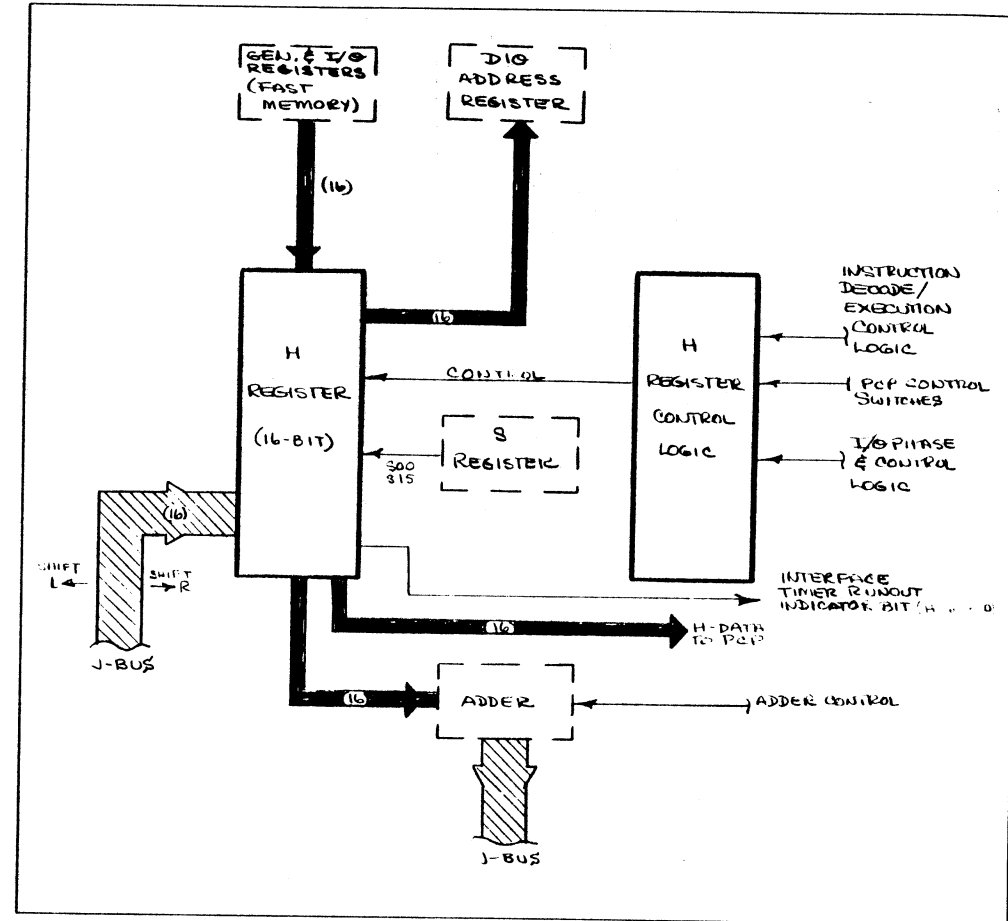


Figure 2-12. H-Register Data Flow Diagram

2-96. S-REGISTER - (REGISTER FUNCTION)

The major portion of S-register operations is performed when the register functions in the arithmetical group as input to the adder to establish (by incrementing the contents of S) the next instruction address for transfer by the J-bus.

2-97. Another important function of the S-register is to perform a right or left shift of its contents during the execution of an instruction requiring a shift operation.

2-98. Refer to figure 2-13. Data transfer to the S-register and within the register is under the command of the S-register control logic. This control logic is directed by terms from the instruction decode/execution control logic, I/O phase and control logic and PCP control switches.

2-99. Data flow to the S-register is from the J-bus or general or I/O registers in fast memory. Data out from the S-register is to the adder for increment or direct transfer to the J-bus. The S-register contents, at the end of instruction execution, is latched in the memory address latches (LA-register) as the address of the next instruction to be accessed from memory.

2-100. During a divide or normalize operation (left shift) the S-register holds the least significant portion of the numerator (or value to be normalized) and the register shifts this word left, bit-by-bit. The S-register also holds the multiplier during a multiply operation and shifts this value right, bit-by-bit. The J-bus and S-register shift simultaneously, left or right during the shift or normalize, divide and multiply operations.

2-101. During a left shift, bit zero of the S-register transfers to bit 15 of the H-register and during a right shift, bit 15 of the J-bus is transferred to bit zero of the S-register. If the right shift is a cyclic shift, S15 transfers to H00.

2-102. During the idle state of the CPU, the S-register can be loaded, via the J-bus, with data in the configuration set by 16 data-bit switches on the PCP. Also, during the idle state, the contents of the S-register (if selected) is displayed by the PCP lamps.

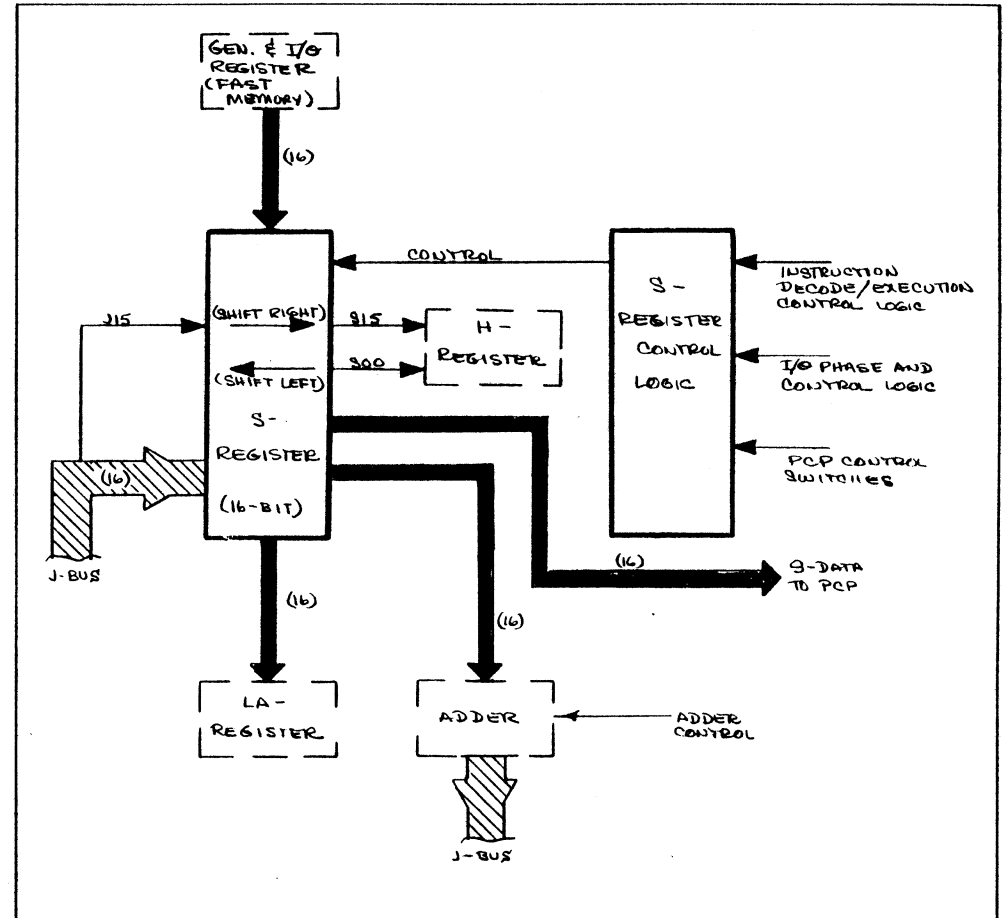


Figure 2-13. S-Register Data Flow Diagram

2-103. LA-REGISTER - (REGISTER FUNCTION)

The major function of the LA-register is to latch 16 bits of memory address information from the J-bus or the S-register at the beginning of a memory cycle. The LA-register also latches the 9 bits (7 through 15) from the Interrupt Address register (INTAD); this is the address dedicated to a particular interrupt service routine stored in memory. Refer to figure 2-14.

2-104. Data transfer to the LA-register is commanded by the LA-register control logic which is directed by instruction decode/execution control, I/O phase and control, PCP control switches and internal/external interrupt logic.

2-105. Data flow to the LA-register is from the J-bus, S-register or the INTAD-register. Data in the LA-register is always present on the memory address lines but becomes effective as an address only when a memory cycle is enabled by the memory interface control logic. At other times, when a memory cycle is inhibited during the execution of a shift, register copy or direct control instruction, the function bits (effective address) in the LA-register are transferred to the LG-register. The function bits in the LG-register are interrogated to control the execution of the instruction. The contents of the LA-register can change during execution, therefore, the LG-register is used to hold the function bits during this time.

2-106. If the Memory Protect option is installed, the five most significant bits (0 through 4) of the address in the LA-register are transferred to the Memory Protect Address register (MPAD). The first four bits of this register point to the 16 bit register which has been assigned to control the protection of a specific 4K of core memory; the fifth bit points to the first or second half of the 4K to be considered. The next three bits (5 through 7) of the LA-register are interrogated by the memory protect logic to determine which page or pages (256 locations each) may be protected. If the address information contained in LA-register bits 0 through 7 infringes upon a protected area, a memory protect violation occurs and is handled by an interrupt routine.

2-107. If a program interrupt occurs the INTAD-register holds the interrupt level dedicated address. This information (9 bits) is latched into the LA-register, an interrupt-entry sequence starts, and a memory cycle is entered.

2-108. When the CPU is in the idle state, a memory address set by the 16 data-bit switches on the PCP is loaded into the S-register, via the J-bus. This address is transferred to the latches (LA-register) when the PCP switch controlling the memory mode operation is set to deposit or fetch data in memory. Also, in the idle state, the contents of LA (if selected) is displayed by lamps on the PCP.

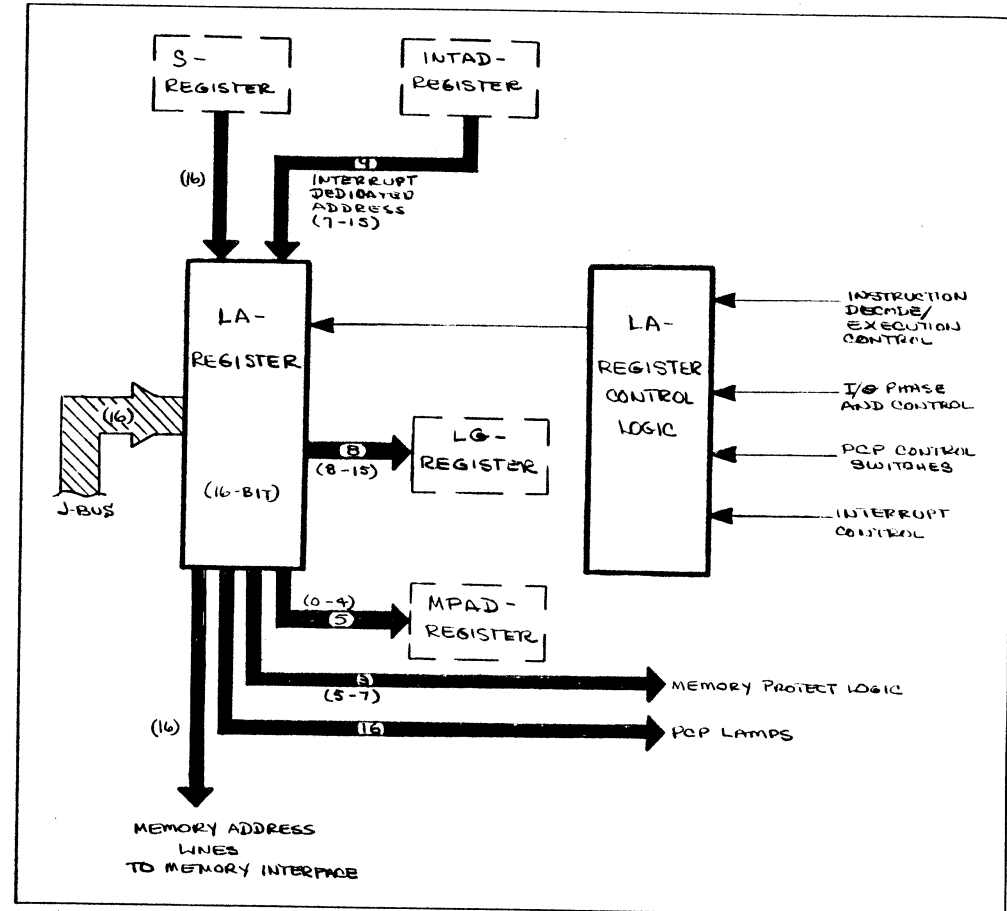


Figure 2-14. LA-Register Data Flow Diagram

2-109. LG- AND LM-REGISTERS - (REGISTER FUNCTION)

The LG- and LM-registers (refer to figure 2-15) are considered together in this description inasmuch as the only data-input to the LM-register is from the LG-register.

2-110. LG-Register. The LG-register is an 8-bit register with the primary function of holding the effective address bits (08 through 15) of an instruction during the preparation and/or execution of instructions for which a memory cycle is not required (viz., shift, register copy and internal read/write direct control). Data transfer to the LG-register is commanded by the phase of the operation. Data from the LG-register is commanded by the logic controlling the destination of the data, however, individual LG bits are used to control other functions within the CPU during instruction execution.

2-111. The effective address bits are transferred to the LG-register from the D-register during the early prep phases and from the LA-register at the end of the prep phase prior to execution. The transfer of the effective address bits to the LG-register is to hold the information for interrogation to establish conditions prior to execution and to control the execution of the instruction.

2-112. In the normal sequence of instruction preparation (effective address computation) the instruction arrives in the D-register in an early phase. In order to fetch the operand from memory, the sequence of data flow takes the displacement field of the instruction in D through the adder for modification, and the final phase of preparation leaves the effective address in the LA-register. At the same time the displacement field is placed in the adder the same bits are placed in the LG-register. Also, at the end of the prep phase the effective address in the LA-register is placed in the LG-register. This information arriving in the LG-register at the end of the prep phase may be the same as that deposited from the D-register in the early phase or it may be modified information as computed by the adder. In either case, the effective address bits are held in the LG-register.

2-113. Inasmuch as the instructions involved with the LG-register do not require an operand from memory, a memory cycle is inhibited and the LG-register contains the effective address of the instruction which is a group of function/operation code bits. These bits are interrogated to control the execution of these non-memory reference type instructions.

2-114. For example, if the instruction is a register copy, the destination register address (LG09-11) as well as the decision to invert the contents of the source register (LG12), are determined during the early prep phase. Also during the early prep phase, if the instruction is a read direct control used for setting the multiple precision mode operation, the contents of LG are the number of registers involved (LG10-12) and the address of the first register (LG13-15) in a sequence of registers. This information is then transferred to the LM-register, Refer to LM-register, paragraph 2-117 and RA-register, paragraph 2-121 for further description.

During this period, the contents of the LG-register are also interrogated for an all-zero condition. If this condition prevails it indicates that the read direct control instruction was for reading the state of the PCP data switches into the accumulator (a general register in fast memory) and not for setting the multiple precision mode. In this case, the transfer of data from the LG-register to the LM-register is inhibited and other control terms are initiated to carry out the execution of reading the data switches.

2-115. At the end of the prep phase, when the effective address is again placed in the LG-register, the LG-register bits are interrogated. If the instruction is a register copy, LG08 controls the decision to clear the destination register before the execution of the instruction. If it is a shift operation, the LG-register contains the type of shift (LG08-10) and the number of shifts (LG11-15). The LG bits indicating the type of shift are used to control the shift instruction execution and the number of shifts are placed in the RA-register for down count. Refer to the RA-register, paragraph 2-121.

2-116. For the remaining non-memory reference instructions (internal read/write direct control) the end of the prep phase leaves the function bits (effective address) in the LG-register. If the instruction involves the general of I/O channel registers in fast memory, LG10-15 are the address bits for the register; these bits are transferred to the RA-register. Refer to RA-register, paragraph 2-121. If the instruction involves an I/O operation (read direct) LG11-15 are the I/O function bits indicating an AIO, HIO, TVD, TIO or SIO, respectively. These bits are output to the IIOP interface during the execution of the I/O instruction. For all other remaining internal read/write direct control instructions, the individual bits, LG08-15 are used to control the instruction execution.

2-117. LM-Register. The LM-register is a 6-bit register (refer to figure 2-15) with the primary purpose of holding effective address bits 10 through 15 of an internal read direct control instruction to set the multiple precision mode operation. Part of the register (LM10-12) is used as a down-counter during the execution of a multiple load or store operation.

2-118. The effective address bits are transferred to the LM-register from the LG-register during the prep phase of the instruction. At this time the LM-register holds the number of registers (LM10-12) involved in the multiple load/store operation and the address of the first register (LM13-15) in the sequence of general registers in fast memory. These address bits are transferred to the RA-register. Refer to the RA-register, paragraph 2-121.

2-119. During the execution of the multiple load/store operation the number of registers in the LM-register is decremented under the control of the LM-register logic which in turn is directed by the instruction decode/execution control logic. This down-count effects an up-count of the RA-register to select each sequential register and also an increment of the effective address in the S-register to load/store data in sequential locations of memory.

2-120. When the CPU is in an idle state the data bits indicating the number of registers involved in the multiple operation (LM10-12) can be selected for display on the PCP by selection control switches on the PCP.

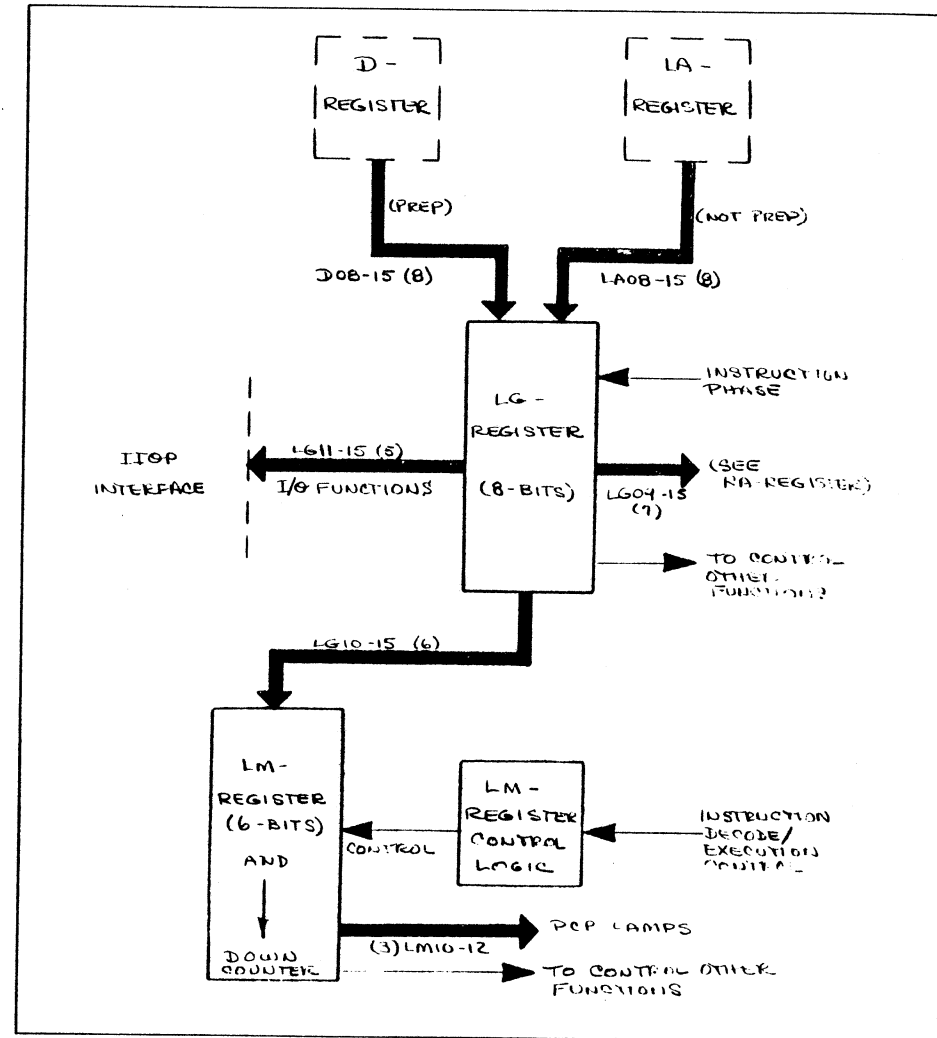


Figure 2-15. LG-And LM-Register, Data Flow Diagram

2-121. RA-REGISTER - (REGISTER FUNCTION)

The RA-register, a five-bit register, is primarily used as the address register that selects the registers in fast memory (general and I/O channel registers) through the fast memory control logic. The RA-register also serves as an up/down counter during instruction execution, I/O service calls and interrupt entry routines. Data transfer to the RA-register is commanded by the RA-register control logic which is directed by the instruction decode/execution control and I/O phase and control logic. Refer to figure 2-16.

2-122. During the execution of specific instructions (copy, read/write direct control, or shift) the LG-register holds the effective address (function) bits of the instruction. For the copy instruction, bits LG09-11 are the destination general register address. Bits LG12-15 are the number of shifts in a shift operation that are decremented by the RA-register and bits LG11-15 are the address of a general or I/O channel register address for a read/write direct control operation.

2-123. During the prep phase of a copy instruction, the three least significant bits of the D-register (D13-15) contain the address of the source register. These bits are transferred to the RA-register to control the selection of the source register before the execution of the instruction.

2-124. During a multiple precision mode operation, the LM-register holds the count of the number of general registers (LM10-12) involved in the operation, also the three-bit address (LM13-15) of the first register in a sequence of registers to be loaded or stored. The three address bits are transferred to the RA-register to select the first general register. While the count of the number of registers is decremented in the LM-register (refer to the LM-register, paragraph 2-117) the RA-register is incremented, thereby addressing each sequential general register. As the RA-register is incremented the effective address in the S-register is incremented to fetch or store data in sequential locations in memory.

2-125. Early in the preparation phases of an instruction, an RIXS bit, 07, from the O-register determines the first step in effective address computation. If this bit is high the contents of the base register (general register number 5) is selected to modify the displacement field to establish the reference address or the effective address of the instruction.

2-126. The RA-register logic also receives two bits, J07 and J08 from the J-bus which determines whether or not the extended accumulator register (No. 6) will be selected prior to instruction execution. These bits are concerned with a double register operation for shift or normalize.

2-127. The H-register control logic transfers a term to the RA-register which holds the least significant RA-bit low (subtracts 1 from the original count of 17) during a phase of the normalizing operation in a case where exactly 16 bit positions of shift results in normalization.

2-128. At the IIOP interface, during the first phase of an I/O service call, the device returns its address on the function response lines to the FRL-register. This data is decoded and is used to set the RA-register to control the selection of the first I/O channel register (even number) which contains the word address portion of the I/O command doubleword. The RA-register is incremented in the second phase, to obtain the flag and byte count portion of the IOCD from the odd numbered register. Also, during an I/O service call, the device is given a specific amount of time to respond to an order. The RA-register is incremented each clock and phase advance is inhibited until the device responds. If the device does not respond within 10.4 microseconds (up-count RA-register to = 11111₂) an interface timer runout indicator term will effect a CPU WAIT condition.

2-129. The RA-register is also used as an up-counter when it is counting the number of shifts taken during a multiply, divide or normalize operation.

2-130. When the CPU is in an idle state the contents of the RA-register can be selected for display on the PCP panel.

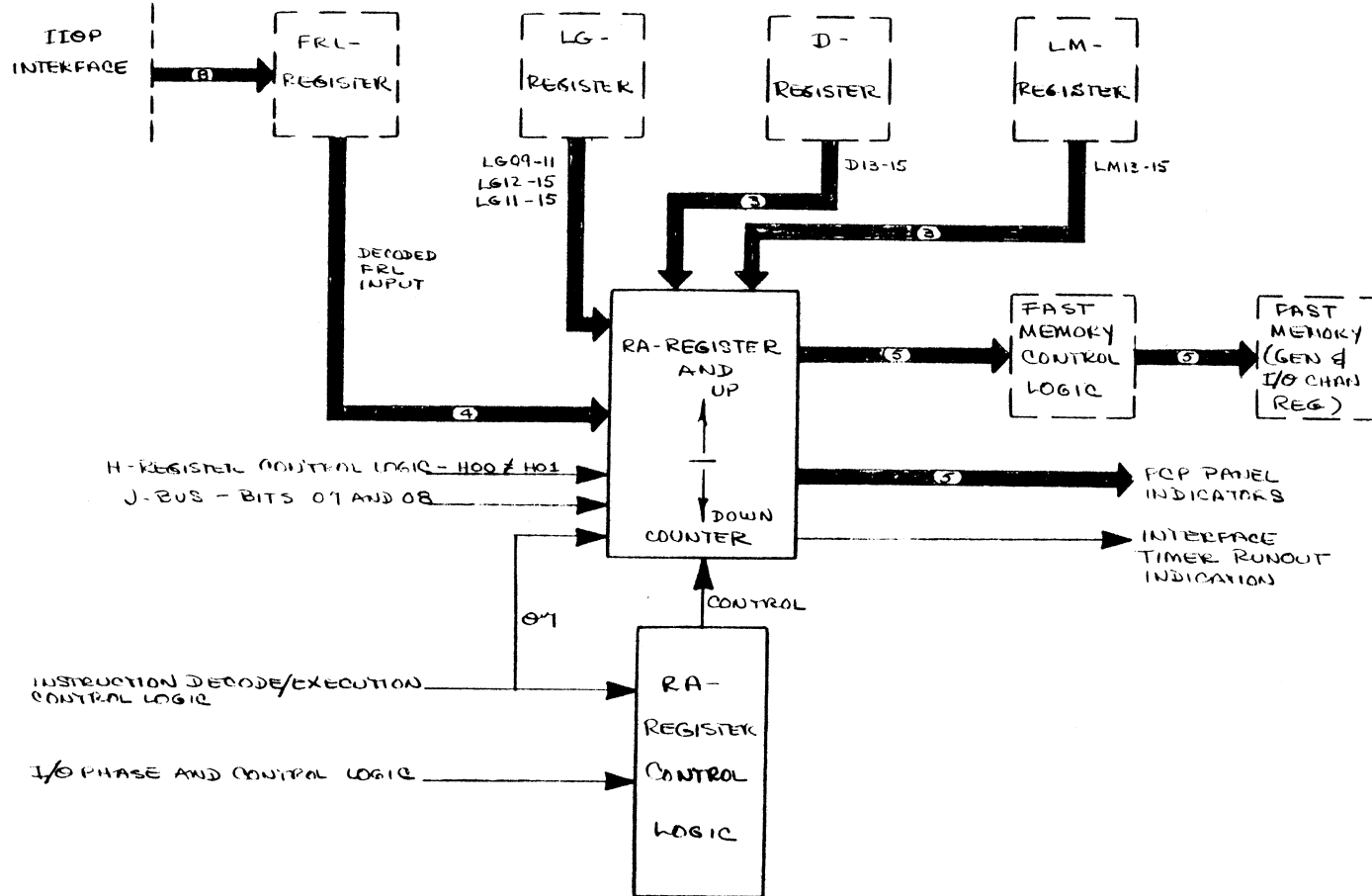


Figure 2-16. RA-Register Data Flow Diagram

2-131 GENERAL AND I/O REGISTERS (REGISTER FUNCTION)

A group of 16-bit registers, commonly called fast memory, is made up of eight general registers and four I/O channels, each channel consisting of two registers each. Refer to figure 2-12. Unlike all other registers in the CPU, these registers are available to the programmer (can be accessed).

2-132. These registers are numbered zero through fifteen and the binary equivalent of the number is the address of the register. The address is held in the RA-register of the CPU or is selected by switches on the PCP when the CPU is in the idle state.

2-133. The general registers are part of the arithmetical group in that the contents are used in carrying out an arithmetical or logical operation called for by an instruction. The I/O channels are used, when the I/O option is installed, to temporarily store the I/O command doubleword used during a service call. The even numbered register (8, 10, 12 or 14) holds the most significant part of the word or the word address for the location of the data table in core memory. The odd numbered register (9, 11, 13, 15) holds the least significant part of the word or the word containing three operation control flags (3 bits) and the byte count (13 bits) of the data table in core memory. During the execution of an I/O service call the device address returned from the device is decoded for input to the RA-register. The RA-register then holds the address of the most significant register (even numbered) of the associated I/O channel.

2-134. Control of the general and I/O channel registers is under the command of the fast memory control logic which is directed by the Instruction decode/execution control, I/O phase and control and Internal interrupt control logic. In order to write into the registers or transfer (read) the contents into another register or J-bus, the register must be addressed. The fast memory control logic selects the register and generates the register-write control when loading the registers in fast memory. When transferring the contents, the fast memory control logic again selects the register, however, the transfer is under the control of the logic associated with the destination of the contents.

2-135. Data flow to the register in fast memory is always from the J-bus. Data flow from a register can be onto the J-bus for input to another general register or another register of the arithmetical group. Data flow from a register in fast memory can also be to the H-register for arithmetical or logical operations and to the S-register during the multiprecision mode and extended arithmetic operations. If the memory protect option is installed the contents of the accumulator will be copied into a memory protect register during a write direct control operation. At this time, the contents (previously loaded) in the accumulator (Reg #7) determines the page (256 locations) or pages that are protected in core memory.

2-136. Although the general registers (0 through 7) are considered, for the most part, general purpose registers, the programmer must be aware of certain specifics regarding these registers. The function of each register can be described as follows:

General Register No. 0 (Z). This register can be accessed by a direct control instruction and a register copy instruction (only as a destination register). Other instructions interpret the address of this register as the contents of a register which equals all zeros.

General Register No. 1 (P). This register has a very specific purpose in that it holds the next address in the program. If the contents are changed, the program will go to the address where the register points.

General Register No. 2 (L). This register is known as the LINK register and is used by the programmer as the name implies.

General Register No. 3 (T). Besides serving as a general purpose register this register holds the exponent which is decremented during a normalizing operation.

General Register No. 4 (X1) and No. 5 (X2). These are general purpose registers, however, the registers also play an important role in effective address computation. A value in register X2 is used in the base addressing scheme and a value in X1 is used for post indexing. Incidentally, the P register can be included in this effective address computation group as it is referred to in relative addressing.

General Register No. 6 (E). This register is known as the extended accumulator. It is used as a general purpose 16-bit register, however, when the extended arithmetic option is installed, this register functions as the name implies, to extend the accumulator to the equivalent of a 32-bit register.

General Register No. 7 (A). This register is the very important and much used accumulator. Most instructions operate on the accumulator and also place the final results in the accumulator. The accumulator is also used in direct I/O communications.

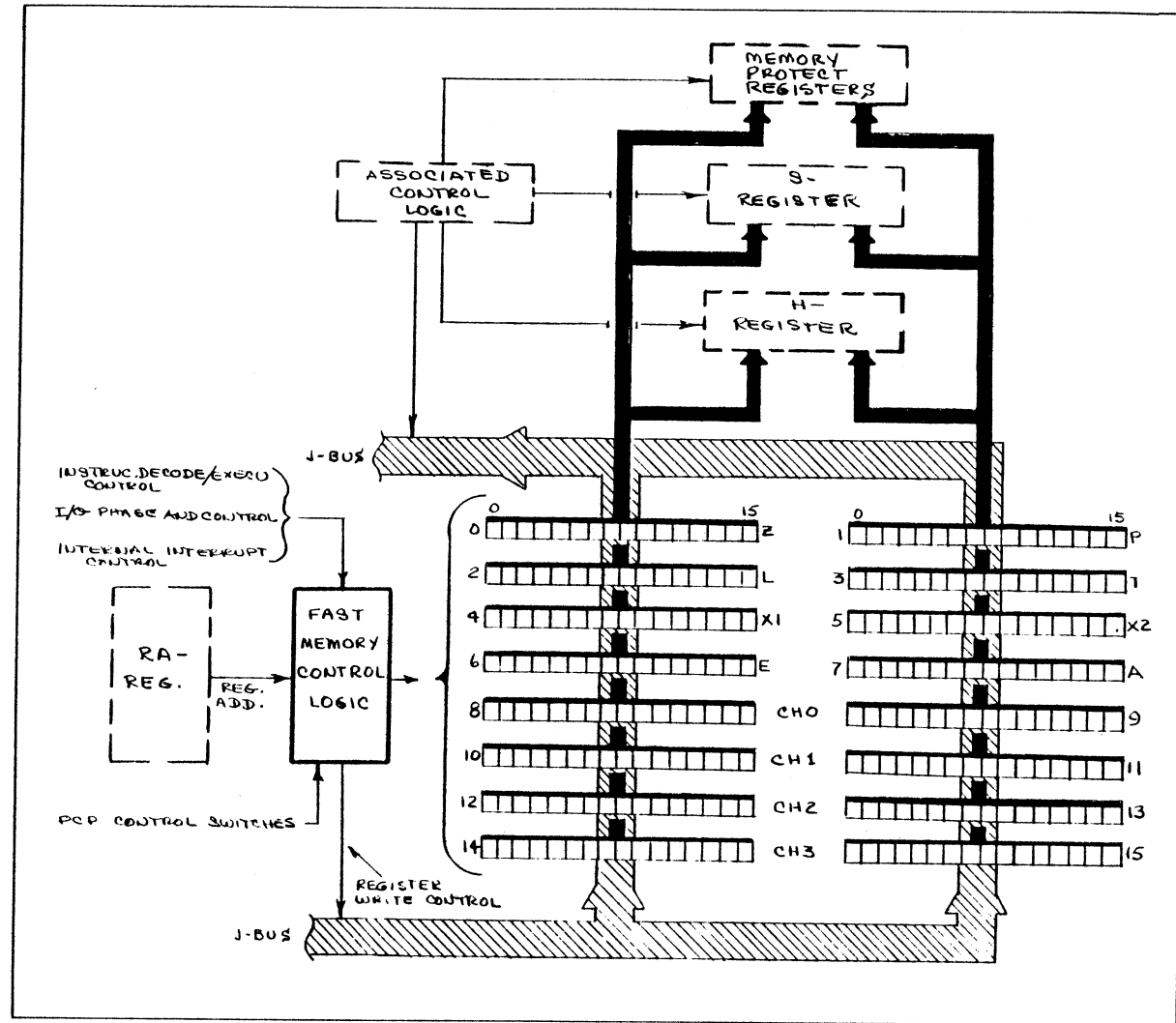


Figure 2-17. General and I/O Channel Registers (Fast Memory) - Data Flow Diagram

2-137 FRL-REGISTER - (REGISTER FUNCTION)

The main purpose for the FRL-register is to receive the data function response lines (8) at the IIOP interface. (Refer to figure 2-18). This data can be the device status or address during an I/O operation, or the device address at the beginning of an I/O service call. The address returned during an I/O service call is decoded by the FRL-register control logic and this code controls the setting of the RA-register to the address of the first I/O channel register associated with the device (refer to RA-register, paragraph 2-128).

2-138. The FRL-register also serves as a switching station for byte one (108-15) of the internal interrupt levels status from the I-register to the D-register during the execution of a read direct control instruction (interrupt mode). Data transfer from the I-register is directed by the D-register control logic (refer to D-register, paragraph 2-78).

During a normalizing operation, the FRL-register receives the number of shifts counted in the RA-register. This data is transferred under the control of the fast memory control logic. The count is then transferred to the D-register for input to the adder. At this time the count is subtracted from the exponent previously loaded in a general register in fast memory (T).

The state of a single bit from the LM-register (LM14) during the normalize operation controls the latch on the FRL-register. Also a single bit from the RA-register controls the subtraction of one from the shift count in the FRL-register if the condition of the normalize operation requires a minus one from the count.

The contents of the FRL-register may be selected for display by the PCP indicators when the CPU is in an idle state.

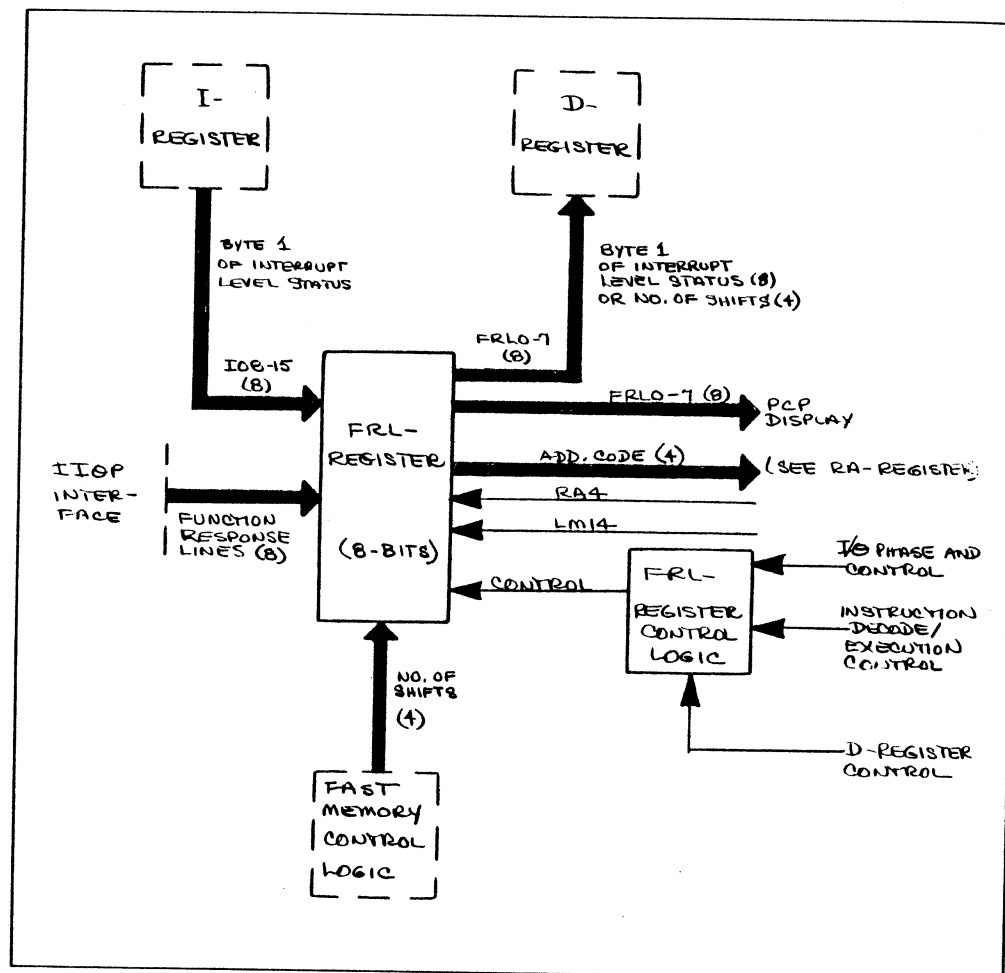


Figure 2-18. FRL-Register, Data Flow Diagram

SECTION III
DETAILED PRINCIPLES OF OPERATION

3-1 INTRODUCTION

This section contains two groups of detailed information, the first group describes the two modes of CPU operation, and the second group consists of logic diagrams covering the logic circuits involved in CPU operation. One mode of operation is the PCP control mode and the other is the program controlled mode of operation. As a further aid to understanding CPU operation, reference should be made to the applicable documentation in the list of related publications located in the front matter of this manual.

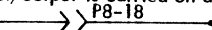
3-2 LOGIC DIAGRAMS

The second group of detailed information contained in this section are logic diagrams indicating the module location, type of logic mechanization used, and pin numbers of all logic terms generated and used within the CPU. These diagrams are all designated by the same figure number, however, each illustration is numbered in sequence with a sheet number (i.e.; sheet x of xx).

3-3. The logic diagrams are read from left to right and/or bottom to right. The signal interconnection between sheets is indicated by a sheet number in parenthesis at the left hand input to the diagram. A broken signal line appears only when a crossover makes it difficult to read the diagram. The signal interconnection between sheets does not reveal the actual wiring pattern. Sheet number input from the left indicates on which sheet the term is generated.

3-4. There are certain exceptions to the rules established in paragraph 3-3. The CPU clock input is applied to each circuit separately, signal interconnection is not necessary as a diagram of the CPU clock (CL, CK, or CLGAT) is provided showing clock output pin numbers of the clock source, modules 13C and 14C.

3-5. Another exception to the rules established in paragraph 3-3 is, that those pin locations where a signal (logic term) is not connected, the designation (open) is used immediately at the input pin to the circuit. This indicates that the pin assignment is open and that a logic level of one (4V) is hardwired in lieu of a logic term. Also, when a hardwired logic level of zero is required, it is designated as ZGNDXXX and this symbol appears immediately at the pin input to the circuit.

3-6. Where the logic term input/output is at a pin on a ribbon cable connector, the location is designated by placing the module location and the pin number within a square 32A
24. Where the logic term input/output is carried on a coaxial cable, a standard cable connector symbol is used  P8-18

3-7. PCP CONTROL MODE OF OPERATION

The PCP mode is concerned with (1) the switches and circuits controlling the operations required by the system operator and (2) the switches and circuits controlling the operations required by the maintenance personnel. The Sigma 3 system operator requires the use of a limited number of controls on the PCP panel. Figure 3-1 is a line drawing of the PCP panel showing all controls and indicators. Those controls and indicators which the system operator is specifically concerned with are indicated by the letter O. Maintenance personnel are concerned with all controls and indicators on the PCP. If a programmer requires the use of the PCP for debugging a program, he will use all controls except AUTO RESET and the SYNC COUNTER group of switches.

3-8. SYSTEM OPERATION PROCEDURE - PCP CONTROL

3-9. No effort is made in this manual to provide detailed information for the system operator. However, Table 3-1 lists the procedural steps for system turn-on, bootstrap program loading and console interrupt. This table is included to point the maintenance personnel to the logic circuits effected by the procedure used for system control.

Table 3-1. System Operation Procedures

Step	Procedure	Expected Results	Figure Ref.
1	Press POWER switch on (Note: S1 in PT14B is set to REMOTE)	PT14B Power Sup. is on PT15B Power Sup. is on PT16B Power Sup. is on PT17B Power Sup. is on POWER lamp is on (+8V from 16B is available)	3-2, Sh 1 3-2, Sh 1 3-2, Sh 2 3-2, Sh 2 3-2, Sh 1
2	Set key-operated switch to UNLOCK and PROTECT switch to OFF	-	3-2, Sh 1

Table 3-1. System Operation Procedures, Contd.

Step	Procedure	Expected Results	Figure Ref.
3	Set PCP switches in the MAINTENANCE area as follows: CLOCK to CONT, INTERRUPT to NORMAL, TIMER to NORMAL, AUTO RESET to OFF, INSTRUCTION to OFF, and OPERAND to OFF. In the OPERATION area, set the switches as follows: MEMORY MODE to NORMAL, and ADDRESS to NORMAL	NORMAL MODE lamp is on	3-3
4	Set DATA switches 8-15 to the number of the device from which the initial program is to be loaded. Set DATA switch 7, also, if the device is on the EIOP	The logic levels set by these switches are KD08-KD15 The logic level KD07, set by this switch is used to reset the load-conditions initiate (LCI) circuit when the simulated SIO instruction is completed and the CPU is in the WAIT state. (KD07+CF).WAIT If the subsequent ORDER is at the EIOP the LCI term must be reset at the IOP (CPU) for the service call.	3-12, Sh 1 3-12, Sh 3
5	Set COMPUTE switch to idle	This position enables +8V to the RESET and LOAD switches.	3-12, Sh 2 , Sh 3

Table 3-1. System Operation Procedures, Contd.

Step	Procedure	Expected Results	Figure Ref.
6	Press RESET switch	The logic level KRESET set by this switch is transmitted as RSTE/2 to system reset circuitry in memory. After a minimum delay of 5 usec, RST and RSTE/1 are returned to the CPU for use within. RST and RST/B enable the setting of D03, D09 and D15, for simulating a direct control read I/O instruction, SIO.	3-12, Sh 2 3-12, Sh 16 3-12, Sh 50 - 52
7	Press LOAD switch	The logic level KLOAD, set by this switch is used to set load-conditions-initiate, LCI. LCI is latched high until the conclusion of the simulated ORDER out for a device on the IOP. At this time LCI is reset by IOPH17. Refer to step 4 for device on EIOP. The setting of LCI effects the following: a. LCI overrides the interface timer circuits, (S/NCFIT . NOFIT/1) b. LCI sets a WAIT condition at the end of simulated SIO instruction, (LCI+OFIT+CFIT) . ENDE	3-12, Sh 3 3-12, Sh 3 3-12, Sh 25

Table 3-1. System Operation Procedures, Contd.

Step	Procedure	Expected Results	Figure Ref.
7 Contd		c. LCI sets up the transfer term JXKD=(LA08+LCI) to transfer device add. set up in step 4 to the J-bus.	3-12, Sh 41
		d. LCI inhibits the setting of JK15/1 (NRADDRXKRA . NLCI). H-reg. cannot be incremented for interface timer during SIO and the S-reg. cannot be updated during the subsequent service call.	3-12, Sh 32
		e. LCI sets H08 to simulate a byte count of 'X'80 (LCI. IOPH03). LCI also forces H15/1 (NLCI,NH15)=LCI+H15. This simulates an odd byte condition allowing the transfer of byte zero in the D-register to the IO data lines. This is the order byte (READ) set in D06 by LCI. (IOPH03 . LCI)	3-12, Sh 54 Sh 3
		f. LCI inhibits a memory cycle during a service call. (NLCI . (S/MC1/2)	3-12, Sh 48
		g. LCI inhibits JXR (IOPH02.NLCI), HXR (IOPH03.NLCI), and HXJ (IOPH05B.NLCI) during the service call	3-12, Sh 15 3-12, Sh 41 Sh 53 Sh 52

Table 3-1. System Operation Procedures, Contd.

Step	Procedure	Expected Results	Figure Ref.
8	Set COMPUTE switch to RUN	CPU clocks are started: KRUN. KCONT-1.CLB=CLOCKEN. CLOCKEN.CLA=CLOCK=CLEN/M.CLB=CLEN.CLA=CK & CL, and RUN lamp is on. The simulated SIO instruction is prepared, and executed, and the subsequent service calls are executed.	3-12, Sh 2 Sh 5 Sh 7 3-4 & 3-5 3-6
		NOTE: Figure 3-4 and 3-5 are self explanatory in regard to data flow during the prep and execution phases of the SIO (bootstrap). However, figure 3-6 requires the support of sequence table 3-2 for explanation of data flow during the execution of the I/O service cycle. The order out (LCI) and subsequent data in (READ) cycles are indicated.	
	Press INTRPT switch	The logic level KINTRPT set by this switch is one of three conditions used to set INTK. INTK is the program interrupt initiated by (1) the INTRPT button (2) logic level TRACE set by INTERRUPT sw during prep (3) logic level DIAGNOSTIC set by INTERRUPT sw at MPE	3-12, Sh 7A Sh 91
		NOTE: When INTK is set high, and if the console interrupt is armed and enabled (I11P and I11N) I11S is set and I11R is high. I11R is decoded resulting in INTAD04 and INTAD02. With the I/O Group 0 interrupts effected (I1119R) INTAD01 is high. These interrupt address decode lines are gated by INTGA and the resultant interrupt address location lines INTAD07, 13, 14 and 15 are high. This configuration of INTAD = a binary equivalent of 263 or 'X'107 which is the address in memory dedicated to the interrupt entry routine for the console interrupt.	

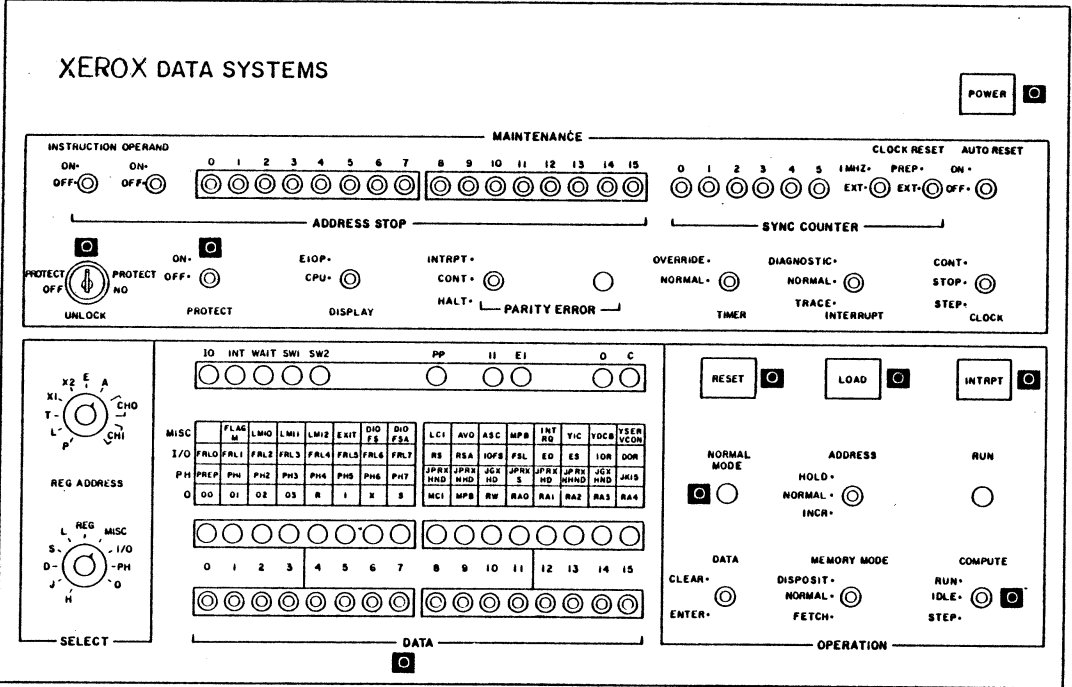


Figure 3-1. Processor Control Panel - PCP Control Mode of Operation

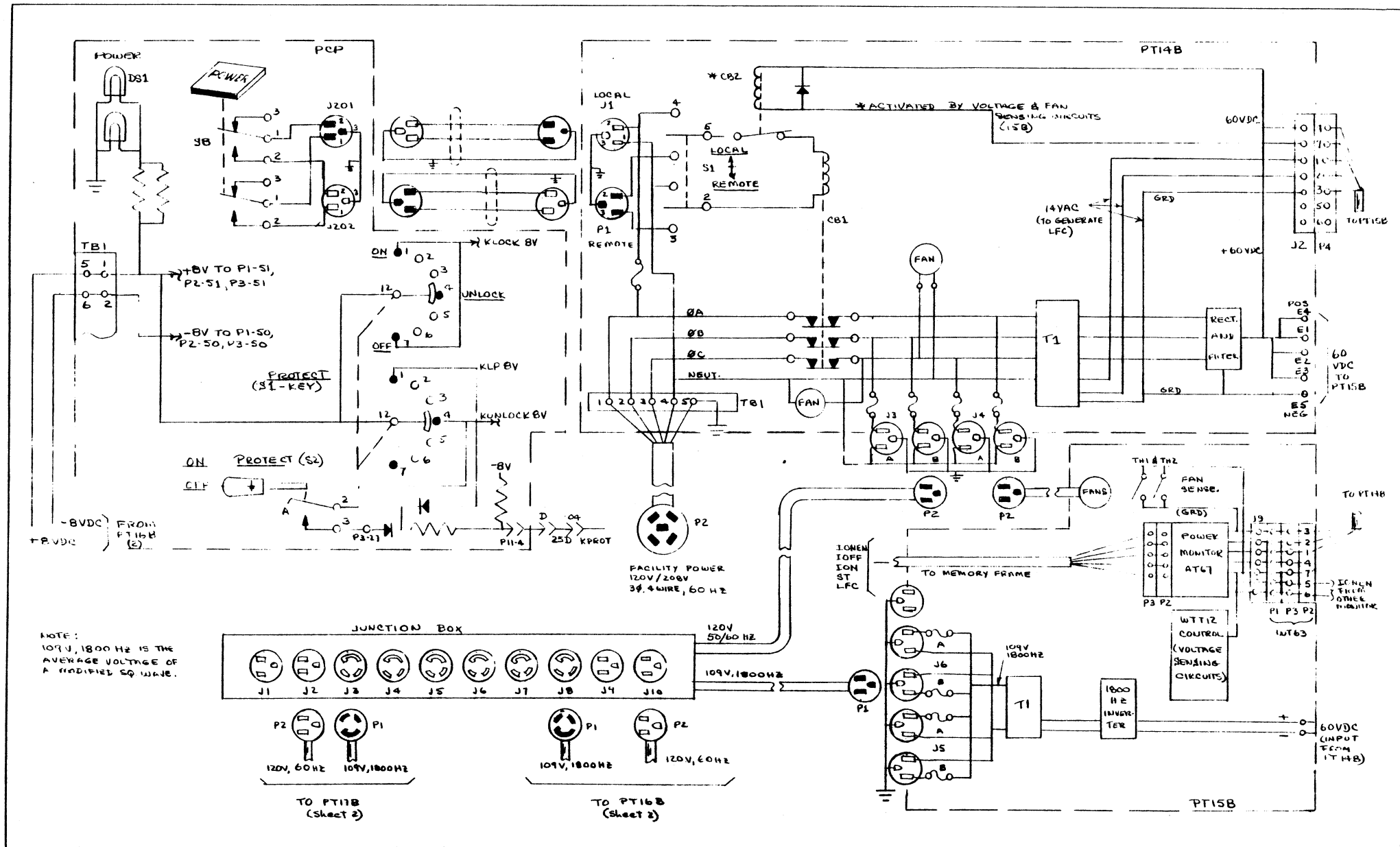


Figure 3-2. Power Distribution Diagram (sheet 1 of 2)

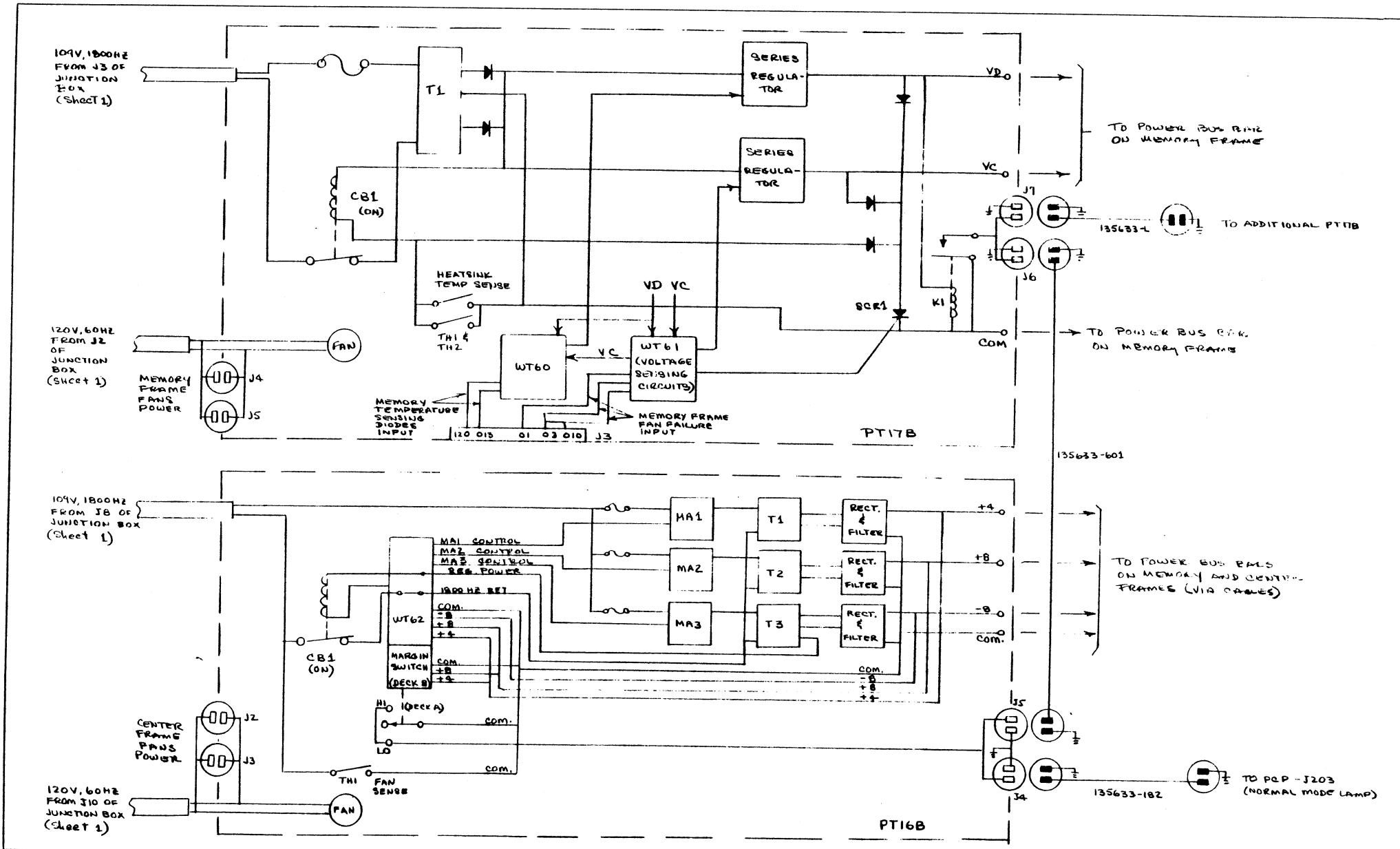


Figure 3-2. Power Distribution Diagram (sheet 2 of 2)

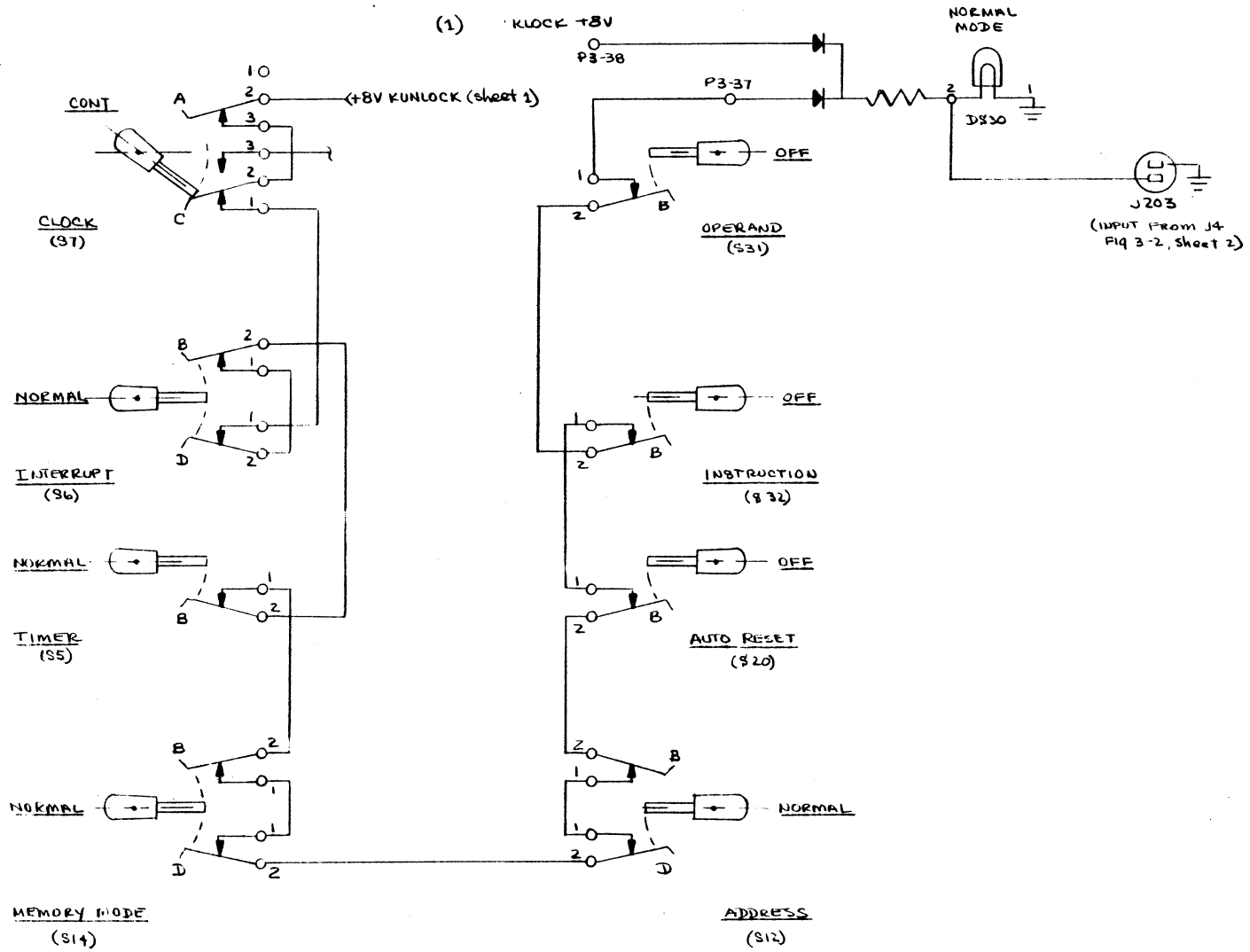


Figure 3-3. Conditions for Normal Mode Indication

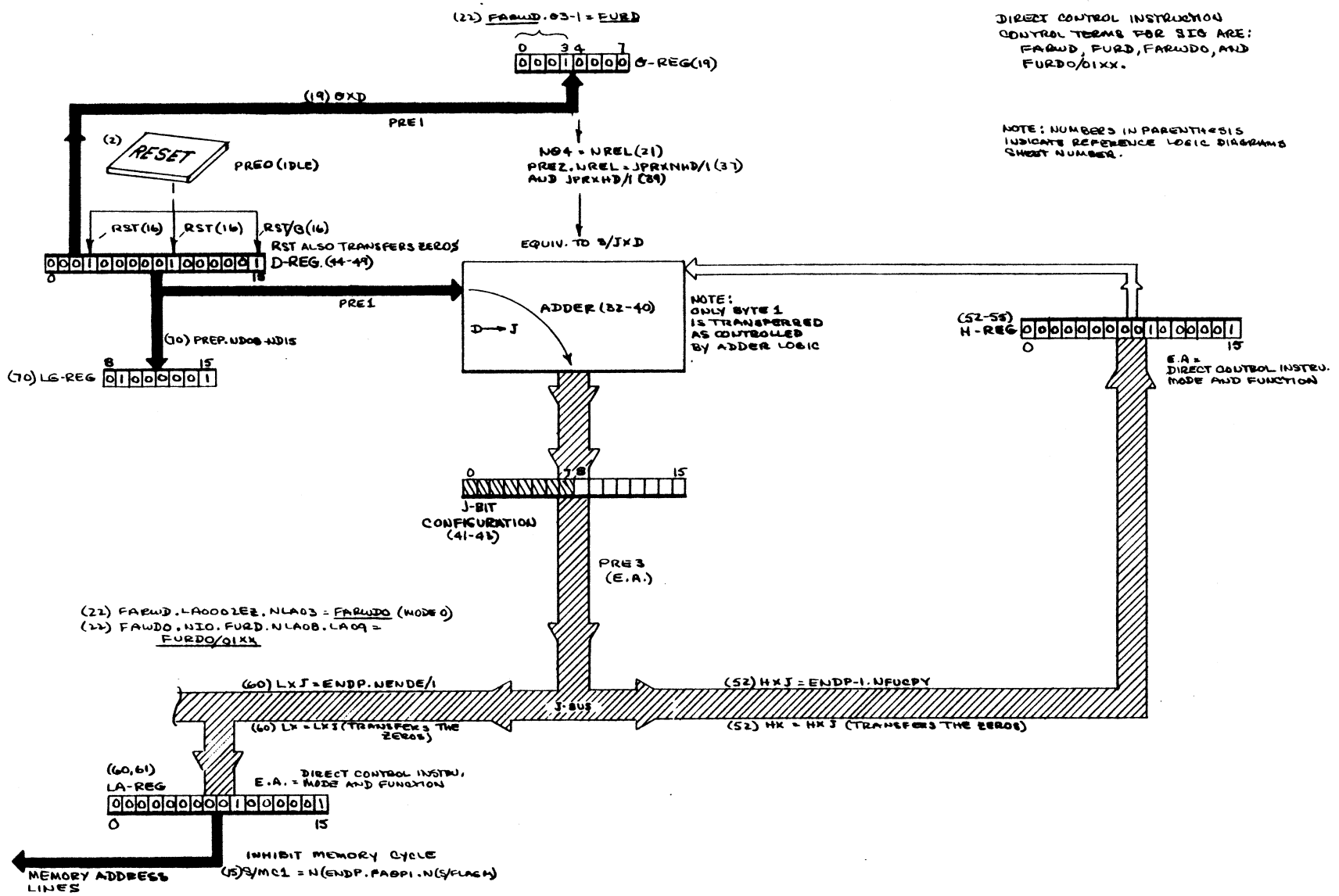


Figure 3-4. Data Flow During Prep Phase of SIO (Bootstrap, LCI)

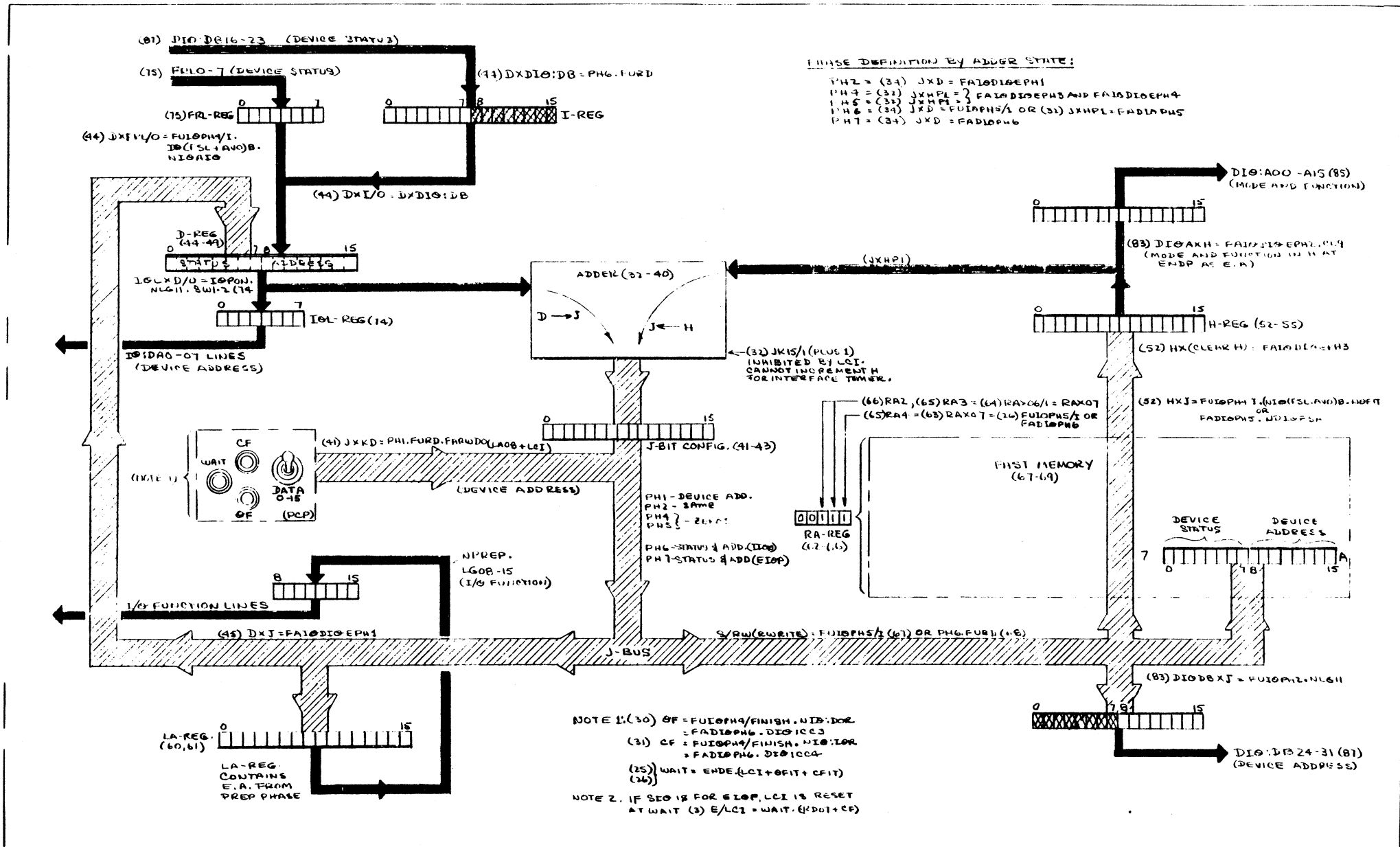


Figure 3-5. Data Flow During Execution of SIO (Bootstrap LCI)

Table 3-2. I/O Service Connection Sequence Control

Phase Control by Adder-State-Rognition				
I/O Phase	Hex Display	Basic Operation	Associated Terms	Comments
01	88	JXH	S/IO/1.NSW1.NSW2	Phase Definition
02	00	JACEZ	JACEZ.SW1	Data Table Address (WA) in even No. I/O channel register is transferred to J-bus - JXR (IOPH03.NLCI)
03	C0	JXHEORD	IOPH02.FSL.NSW1.NSW2	Phase Definition
04	C0	JXHEROD	IOPH03.SW1	Phase Definition
05	66	JXHMI	IOPH04.SW1	Decrement byte count (BC) in H-reg.
06	11	JXSP1	IOPH05B.NSW1.NSW2	Increment WA from S-register
07	66	JXHMI	IOPH0609.NIOED.NBCEZ	Decrement byte count in H-reg. if not end data and not count done
08	02	JGXHND	IOPH07.SW1	Phase definition
09	11	JXSP1	IOPH08.SW1	Increment WA from S-register
10	02	JGXHND	IOPH09.IOG.H01	1st Data Chain Phase
11	11	JXSP1	IOPH10.SW2	2nd Data Chain Phase - Increment WA from S-register.
12	48	JXD	IOPH11.SW1	3rd Data Chain Phase - 1st and 2nd word of new IOCD in D-reg. transferred to J-bus.
13	48	JXD	IOPH12.IOG.NSW1.NSW2	Terminal Order Phase Definition - To insure transfer of MPE bit in D-reg. is transferred into 2nd word of new IOCD
14	49	JXDP1	IOPH13.JK15	Phase Definition
15	10	JPRXS	IOPH14.(IOADV+OFIT)	Updated WA in S-register is transferred to J-bus.
16	88	JXH	IOPH15.SW1	Updated BC in H-register is transferred to J-bus.
17	00	JACEZ	JACEZ.NSW1.NSW2	Adder unused. Next instruction address from P-reg. is on J-bus - JXR=ENDE/2

Table 3-2. I/O Service Connection Sequence Control, Contd.

Control by Data Transfer Terms				
I/O Phase	Hex Display	Basic Operation	Associated Terms	Comments
		FRLXIO:FR	IO(FSL+AVO)B	FRL-register contains device address when device returns FSL or AVO
		FRLX	IO(FSL+AVO)EN	FRL-register is cleared prior to IO (FSL+AVO)B
		IOADV	IOPH01.IO(FSL+AVO)B	At receipt of FSL advance out of IOPH01. At receipt of AVO or OFIT advance out of IOPH01 to IOPH17 and bring up next instruction address from P-register.
			IOPH05060914.(IORSB+OFIT)	At receipt of RS advance out of IOPH05 (all ORDERS). At receipt of RS advance out of IOPH09 (Data-In/Data-Out). At receipt of RS advance out of IOPH14 (Terminal Order). At OFIT advance to IOPH15 and establish WAIT.
			IOPH09.IOG	Used only in IOPH09 where final byte is being stored (Data-In). No RS is necessary - this inhibits IT upcount.
		RAUC/1	IOPH01.NIOADV/1	RA-register is set for use as IT (counts up until the receipt of FSL)
			IOPH02.(NIOAVO.NOFIT)	RA-register is set to increment to odd I/O channel address
			IOPH05060914.NIOADV/2	RA-register is set for use as IT (counts up until receipt of RS)
			IOPH11.IOG	RA-register is set to increment to odd I/O channel address to insert MPE flag in 2nd word of new IOCD
			IOPH15	RA-register is set to increment to odd I/O channel address to transfer updated 2nd word of IOCD
		RAX01	RAX01/EN.IOADV	RA-register is set to 1 prior to use as IT in IOPH01 or to address of P-reg. (1) at a forced ENDE/2
			IOPH03.NIORST	RA-register is set to 1 for IT in IOPH05 or to P-reg. add. (1) at a forced ENDE/2
			IOPH08 IOPH13	RA-register is set to 1 for IT in IOPH09 and IOPH14 or to P-reg. add. (1) at a forced ENDE/2
			IOPH16	RA-register is set to address of P-reg. at ENDE/2 for next instruction ADD.

Table 3-2. I/O Service Connection Sequence Control, Contd.

Control by Data Transfer Terms, Contd.				
I/O Phase	Hex Display	Basic Operation	Associated Terms	Comments
		RA-register	(During IOPH02,05,06, 10 and IOPH15) (During IOPH03 and 16) (During IOPH12-)	The RA-register contains the decoded device address which is the address of the even no. I/O channel register. The RA-register contains the address of the odd no. I/O channel register. As a result of an MPE indication (see RAUC/1) the RA-register contains the address of the odd no. I/O channel address.
		DXMRC	(MC2.NMWRITE)-present in IOPH04-all orders IOPH08-Data (3 passes) IOPH11-IOCD (2 passes)	D-register contains a word from the data table as the result of a memory cycle - S/MC1 = LXJ.NLCI is preset in IOPH02. It is also preset in IOPH06 and IOPH09 if the BC is even (NH15) after decrementing in IOPH05 and IOPH07. A memory cycle is inhibited if BC is odd (IOPH060914IOR.H15) or end data is high (IO:ED), count is not done and data chaining is not called for (IOPH060914IOR.(IOED/EN + IO:ED). NIO (H01.BCEZ).
		S/MWRITE	IOPH0609BDI.(IOED+NH15). NIOG	D-register contains a word to be written into the data table after data from a device has altered the word from the data table. A memory cycle is required for writing-in and S/MC1 is preset in IOPH06, IOPH09 if the BC is even. A memory cycle is inhibited if the BC is odd (refer to DXMRC for conditions). In order to write during a memory cycle, S/MWRITE must be high (MREAD low). This takes place in IOPH06 and 09 if the BC is even (NH15) or end data is present and count is not done.
		IOLXD/0	IOLXD/EN.H15.IORSBEN	During an Order-out/Data-out, byte 0 is transferred to the IOL-register for output to the device if RS has been received, the BC is odd and count done is not present.

Table 3-2. I/O Service Connection Sequence Control, Contd.

Control by Data Transfer Terms, Contd.				
I/O Phase	Hex Display	Basic Operation	Associated Terms	Comments
		IOLXD/1	. . NH15 . .	(Refer to IOLXD/0 ... Byte 1 in D is transferred to the IOL-register for output to the device if the BC is even, etc.)
		DXIO:DA/0	IOPH0609BDI.IORSB.H15	Device data is transferred to byte 0 in the D-reg. if RS is received and BC in IOPH05 and 07 is odd.
		DXIO:DA/1	. . NH15 . .	Device data is transferred to byte 1 in the D-reg. if RS is received and BC is even (refer to DXIO:DA/0)
		LXJ/1	IOPH02.(NIOAVO.NOFIT)	Word Address (WA) of data table transferred to LA-register if AVO has not been received and IT has not runout.
			IOPH0609B.IOIOR.NH15	Incremented WA transferred to LA-reg. if order/data-out and BC is even after decrement in IOPH05/IOPH07.
			ENDE/2	Next instruction address transferred from P-register.
		SXJ	IOPH02	WA from I/O channel register (even) to increment.
			IOPH0609B-1.(NIOG.NH15)	To increment WA IOPH06 and IOPH09 if BC is even and count is not done.
			IOPH11.NIOG	Address of 1st word of new IOCD to increment for second word.
		LXS	IOPH0609B.IODI.H15.SW1 (Data-In)	Incremented WA in S-reg. transferred to LA-register during IOPH09 only, if BC in IOPH07 is odd. LXS is inhibited in IOPH06 if BC is odd in IOPH05.
			IOPH09.IOG.H01	1st phase of Data Chain. WA of 1st word of IOCD is transferred to LA-reg.
			IOPH12.NIOG	3rd phase of Data Chain. WA of 2nd word of IOCD is transferred to LA-register.

Table 3-2. I/O Service Connection Sequence Control, Contd.

Control by Data Transfer Terms, Contd.				
I/O Phase	Hex Display	Basic Operation	Associated Terms	Comments
		HXJ	IOPH05B.NLCI	Transfer decremented BC back into the H-register. This is inhibited for LCI. Transfer decremented BC back into the H-register, as long as count is not done. For LCI, transfer all zeros to H-reg. instead of BC from I/O channel reg. (odd) Zeros to H-register at ENDE/2 Preset for transfer of Order-in from device to the even No. I/O channel reg. Preset for transfer of updated WA to I/O channel reg. (even) when end data is received. IOPH13 definition - Preset RW for transfer of MPE bit in T.O. to flag position in odd I/O channel register. Preset for transfer of updated WA in even I/O channel register. Preset for transfer of updated BC in odd I/O channel register.
			IOPH07.NIOG	
		HX	IOPH03.LCI	
		S/RW/1	IOPH17	
			IOPH060I	
			IOPH0609B-1.IOEDZDATA	
			IOPH12.IOG	
			IOPH14.IORSB.NIOOI. (NH01+NBCEZ) IOPH15.NIOOI. (NH01+NBCEZ)	

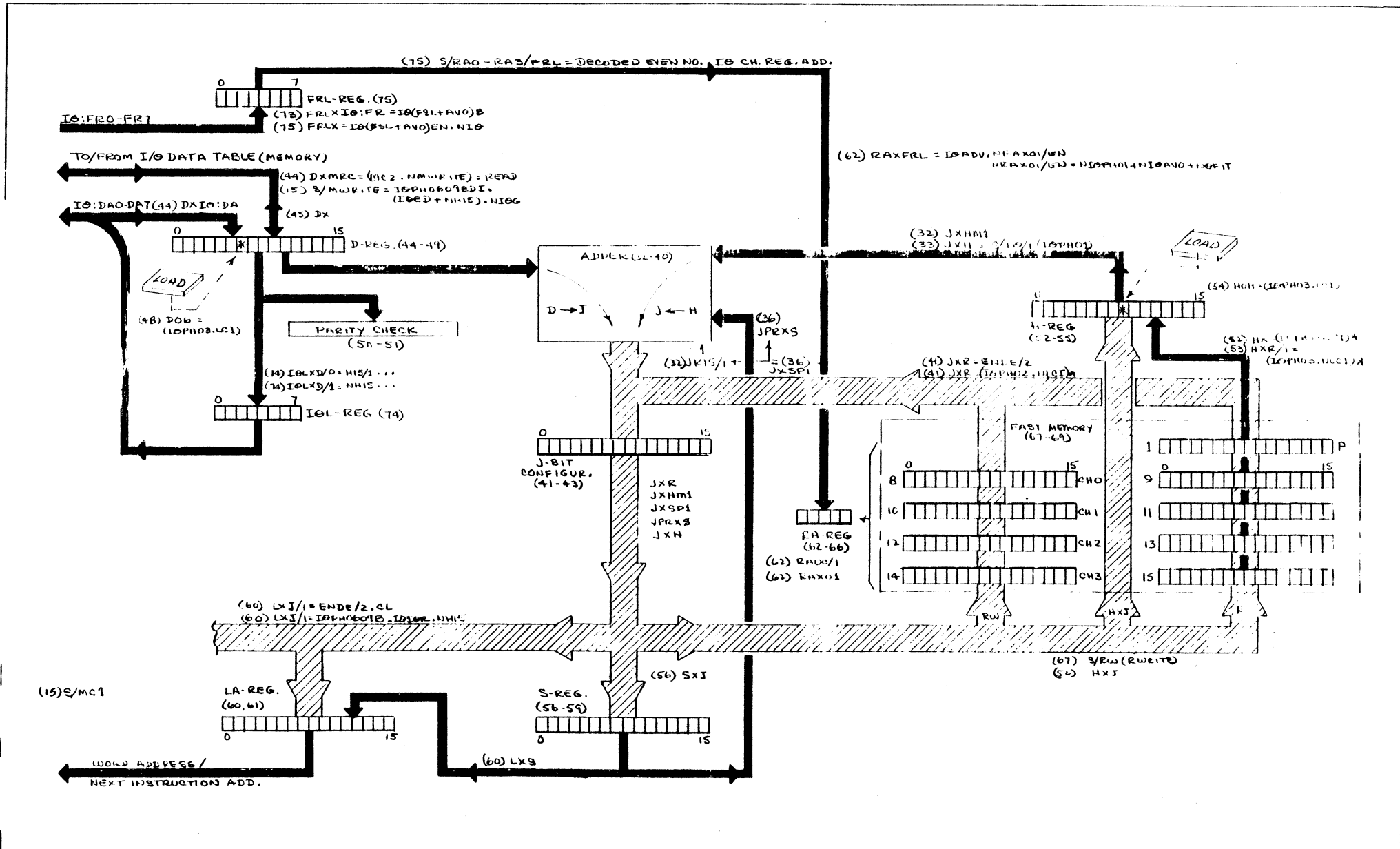


Figure 3-6. I/O Service Call Data Flow Diagram

3-10 FETCHING AND STORING DATA - PCP CONTROL

3-11. Table 3-3 contains step-by step procedures for fetching and storing data using the switches on the PCP panel. This table is included to point the maintenance personnel to the logic circuits effected by these procedures.

Table 3-3. Fetching and Storing Data Procedures - PCP Control

Step	Procedure	Expected Results	Figure Ref. 3-12
Procedure A: Fetch Data From Memory Location and Display Data			
1	Set COMPUTE switch to center position (idle)	Logic level KIDLE set by this position resets the NIDLE (STEP/RUN) latch. NIDLE+KCONT-1 = PRE0 (Idle phase)	Sh 2 Sh 17
2	Set DATA toggle switches 0-15 to desired memory address	Logic levels set by these switches are KD00 - KD15	Sh 1
3	Rotate SELECT switch to S	Logic level KSEL/S is high. KSEL/S is used to gate data transfer term SXJ when DATA switch (step 4) is actuated. KSEL/S also gates the contents of S to the lamp driver circuits for display	Sh 9 Sh 56 Sh 10-13
4	Momentarily set DATA switch to ENTER	Logic level NKENTER/B is low. JXKD is high and the address set by switches in step 2 is transferred to the J-bus. Logic level KENTCLR/B is high and transfer term SXJ is high (KSEL/S, KENTCLR/B = SXJ). Logic level KENT/CLEAR is high. KENT/CLEAR, IDLE, CLB = STEP. STEP, IDLE, CLA = CLOCK = CLEN = CL, CK and RUN lamp is on. A momentary burst of clocks loads the S-register with address on the J-bus.	Sh 8 Sh 41 Sh 56 Sh 8 Sh 5 Sh 7 Sh 58
5	Set MEMORY MODE switch to FETCH	Logic level NKPROGRAM is high and KDEPOSIT is low. NKPROGRAM, PRE0 = LXS. At the receipt of the first CL (step 6) the address in S is transferred to the memory address latches (LA-register). With KDEPOSIT low, S/MWRITE is low (MWRITE/SEN1, KDEPOSIT, PRE0) making MREAD high.	Sh 8 Sh 60 Sh 60,61 Sh 15

Table 3-3. Fetching and Storing Data Procedures - PCP Control, Contd.

Step	Procedure	Expected Results	Figure Ref. 3-12
6	Procedure A Contd. Momentarily set COMPUTE switch to STEP and return MEMORY MODE SW TO NORMAL	LXS = S/MC1. A memory cycle is initiated. Logic level KSTEP is high. KSTEP = NIDLE. NIDLE, KCONT-1 = STEP = CLOCK = CLEN = CK, CL and RUN lamp is on. A burst of clocks executes the memory cycle initiated in Step 5 and data is fetched from address in LA-register and deposited in the D-register. MC2.NMWRITE = DXMRC/1/0. MRC00-15.DXMRC/1/0=D00-D15.	Sh 2 Sh 5 Sh 7
7	Rotate SELECT switch to D	Logic level KSEL/D is high. KSEL/D.ND00-ND15 = KDIS00-KDIS15. Contents of D-register is displayed on PCP by lamp driver circuits. (KDIS00-15.NDISPLAY/J) = KLD00-KLD15	Sh 9 SH 10-13
Procedure B: Fetch and Display Data From Successive Locations			
1	Perform STEPS 1-7 of Procedure A	Contents of the first selected address is in the D-register and is displayed. Note: When COMPUTE switch is released from Step to Idle (PRE0) S/JXSP1 is high. The contents of S-register (Step A4) is transferred to the adder (S/JXSP1 = S/JPRXS) and a one is placed in the LSB position of the adder (S/JXSP1 = JK15/1). An addition is performed and the incremented address is on the J-bus at 1st clock (PRE1)	Sh 36 Sh 32

Table 3-3. Fetching and Storing Data Procedures - PCP Control, Contd.

Step	Procedure	Expected Results	Figure Ref. 3-12
2	Procedure B Contd. Set ADDRESS switch to INCR	Logic level KADDRINCR is high. At the 2nd clock (PRE2) the incremented address on the J-bus is transferred to the S-reg. (KADDRINCR.PRE1= SXJ).	Sh 8 Sh 56
3	Momentarily set COMPUTE switch to STEP	At the first STEP (burst of clocks) the contents of D do not change since the address in the LA-register did not change. However, at the 2nd setting to STEP the contents of the next sequential location is in the D-register. This will continue at each following STEP setting.	
4	Return PCP switches to NORMAL MODE condition	(Refer to table 3-1, step 3)	
Procedure C: Store Data in a Designated Memory Location			
1	Perform Steps 1 through 4 of procedure A	The S-register contains the address of the memory location.	
2	Rotate SELECT switch to D	Logic level KSEL/D is high. KSEL/D gates the transfer term DXJ when DATA switch (Step 4) is actuated. KSEL/D also gates the contents of D to the lamp driver circuits for display.	Sh 9 Sh 10-13

Table 3-3. Fetching and Storing Data Procedures - PCP Control, Contd.

Step	Procedure	Expected Results	Figure Ref. 3-12
3	Procedure C Contd. Set DATA toggle switches 0-15 to desired storage value	Logic levels set by these switches are KD00-KD15	Sh 1
4	Momentarily set DATA switch to ENTER	Logic level NKENTER/B is low. JXKD is high and the desired storage value set in Step 3 is transferred to the J-bus. Logic level KENTCLR/B is high and transfer term DXJ is high (KSEL/D.KENTCLR/B = DXJ). Logic level KENT/CLEAR is high. KENT/CLEAR.IDLE.CLB = STEP. STEP.IDLE.CLA = CLOCK.CLB = CLEN.CLA = CL, CK and RUN lamp is on. A momentary burst of clocks loads the D-reg. with desired storage value, and this value is displayed.	Sh 8 Sh 41 Sh 45 Sh 8 Sh 5 Sh 7
5	Set MEMORY MODE to DEPOSIT	Logic levels KDEPOSIT and NKPROGRAM are high. Data transfer term DXJ remains high (KDEPOSIT.PRE0 = DXJ). NKPROGRAM.PRE0 = LXS. At 1st clock the address of memory location is transferred from S to the memory address latches (LA-register). LXS = S/MC1 and a memory cycle is initiated. MREAD is low as S/MWRITE is high (MWRITE/SEN1. KDEPOSIT.PRE0)	Sh 8 Sh 45 Sh 60 Sh 61 Sh 15

Table 3-3. Fetching and Storing Data Procedures - PCP Control, Contd.

Step	Procedure	Expected Results	Figure Ref. 3-12
6	Procedure C Contd. Momentarily set COMPUTE switch to STEP and return MEMORY MODE TO NORMAL	Logic level KSTEP is high. KSTEP = NIDLE NIDLE.KCONT-1 = STEP = CLOCK = CLEN = CK, CL and RUN lamp is on. A burst of clocks executes the memory cycle initiated in step 5 and the data in the D-register is stored in the memory location established in step 1.	Sh 2 Sh 5 Sh 7

3-12 PROGRAM CONTROLLED OPERATION

3-13. When under the control of a program, the data flow within the CPU is according to the instructions contained in the program (software). Each instruction is interpreted, prepared and executed by the logic circuits (hardware) which constitutes the CPU.

3-14. When the Sigma 3 system is performing in the normal RUN mode of operation, the CPU logic circuits are automatically executing the program (sequential instructions) controlling the processing of data. During program or machine maintenance procedures, however, it may be necessary to step through a program or a section of the program and thereby execute and stop after the execution of each instruction. This type of procedure is manually controlled at the PCP by setting the MEMORY MODE and ADDRESS switches to NORMAL and momentarily setting the COMPUTE switch from idle to STEP, each STEP thereby fetching and executing the next sequential instruction.

3-15. The PCP also provides the means to manually clock-step through the preparation and execution phases of an individual instruction. It is this mode of operation that is used in this section to describe the detailed principles of the program-controlled operation of the CPU.

3-16 INSTRUCTION PREPARATION SEQUENCE

The instruction preparation sequence is the interrogation of the instruction to establish control logic levels for the execution of that particular instruction. All instruction operation-code fields (bits 0-3) are interrogated in the same manner, however, the remaining RIXS field (bits 4-7) and displacement field (bits 8-15) for all instructions except COPY, are examined to establish a 16-bit effective address. This effective address falls into two classifications, one where it is used to access memory and the other where it is used to establish the mode and function of the instruction. The effective address used to access memory involves the Memory Reference instructions (LDA, STA, LDX, ADD, SUB, AND, IM, and B) and the Branch on Condition (BOC). The Memory Reference instruction accesses memory for the operand of the instruction and the BOC accesses memory for the next instruction. The effective address to establish mode and function involves the SHIFT and DIRECT CONTROL instructions.

3-17. No effective address is established for the COPY instruction, the RIXS and displacement fields are not used as such. Bits 4 through 15 are interrogated to establish the necessary control logic levels for executing the operations between any two general registers within the CPU.

3-18. The effective address of an instruction is established by the examination of the RIXS field to compute the final 16-bit address value, starting with the displacement field value of the instruction. For the effective address to access memory, the R and S bits (4 and 7) are examined to determine the reference address which can be non-relative (R=0, S=0) or base-relative (R=0, S=1) or R bit alone, self-relative (R=1). The BOC instruction automatically invokes self-relative addressing and the effective address is computed in this manner. Bits 4, 5, and 6 are examined to establish the conditions for branch).

3-19. The I-bit is examined to determine the direct address, this is equal to the reference address (I=0) or the reference address is treated as an indirect address (I=1). The X-bit is examined to determine the effective address which is equal to the direct address (X=0) or the direct address plus the contents of general register X1 (X=1).

3-20. For those instructions where the effective address is used, not for memory access but to establish mode and function, the RIXS field invokes non-relative, direct addressing (RIXS=0000) or base-relative, direct addressing (RIXS=0001). In the first case, the configuration of the displacement field (8-15) and zeros extended from bit 7 to bit zero, the effective address equals mode 0 (bits 0-3) and function (bits 4-15) of the Direct Control instruction, or the type and number of shifts of a shift instruction (bits 0-7 are ignored). Where the RIXS field invokes base-relative addressing, the mode and function information is loaded into general register X2 and added to the displacement field (RIXS=0001) to establish the final configuration.

3-21. Figure 3-8 shows the data flow within the CPU during the instruction preparation phases, and table 3-5 supports the figure in explanation of the control of data flow. To clock-step through the preparation phases, perform procedure in table 3-4.

Table 3-4. Instruction Preparation - Clock-step Procedure

Step	Procedure	Expected Results	Figure Reg. 3-12
1	Perform Data fetch Procedure A of table 3-2. Place the address of the desired instruction in the S-register and bring the instruction into D.	The D-register contains the instruction to be stepped through the preparation phases.	
2	Set CLOCK switch to STOP	Logic level KSTOP is high. KSTOP.IDLE.KIDLE.CLB = STEP. STEP.IDLE.CLA = CLOCK = CLEN/M. CLB = CLEN.CLA = CK,CL and RUN lamp is on (clock latch is off).	Sh 2 Sh 5 Sh 7
3	Set COMPUTE switch to RUN	Logic level KRUN and NIDLE are high. NSTEP/M.CLB = STEP/E = NSTEP.CLA = NCLOCK. Run lamp is off (clock latch is on). NIDLE.PRE0 = PRE0.	Sh 2 Sh 7 Sh 17
Note: Steps 1-3 are preparatory to clocking through the prep phases of the instruction. Step 4 starts the sequence.			
4	Momentarily set CLOCK switch to STEP	Logic level KSCLK is high. KSCLK.KRUN.CLB = STEP. This is the first condition for lifting the clock latch. When CLOCK switch is returned to STOP, KSTOP is high. STEP.KSTOP.CLB = CLOCKS.CLA = CLOCK, CLOCK = CLEN/M.CLB = CLEN.CLA = CK,CL. The clock latch is removed on each return to KSTOP allowing one CL, CK, each STEP. Each clock executes a phase in the instruction preparation phase sequence, 1-7. The PREP lamp is on during all prep phases if SELECT switch is set to PH. (KSEL/PH.NPREP-1 = KDIS00 = KLD00).	Sh 2 Sh 5 Sh 2 Sh 5 Sh 7 Sh 17, 18 Sh 11

3-22 SOURCE OF NEXT INSTRUCTION ADDRESS

Prior to the instruction preparation sequence, the address of the instruction must be brought into the memory address latches, the LA-register, in order to fetch the instruction from memory into the D-register. In the NORMAL RUN mode of operation, the next instruction address is selected at the end of the execution phase (ENDE) of the previous instruction. Depending on the type of previous instruction ending, the new address can come from the H-register, the P-register or the S-register.

3-23. Figure 3-7 shows the address data flow to the LA-register. For instructions having an ENDE/3 type of ending, BOC, the address of the next instruction is the effective address previously placed in the LA-register and H-register at the end-preparation phase of the instruction. This effective address (E.A.) is used at ENDP of FABR/1 (ENDE/3) and execution PH2 of FABR/X (also ENDE/3) if the branch is to the effective address.

3-24. For instructions having an ENDE/2 type ending, FUSFT, FAMANY, and after an I/O service call or interrupt sequence, the next instruction address is from the P-register. The I/O or interrupt sequence, interrupts the program at ENDE.IRQ to perform the sequence. Note (figure 3-7) that ENDE-1.IRQ will inhibit S/JK15/1 which is used for an H+1 or S+1 to update P when entering the PREP phase of the next instruction. When the program is interrupted, the P-register is not updated and, at the conclusion of the IRQ, the return to the program is to the address which was in P at the time of the interrupt.

3-25. During the execution of FAMANY, the effective address is incremented in the S-register and therefore the S-register does not contain the next address and it is necessary to return to the P-register. During FUSFT, the S-register is used as a shift-register and cannot contain the next instruction address, therefore it is necessary to return to the P-register for the next address. For the remaining instructions, those having an ENDE/1 type ending, the next instruction address is from the S-register. This is for the major portion of the instructions, including PH2 of FABR/X if it falls through.

3-26. In the step-clock mode of operation, the ENDE conditions have no significance, for the instruction address has been placed in the LA- and S-registers by the DATA toggle switches on the PCP. The preparation sequence (table 3-5) therefore, begins with PRE0 (idle state) instead of PRE1 as in the NORMAL RUN mode. In addition, it should be noted that the hex-display for PH in PRE1, table 3-5, does not reveal the address control for H plus 1 (the updating of P using the incremented E.A.) 8089. This display is present only at an ENDE/3 type of previous instruction ending. The S+1 indication in column J for PRE1 would also contain an H+1 or S(P)+1 for updating the P-register, depending on the type of ending.

NOTES:

1. ENDE/3 = ENDP OF FABR/1 AND EXECUTION PH2 OF FABR/X IF (X1) PLUS 1 ≠ 0.
2. ENDE/2 = EXECUTION PH4/PH5 OF FUSFT = LAST ITERATION OF EXECU PH3 OF FAMILY = IDPH16 OF I/O SERVICE CALL
3. ENDE/1 = EXECU PH2 OF FABR/X IF (X1) PLUS 1 = 0. ALL OTHERS EXCLUDING ENDE/2 AND ENDE/3.
4. ENDE/1 + ENDE/2 + ENDE/3 = ENDE (14)
5. ENDE = S/PREP/1. NTRAP. N(S/IO/L). N(S/INT/1). N(S/WAIT) = S/PREP/2 = PREP (17). INITIATES PREP PHASE.
6. ENDE, NIRQ = S/MC1 (15)
INITIATES A MEMORY CYCLE FOR FIRST PREP PHASE.
7. ENDE = RAX01 - S/RA4 = ADD. OF P-REGISTER K FOR ENDE OF ALL INSTRUCTIONS.
8. ALL NUMBERS IN PARENTHESIS ARE THE SHEET NOB. OF REFERENCE LOGIC DIAGRAM FIGURE 2 - .

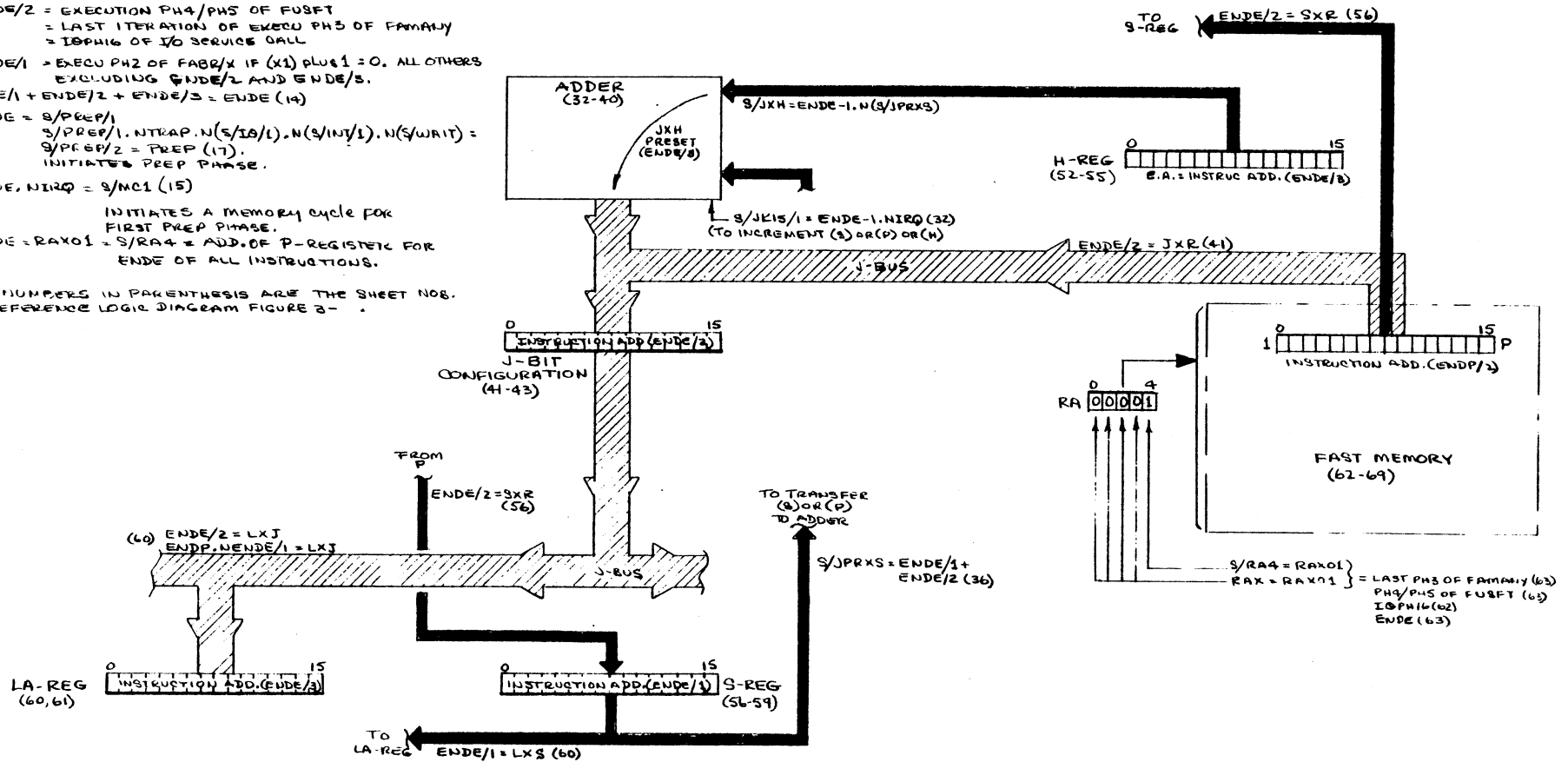


Figure 3-7. Source of next Instruction Address at End of Instruction Execution (ENDE)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12
	PH	O	LA	S	D	J	H	P	SW1		
RIXS= 0 thru 7	8000 (PRE0)	0001* (RA4)	Instruction Address	Instruction Address	Instruction	Instruction Address	-	Instruction Address	0	(Refer to table 3-4 for control logic and figure reference) *A previous execution phase or ENDE brings up S/RA4. RA4 = address of P-register. This remains until PRE3.	Sh 63,65
1st	8011 (PRE1)	0001* (RA4)		↓		S + 1	-		0	Adder control S/JXSP1 preset in PRE0. PRE0=S/JXSP1. S/JXSP1=S/JPRXS and S/JK15. (PRE1.KADDRINCR) presets SXJ & PRE1 also equals S/JPRXS for an S to J transfer in PRE2. HXR is preset by PRE1. JPRXS for a P to H transfer in PRE2. PRE1.NKDEPOSIT=S/RW. This brings RWRITE high in PRE2 for writing into P in PRE3.	Sh 36 Sh 36,32 Sh 56,36 Sh 53 Sh 68
2nd	8010 (PRE2)	XX21 (RW, RA4)		↓		Next Instruction Address		Next Instruction Address (P)	0	The contents of P (this instruction address) is transferred to H. The next instruction address in S is transferred to J. RW = RWRITE. Byte 0 of D-register (operation and RIXS fields) is in the O-register for decoding, OXD. The adder is preset to perform a JXD transfer (PRE2.NREL=S/JPRXNHD and PRE2.NREL=S/JPRXHD) in PRE3. PRE2 = S/SW1. PRE2.NFUCPY presets RAZ address of XI for PRE3.	Sh 19-22 Sh 37,39 Sh 28 Sh 66
<p>NOTE: A TRAP condition (MUL, DIV instruc without extended arithmetic option, memory protect violation or the MPE interrupt indication) will set a WAIT at ENDP.</p>											
3rd RIXS=0 and FUCPY	8040 (PRE3)	X004*** (RA2)	Instruction Address	Instruction Address	Instruction	Displ Fld and zeros in byte 0= Effective address (E.A.) or bits 8-15 of FCPY	Instruction Address	Next Instruction Address	1	RWRITE.CK = Contents of J (next instruction address) is loaded (35 nsec prior to JXD) into P, RA4 (PRE2) = address of P. The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 are zero. **For FUCPY, the 2nd MSD in the hex-display for 0 can be any value from 0 to E, excluding 3, 7, and B. ***For FUCPY, the LSD in the hex-display for 0 can be any value from 2 through 7 depending on the address of the source register. (PRE2.FUCPY).DI3,DI4,DI5=S/RA2,S/RA3,S/RA4, respectively). For FUCPY, ENDP.FUCPY presets HXR for source register transfer to the H-register in PH1. Excluding FUCPY, ENDP-1.NFUCPY presets HXJ for effective address transfer to the H-register in PH1. For all instructions, ENDP.NENDE/1 presets LXJ for effective address transfer to the LA-register in PH1. With LXJ high, a memory cycle is initiated (S/MC1) for PH1 for all memory reference instructions (incl B,BOC, excl SHIFT). All other instructions (FAOP1) inhibit S/MC1. Excluding FUCPY, and FAMANY, the address of the accumulator (RA2, RA3, RA4) is preset by ENDP.NENDE.N(FUBOC+FUCPY).NFAMANY=RAX07=RAX06/1 and S/RA4. RAX06/1=S/RA2 and S/RA3 and RAX. For FAMANY, the address of the starting reg is preset by (ENDP.FAMANY).LM13,LM14,LM15=S/RA2,S/RA3,S/RA4. Also ENDP.FAMANY presets SXJ transfer to increment E.A. of associated instruction.	Sh 69 Sh 65 Sh 37,39-42 Sh 65,66 Sh 53 Sh 52 Sh 60 Sh 15 Sh 22 Sh 63-66 Sh 65-66 Sh 56

Table 3-5. Instruction Prep Phase Sequence Control (sheet 1 of 9)

S Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12	
	PH	0	LA	S	D	J	H	P	SW1			
S=1	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on sheet 1 of this table, with one exception. The second MSD of the Hex display for 0 will indicate the value of the associated RIXS field)									1	<p>RWRITE.CK=Contents of J (Next Instruc Add.) is loaded(35 nsec prior to JXD) into P, RA4 (PRE2) = address of P. The adder performs a JXD on byte 1 only. PRE inhibits the transfer of byte 0, therefore, J00-J07 are zero.</p> <p>((PRE3+4).NENDP) presets DXJ for transfer of J to D in PRE4.</p> <p>((PRE3+4).NENDP) presets HXR to transfer the contents of X2 to H in PRE4.</p> <p>PRE2.NFUCPY.NREL.07-S/RA4 & S/RA2=PRE2.NFUCPY=X2.</p> <p>((PRE3+4).NENDP).N/(S/PRE5)=S/JXHPD presets an ADD operation for PRE4. ((PRE3+4).NENDP) presets RA2 for PRE4.</p> <p>Contents of X2 are transferred to the adder (H-reg) and the contents of D (partial E.A.) are in the adder. An ADD operation is performed to produce the final E.A. JXHPD=JPRXHND, JPRXNHD, JGXHD.</p> <p>ENDPREP (ENDP-1)</p> <p>ENDP-1.NFUCPY presets HXJ for transfer of E.A. to H in PH1.</p> <p>ENDP.NENDE/1 presets LXJ for transfer of E.A. to LA in PH1.</p> <p>LXJ presets a memory cycle (S/MC1) for PH1 for all memory reference instruc. (incl B,BOC,excl shift). All other instructions (FAOP1) inhibit S/MC1.</p> <p>For all instructions except FAMANY, the address of the accumulator (RA2, RA3, RA4) is preset for PH1 by ENDP.NENDE.N(FUBOC+FUCPY).</p> <p>NFAMANY=RAX07=RAX06/1 and S/RA4. RAX06/1=S/RA2, S/RA3, RAX.</p>	<p>Sh 69 Sh 65 Sh 37,39-42</p> <p>Sh 45 Sh 53 Sh 65 Sh 35 Sh 63,66</p> <p>Sh 37,39-42</p> <p>Sh 52 Sh 60 Sh 15 Sh 22</p> <p>Sh 63-66</p>
	d	8048 (PRE3)	X105 (RA2&RA4)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros	Instruction Address	Next Instruction Address			
	80E0 (PRE4)	X104 (RA2)	↓	↓	Contents of J=subtotal E.A.	H+D=subtotal E.A. plus (X2)=E.A.	Contents of X2	↓				
S=2	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) of sheet 1 of this table, with one exception. The second MSD of the Hex-display for 0 will indicate the value of the associated RIXS field)									1	<p>RWRITE.CK=Contents of J (Next instruc add) is loaded(35 nsec prior to JXD) into P, RA4 (PRE2) = address of P. The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 are zero.</p> <p>((PRE3+4).NENDP) presets DXJ for transfer of J to D in PRE4.</p> <p>((PRE3+4).NENDP) presets HXR to transfer the contents of X1 to H in PRE4.</p> <p>PRE2.NFUCPY=S/RA2 (add. of X1) for PRE3.</p> <p>((PRE3+4).NENDP).N/(S/PRE5)=S/JXHPD presets an ADD operation for PRE4. ((PRE3+4).NENDP) presets RA2 for PRE4.</p> <p>Contents of X1 are transferred to the adder (H-reg.) and the contents of D (partial E.A.) is in the adder. An ADD operation is performed to produce the final E.A. JXHPD=JPRXHND, JPRXNHD, JGXHD.</p> <p>For ENDP conditions see RIXS=1</p>	<p>Sh 69 Sh 65 Sh 37,39-42</p> <p>Sh 45 Sh 66 Sh 35 Sh 63,66</p> <p>Sh 37,39-42</p>
	h	8048 (PRE3)	X204 (RA2)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros in byte 0	Instruction Address	Next Instruction Address			
	80E0 (PRE4)	X204 (RA2)	↓	↓	Contents of J=subtotal E.A.	H+D subtotal E.A. plus (X1)=E.A.	Contents of X1	↓	ENDP			

Table 3-5. Instruction Prep Phase Sequence Control (sheet 2 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12	
	PH	0	LA	S	D	J	H	P	SW1			
RIX=3	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on sheet 1 of this table, with one exception. The second MSD of the Hex-display for 0 will indicate the value of the associated RIXS field)											
3rd	8048 (PRE3)	X305 (RA2, RA4)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros in byte 0	Instruction Address	Next Instruction Address	1	<p>RWRITE.CK = Contents of J (next instruc add) is loaded(35 nsec prior to JXD)into P, RA4 (PRE2) = add. of P. The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore J00-J07 are zero.</p> <p>((PRE3+4).NENDP) presets DXJ for transfer of J to D in PRE4</p> <p>((PRE3+4).NENDP) presets HXR to transfer the contents of X2 to H in PRE4.</p> <p>PRE2.NFUCPY.NREL.07 = S/RA4 and PRE2.NFUCPY = S/RA2.</p> <p>RA2, RA4 = add of X2.</p> <p>((PRE3+4).NENDP).N(S/PRE5)=S/JXHPD presets an ADD operation for PRE4. ((PRE3+4).NENDP) presets RA2 for PRE4.</p> <p>((PRE3+4).NENDP). . . =S/SW1</p>	<p>Sh 69 Sh 65 Sh 37,39-42</p> <p>Sh 45 Sh 53 Sh 65 Sh 66 Sh 35 Sh 63,66 Sh 28</p>	
4th	80E0 (PRE4)	X304 (RA2)			Contents of J=subtotal E.A.	H+D= subtotal E.A. plus (X2)	Contents of X2	1	<p>Contents of X2 is transferred to the adder (H-register) and the contents of D (subtotal E.A.) is in the adder. An ADD operation is performed to produce another subtotal E.A.</p> <p>JXHPD = JPRXHND, JPRXNHD, JGXHD.</p> <p>((PRE3+4).NENDP) presets DXJ for transfer of J to D in PRE7</p> <p>((PRE3+4).NENDP) presets HXR to transfer the contents of X1 to H in PRE7. ((PRE3+4).NENDP) presets RA2 for PRE7</p> <p>((PRE3+4).NENDP).N(S/PRE5)=S/JXHPD presets an ADD op for PRE7</p>			<p>Sh 38,39-42 Sh 45 Sh 53 Sh 63,66 Sh 35</p>
5th	80E0 (PRE7)	X304 (RA2)	↓	↓	Contents of J=subtotal E.A.	H+D= subtotal E.A. plus (X1)=E.A.	Contents of X1	ENDP	<p>Contents of X1 is transferred to the adder (H-register) and the contents of D (subtotal E.A.) is in the adder. An ADD operation is performed to produce the final E.A.</p> <p>JXHPD = JPRXHND, JPRXNHD, JGXHD.</p> <p>For ENDP conditions see RIXS=1</p>			<p>Sh 37,39-42</p>

Table 3-5. Instruction Prep Phase Sequence Control (sheet 3 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12
	PH	0	LA	S	D	J	H	P	SW1		
RIXS=4	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on Sheet 1 of this table, with one exception. The second MSD of the Hex-display for 0 will indicate the value of the associated RIXS field)										
3rd	8048 (PRE3)	X404 (RA2)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros in byte 0 (Indirect Address)	Instruction Address	Next Instruction Address	1	<p>RWRITE.CK = contents of J (next instruc add) is loaded (35 nsec prior to JXD) into P; RA4 (PRE2) = address of P.</p> <p>The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 = zero.</p> <p>(S/JXHM1)+(S/PRE5)=S/JGXHND (JXH). This defines PRE5, however, H to J in PRE5 is not significant for this RIXS.</p> <p>((PRE3+4).NENDP) presets DXJ for a J to D transfer (Not Significant)</p> <p>((PRE3+4).NENDP) presets HXR for an X2 or X1 transfer to H (not significant). ((PRE3+4).NENDP).N(S/PRE5) presets S/JXHPD for an ADD op in PRE5 (not significant).</p> <p>((PRE3+4).NENDP) presets RA2 for PRE5 (not significant).</p> <p>ENDE/2+(S/PRE5) presets LXJ for transfer of J to LA in PRE5.</p> <p>LXJ presets a memory cycle (S/MC1) for PRE5.</p> <p>((PRE3+4).NENDP).JGXHD.O5+(O5.NO7)=S/PRE5</p>	<p>Sh 69</p> <p>Sh 65</p> <p>Sh 37,39-42</p> <p>Sh 38</p> <p>Sh 45</p> <p>Sh 53</p> <p>Sh 35</p> <p>Sh 63,66</p> <p>Sh 60</p> <p>Sh 15</p> <p>Sh 18</p>
4th	8002 (PRE5)	X484 (MC1, RA2)	Indirect Address		Indirect Address	Not signif	Contents of X1			<p>Indirect address on J transferred to LA-register and memory cycle started (MC1). (PRE5+6).N06 presets HX to clear H-register.</p> <p>(PRE5+6).MC1 presets S/JGXHND (JXH). This defines PRE6 and presets an H to J transfer in PRE6.</p>	<p>Sh 52</p> <p>Sh 38</p>
5th	8002 (PRE6)	X404 (RA2)			Contents of Indirect Address	Not signif	All zeros			<p>DXMRC.MRC00-MRC15 = D00-D15, memory read data (contents of indirect address) in the D-register. (PRE5+6).NMC1=S/JXHPD presets an ADD operation for PRE7.</p>	<p>Sh 35</p>
6th	80E0 (PRE7)	X404 (RA2)				H+D= Contents of D plus zeros=E.A.	All zeros		ENDP	<p>Contents of H (all zeros) and contents of D are added to produce the effective address.</p> <p>JXHPD = JPRXHND, JPRXNHD, JGXHD</p>	<p>Sh 37,39-42</p>
										For ENDP conditions see RIXS = 1	

Table 3-5. Instruction Prep Phase Sequence Control (sheet 4 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)										Comments	Figure Ref. 3-12
	PH	0	LA	S	D	J	H	P	SW1			
RIXS=5	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on Sheet 1 of this table, with one exception. The second MSD of the Hex-display for 0 will indicate the value of the associated RIXS field)											
3rd	8048 (PRE3)	X505 (RA2, RA4)	Instruction Address ↓	Next Instruction Address ↓	Instruction	Displ Fld and zeros in Byte 0	Instruction Address	Next Instruction Address ↓	1	RWRITE.CK = contents of J (next instruc add) is loaded (35 nsec prior to JXD) into P; RA4 (PRE2) = address of P. The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 are zero. ((PRE3+4).NENDP) presets DXJ for a J to D transfer in PRE4. ((PRE3+4).NENDP) presets HXR for an X2 trans to H in PRE4. PRE2.NFUCPY, NREL.07 presets RA4 and PRE2.NFUCPY preset RA2 for address of X2. ((PRE3+4).NENDP).N(S/PRE5) = S/JXHPD presets an ADD operation for PRE4. ((PRE3+4).NENDP). . . =S/SW1 PRE3+4.NENDP presets RA2 for PRE4 (not significant)	Sh 69 Sh 65 Sh 37,39-42	
4th	80E0 (PRE4)	X504 (RA2)	↓	↓	Contents of J (Displ Fld)	H+D= Contents of D plus X2= indirect add	Contents of X2	↓	1	Contents of X2 are transferred to adder (H-register). Contents of D (disp field) and H are added to produce an indirect add. (S/JXHMI)+(S/PRE5) presets JGXHND (JXH). This defines PRE5 and presets an H to J transfer (not significant) in PRE5. ((PRE3+4).NENDP) presets DXJ to transfer J to D in PRE5 (not significant). ((PRE3+4).NENDP) presets HXR to transfer X1 to H in PRE5 (not significant). ((PRE3+4).NENDP).N(S/PRE5) = S/JXHPD presets an ADD op in PRE5 (not significant). ((PRE3+4).NENDP).JGXHD.O5+(O5, NO7) = S/PRE5 ENDE/2+(S/PRE5) presets LXJ to transfer J to LA in PRE5. LXJ presets a memory cycle (S/MC1) for PRE5. ((PRE3+4).NENDP) presets RA2 for PRE5 (not significant).	Sh 38 Sh 45 Sh 53 Sh 35 Sh 18 Sh 60 Sh 15 Sh 63,66	
5th	8002 (PRE5)	X584 (MC1, RA2)	Indirect Address from J ↓	↓	Contents of J (indirect address)	Not signif.	Contents of X1	↓		Indirect address on J transferred to LA-register, memory cycle is started (MC1). (PRE5+6).NO6 presets HX to clear H (all zeros) in PRE6. (PRE5+6).MC1 = S/JGXHND (JXH). This defines PRE6 and presets an H to J transfer in PRE6.	Sh 52 Sh 38	
6th	8002 (PRE6)	X504	↓	↓	Contents of indirect address ↓	Not signif.	All zeros	↓		DXMRC.MRC00-MRC15 = D00-D15, memory read data (contents of indirect address) is in the D-register. (PRE5+6).NMC1 = S/JXHPD presets an ADD operation for PRE7.	Sh 44,46-49 Sh 35	
7th	80E0 (PRE7)	X504	↓	↓	↓	H+D= Contents of D plus zeros=E.A.	All zeros	↓		Contents of H (all zeros) and contents of D are added together to produce the E.A. JXHPD = JPRXHND, JPRXNHD, JGXHD	Sh 37,39-42	
									ENDP	For ENDP conditions see RIXS = 1		

Table 3-5. Instruction Prep Phase Sequence Control (sheet 5 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-1
	PH	0	LA	S	D	J	H	P	SW1		
RIXS=6	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on Sheet 1 of this table, with one exception. The second MSD of the hex-display for 0 will indicate the value of the associated RIXS field)										
3rd	8048 (PRE3)	X604 (RA2)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros in byte 0 (indirect address)	Instruction Address	Next Instruction Address	1	<p>RWRITE.CK = contents of J (next instruc add) is loaded(35 nsec prior to JXD) into P; RA4 (PRE2) - address of P.</p> <p>The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 = zero.</p> <p>$(S/JXHM1)+(S/PRE5) = S/JGXHND$ (JXH). This defines PRE5, however, H to J in PRE5 is not significant for this RIXS.</p> <p>$((PRE3+4).NENDP)$ presets DXJ for a J to D transfer to PRE5 (not significant)</p> <p>$((PRE3+4).NENDP)$ presets HXR for an X1 transfer to H in PRE5.</p> <p>$((PRE3+4).NENDP).N(S/PRE5)$ presets S/JXHPD for an ADD op in PRE5 (not significant).</p> <p>$((PRE3+4).NENDP)$ presets RA2 for PRE5 (not significant)</p> <p>$ENDE/2+(S/PRE5)$ presets LXJ for transfer of J to LA in PRE5.</p> <p>LXJ presets a memory cycle (S/MC1) for PRE5.</p> <p>$((PRE3+4).NENDP).JGXHD.O5+(O5.NO7) = S/PRE5$</p>	Sh 69 Sh 65 Sh 37,39-4 Sh 38
4th	8002 (PRE5)	X684 (MC1, RA2)	Indirect Address		Indirect Address	(Not signif)	Contents of X1			<p>Indirect address on J transferred to LA-register and memory cycle is started (MC1). $(PRE5+6).MC1$ presets S/JGXHND (JXH).</p> <p>This defines PRE6 and presets an H to J transfer in PRE 6 (not significant).</p>	Sh 38
5th	8002 (PRE6)	X604 (RA2)			Contents of indirect address	(Not signif)	Contents of X1			<p>$DXMRC.MRC00-MRC15 = D00-D15$, memory read data (contents of indirect address) in the D-register. $(PRE5+6).NMCI$ presets an ADD operation for PRE7.</p>	Sh 35
6th	80E0 (PRE7)	X604 (RA2)				H+D= Contents of D plus (X1) =Eff. Add.	Contents of X1		ENDP	<p>Contents of X1 (H-register) is in the adder and also the contents of D. An ADD operation is performed to produce the E.A.</p> <p>$JXHPD = JPRXHND, JPRXNHD, JGXHD$</p> <p>For ENDP conditions see RIXS = 1</p>	Sh 37,39-4

Table 3-5. Instruction Prep Phase Sequence Control (sheet 6 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12
	PH	0	LA	S	D	J	H	P	SW1		
RIXS=7	(Refer to PRE0 through PRE2 of RIXS configurations (0-7) on sheet 1 of this table, with one exception. The second MSD of the hex-display for 0 will indicate the value of the associated RIXS field)										
3rd	8048 (PRE3)	X705 (RA2, RA4)	Instruction Address	Next Instruction Address	Instruction	Displ Fld and zeros in byte 0	Instruction Address	Next Instruction Address	1	<p>RWRITE.CK = contents of J (next instruc add) is loaded(35 nsec prior to JXD)into P; RA4 (PRE2) = address of P.</p> <p>The adder performs a JXD on byte 1 only. PRE3 inhibits the transfer of byte 0, therefore, J00-J07 are zero.</p> <p>((PRE3+4).NENDP) presets DXJ for a J to D transfer in PRE4.</p> <p>((PRE3+4).NENDP) presets HXR for an X2 transfer to H in PRE4.</p> <p>PRE2.NFUCPY.NREL.07 preset RA4 and PRE2.NFUCPY preset RA2 for address of X2. ((PRE3+4).NENDP).N(S/PRE5)=X/JXHPD presets an ADD operation for PRE4. ((PRE3+4).NENDP). . =S/SW1</p> <p>PRE3+4.NENDP presets RA2 for PRE4 (not significant)</p>	Sh 69 Sh 65 Sh 37,39-42
4th	80E0 (PRE4)	X704 (RA2)			Contents of J (Displ Fld)	H+D Contents of D plus (X2) = indirect address	Contents of X2		1	<p>Contents of X2 is transferred to adder (H-register). Contents of D (displ fld) and H are added to produce an indirect address.</p> <p>(S/JXHM1)+(S/PRE5) presets JGXHND (JXH). This defines PRE5 and presets an H to J trans (not significant) in PRE5.</p> <p>((PRE3+4).NENDP) presets DXJ to trans J to D in PRE5 (not significant).</p> <p>((PRE3+4).NENDP) presets HXR to trans X1 to H in PRE5.</p> <p>((PRE3+4).NENDP.N(S/PRE5) = S/JXHPD presets an ADD op in PRE5 (not significant).</p> <p>ENDE/2+(S/PRE5) presets LXJ to trans J to LA in PRE5.</p> <p>LXJ presets a memory cycle (S/MC1) for PRE5.</p> <p>((PRE3+4).NENDP) presets RA2 for PRE5.</p> <p>((PRE3+4).NENDP).JGXHD.O5+(O5.NO7) = S/PRE5</p>	Sh 38 Sh 45 Sh 53 Sh 35 Sh 60 Sh 15 Sh 63,66 Sh 18
5th	8002 (PRE5)	X584 (MC1, RA2)	Indirect Address		Indirect Address	(Not signif)	Contents of X1			<p>Indirect address on J transferred to LA-register, memory cycle is started (MC1). ((PRE5+6).MC1) = S/JGXHND (JXH). This defines PRE6 and presets an H to J transfer in PRE6.</p>	Sh 38
6th	8002 (PRE6)	X704 (RA2)			Contents of Indirect Address	(Not signif)	Contents of X1			<p>DXMRC.MRC00-MRC15 = D00-D15, memory read data (contents of indirect address) is in the D-register.</p> <p>(PRE5+6).NMC1 = S/JXHPD presets an ADD operation for PRE7.</p>	Sh 44,46-49 Sh 35
7th	80E0 (PRE7)	X704 (RA2)				H+D= Contents of D plus (X1)	Contents of X1			<p>Contents of H (contents of X1) and contents of D are added to produce the effective address.</p> <p>JXHPD = JPRXHND, JPRXNHD, JGXHD</p>	Sh 37,39-42
								ENDP		For ENDP conditions see RIXS = 1	

Table 3-5. Instruction Prep Phase Sequence Control (sheet 7 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12
	PH	0	LA	S	D	J	H	P	SW1		
RIXS= 8-F	8000 (PRE0)	0001* (RA4)	Instruction Address	Instruction Address	Instruction	Instruction Address	-	Instruction Address	0	(Refer to table 3-4 for control logic and figure ref.) *A previous execution phase or ENDE brings up S/RA4. RA4 is address of P-reg. RA4 remains until PRE3. PRE0=S/JXSP1 presets an S plus 1 to J in PRE1. S/JXSP1=S/JPRXS for the S to J trans and S/JK15 for the plus 1.	Sh 63,65 Sh 36 Sh 32
1st	8011 (PRE1)	0001* (RA4)	↓	↓	↓	S+1	-	↓		PRE1.JPRXS presets HXR to trans P to H in PRE2. PRE1 = S/JPRXS presets an S to J trans in PRE2. PRE1.NKDEPOSIT = S/RW presets RWRITE for PRE2. PRE1.KADDRINCR presets SXJ for S+1 (J) to S transfer in PRE2.	Sh 53 Sh 36 Sh 68 Sh 56
2nd	8010 (PRE2)	XX21 (RW, RA4)	↓	Next Instruction Address	↓	Next Instruction Address	Instruction Address (P)	↓		The contents of P (this instruc add) is transferred to H. The next instruc add. in S is transferred to J. Byte 0 of the D-register (operation and RIXS fields) is in the 0-register. PRE1.NMC1.CL-10=OXD, trans byte 0 in D to 0-register for decoding. REL.PRE2.N(S/JXHPD/1)=S/JXHPD presets for an ADD op in PRE4. PRE2.NFUCPY presets RA2 for PRE4. RWRITE is high for writing the next instruc add. on J-bus into P-reg. PRE2 = S/Sw1.	Sh 19-22 Sh 35 Sh 66 Sh 69 Sh 28
3rd	80E0 (PRE4)	XX04 (RA2)	↓	↓	↓	H+D= Contents of D plus (H)= Self-relative add. (This equals E.A. for RIXS= 100S, only)	↓	Next Instruction Address	1	RWRITE.CK = contents of J (next instruc add.) is trans to P-reg. 35 nanoseconds prior to the ADD op. The ADD op is carried out as follows: A straight forward H plus D is performed on byte 1, JXHPD=JPRXHND/1, JPRXNHD/1, JGXHD/1. If the sign (S) bit is positive (0) an H to J is performed on byte 0, JXH=JPRXHND/0, JPRXHND/0, and the carry out of byte 1 (JK07) is also transferred. If the sign (S) bit is negative (1) an H minus 1 is performed on byte 0, JXHMI=JPRXNHD/0, JPRXNHD/0, JGXHD/0 and JGXNHD/0, and the carry out of byte 1 (JK07) is also incl. The control logic for byte 0 is as follows: REL.07.PREP-1.SW1=DSPW and REL.N07.PREP-1.SW1=DSPZ. DSPZ.NJPRXHD=JPRXHND/0 and JPRXHND/1.NDSPW=JPRXHND/0. JGXHD/1.NDSPZ=JGXHD/0 and JPRXNHD/1.NDSPZ.NPRE3=JPRXNHD/0 and DSPW = JGXHND/0, JPRXNHD/0. ((PRE3+4).NENDP)=RAX04=S/RA2 (Signif for RIXS=101S, 111S) ((PRE3+4).NENDP) presets DXJ for trans of J to D (101S, only) ((PRE3+4).NENDP) presets HXR to trans X1 to H (101S, 111S) ((PRE3+4).NENDP).N(S/PRE5)=S/JXHPD presets ADD op for PRE7 (signif for 101S, only). ((PRE3+4).NENDP.JGXHD.O5+(O5.NO7) = S/PRE5 ENDE/2+(S/PRE5) presets LXJ for trans of indirect add. on J to LA-reg. in PRE5. LXJ presets a memory cycle (S/MC1) for PRE5 (signif for 110S and 111S). (S/JXHM1)+(S/PRE5) presets JGXHND. This defines PRE5 (signif for 110S, 111S). For ENDP see RIXS=1 plus ENDP for BOC: ENDP-2.FABR/X=SXR to trans X1 to S in PH1. ENDP-2.FABR/X=S/JXSP1 to increment S in PH1 and ENDP-2.FABR/X=RAX01=S/RA4 presets address of P for transfer of P to S for PH1/PH2-((X1)+1=zero condition (ENDE/1)). ENDP.FARB/1=ENDE/3=S/MC1.	Sh 36-42 Sh 39-42 Sh 36-42 Sh 39 Sh 63,66 Sh 18 Sh 60 Sh 15 Sh 38 Sh 56,36 Sh 63,65 Sh 14,15

Table 3-5. Instruction Prep Phase Sequence Control (sheet 8 of 9)

RIXS Field & Clock	DISPLAY (Ref. Figure 3-8)									Comments	Figure Ref. 3-12	
	PH	0	LA	S	D	J	H	P	SW1			
RIXS= 101S 4th	80E0 (PRE7)	XX04 (RA2)	(Refer to RIXS = 8-F for PRE0 through PRE4)								The self-relative address was transferred from J to D and the content of X1 was transferred to H. An ADD was performed to produce the E.A. JXHPD = JPRXHND, JPRXNHD, JGXHD. For ENDP see RIXS = 1.	Sh 36-42
			Instruction Address	Next Instruction Address	Contents of J=self-rela-tive add.	H+D= Contents of D plus X1= E.A.	Contents of X1	Next Instruction Address	ENDP			
RIXS= 110S & 111S 4th	8002 (PRE5)	XX84 (MC1, RA2)	(Refer to RIXS = 8-F for PRE0 through PRE4)								Indirect address on J is transferred to LA-register, memory cycle started (MC1). (PRE5+6).MC1 presets S/JGXHND. This defines PRE6. (PRE5+6).NO6 presets an HX to clear H-register in PRE6(110s, only) DXMRC.MRC00-MRC15 = D00-D15, memory read data (contents of indirect address) in D-register. (PRE5+ç).NMC1 = S/JXHPD presets an ADD operation for PRE7. Contents of H and contents of indirect address (D-register) are added to produce E.A. For ENDP see RIXS = 1.	Sh 38 Sh 52 Sh 44,46-49 Sh 35
5th	8002 (PRE6)	XX04 (RA2)	Self-rel add from J = Indirect add	Next Instruction	Indirect Address	Not signif.	Contents of X1	Next Instruction Address				
6th	80E0 (PRE7)	XX04 (RA2)	↓	↓	↓	Not signif	All zeros (110S) Contents of X1 (111s)	↓	ENDP			
						H+D= Contents of Indirect add plus zeros = E.A. (110S) H+D= Contents of Indirect add plus (X1) = E.A. (111S)	↓					

Table 3-5. Instruction Prep Phase Sequence Control (sheet 9 of 9)

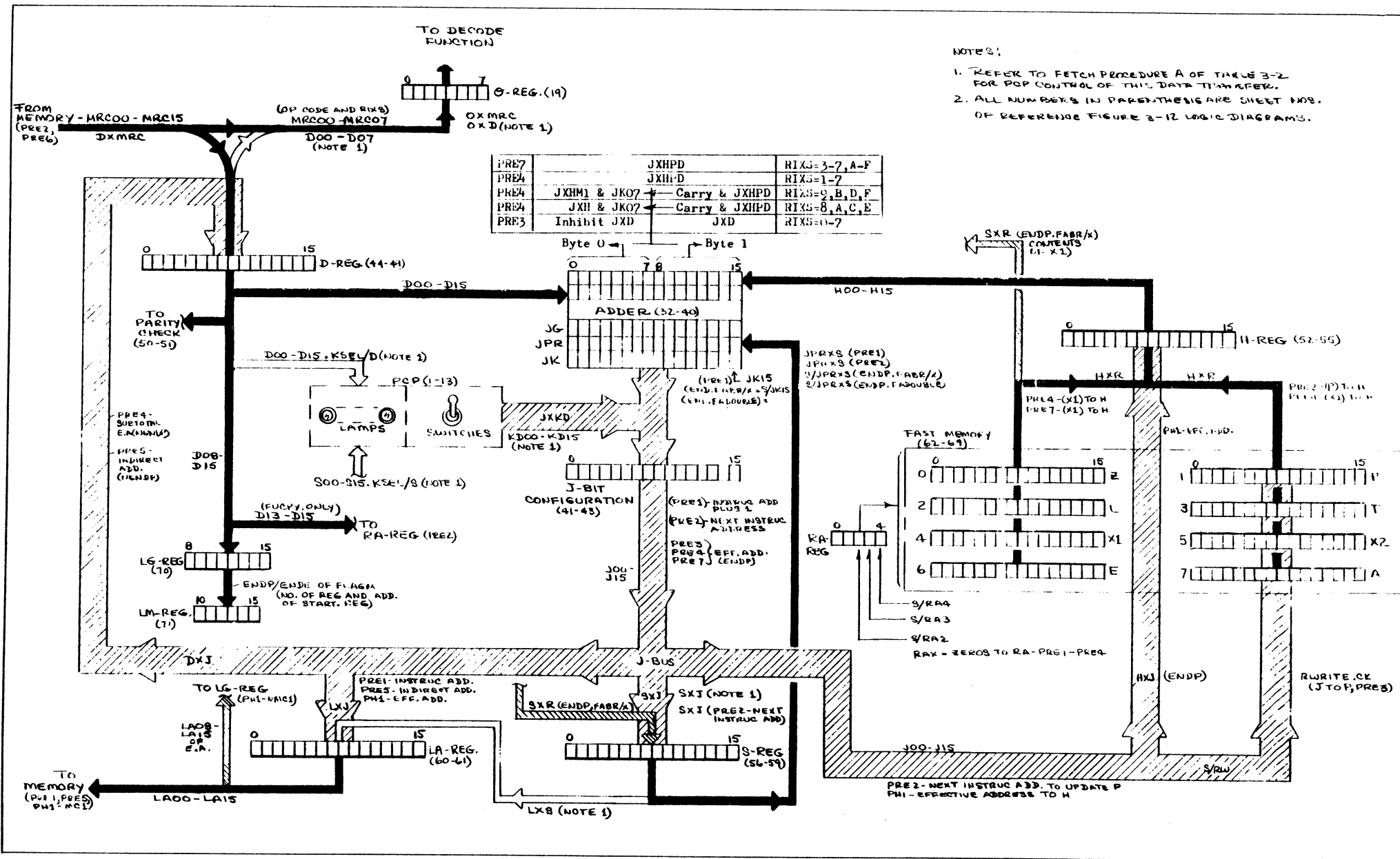


Figure 3-8. Instruction Prep Sequence - Data Flow During RIXS Computation

3-27 INSTRUCTION EXECUTION SEQUENCE

If there is no condition to abort an instruction (this is evident by a WAIT condition at end of preparation) the very next clock after PRE7 brings the instruction into the first phase of execution, PH1. Each clock thereafter brings the instruction to the next sequential phase (1 through 7) providing the control logic does not inhibit phase advance, in which case each clock reiterates the selected phase until the inhibit is lifted.

3-28. To simplify the explanation for the execution phases, the instructions are grouped as presented in the Instruction Preparation Sequence, paragraph 3-16; the groups are Direct Control, Memory Reference, FUCPY and SHIFT instructions.

3-29 DIRECT CONTROL INSTRUCTION

Figure 3-9 represents the data flow during direct control instruction execution (excluding those pertaining to IO). Table 3-6 supports this figure and can be considered an extension of table 3-5, Instruction Prep Phase Sequence Control.

NOTE: Similar sequences of instruction execution for memory reference, FUCPY and SHIFT instructions are to be added.

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-1
PH	O(RA)	D-Register	J-Bus	H-Register		
40XX (PH1)	RA=00111	-	Contents of (A) - Not Significant	1. E.A.=0000 0000 000n nnnn (RD)	Accumulator (A) address is preset at ENDP. The transfer JXR = PH1-2.FARWD.NJXKD has no significance, however, as the instruction is involved with registers other than (A) in PH1. The address of (n) for PH2 is preset by: S/RA0 = PH1.FARWDO/GR, RA0104XLG1215 = PH1.FARWDO/GR, S/RA1 and S/RA3 = RA0104XLG1215.LG12, LG14 S/RA2 = RAX04 = RA0104XLG1215.LG13 S/RA4 = RAX01 = RA0104XLG1215.LG15	Sh 41 66 64 66,6 66 65
	RA=00111	-	Contents of (A) - Not Significant	2. E.A.=0000 0000 010n nnnn (WD)	The transfer JXR=PH1-2.FARWD.NJXKD has no significance as the instruction is involved with registers other than (A) in PH1. The address of (n) for PH2 is preset by: S/RA0 = PH1.FARWDO/GR.LG11, RA0104XLG1215 = PH1.FARWDO/GR, S/RA1 and S/RA3 = RA0104XLG1215.LG12, LG14 S/RA2 = RAX04 = RA0104XLG1215.LG13 S/RA4 = RAX01 = RA0104XLG1215.LG15	41 Sh 66 64 66,6 66 65
	RA=00111	-	Contents of (A) for transfer to a general/I/O-register (n)	3. E.A.=0000 0000 000n nnnn (WD)	The contents of (A) is transferred to J-bus by JXR = S/RW/2 = PH1.FUWD/XX0X.FARWDO/GR.NLG09. S/RW/2 presets DXJ and JXD for a J → D → J transfer in PH2. S/RW/2 presets RW=RWRITE in PH2 for writing into (n) PH7. The address of (n) is preset by: S/RA0 = PH1.FARWDO/GR.LG11, S/RA1 and S/RA3 = RA0104XLG1215.LG12, LG14 S/RA2 = RAX04 = RA0104XLG1215.LG13 S/RA4 = RAX01 = RA0104XLG1215.LG15	Sh 41 45,3 68 66 66,6 66 65
	RA=00111	-	Configuration of PCP DATA toggle switches 0-15 for transfer to (A)	4. E.A.=0000 0000 1000 0000 (RD)	Logic levels from Data switches 0-15 are transferred to the J-bus by: JXKD = PH1.FURD.FARWDO.(LG08 + LCI). PH1.FARWDO/1XXX.NLG09 presets DXJ and PH1.FURDO/1XXX presets a JXD for a J → D → J transfer in PH7. PH1.FARWDO/1XXX presets a RW=RWRITE to enable writing into (A). PH1.FARWDO/1XXX.NLG09 = S/PH7	Sh 41 45 34 68 23
	RA=00111	-	Contents of (A) to alter status-bit configuration - byte 0 = zero and byte 1 = status-bit information	5. E.A.=0000 0000 1100 0000 (WD)	The contents of (A) are transferred to the J-bus by JXR=PH1.FARWD.NJXKD. PH1.FARWDO/1XXX.LG09.NLG11 presets STATUSXJ for gating J08, 10, 11, 14 and 15 to set appropriate status bits in PH2.	Sh 41 29 30,31

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 1 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12	
PH	O(RA)	D-Register	J-Bus	H-Register			
40XX (PH1- Contd)	RA=00111	-	Contents of (A) - Not Significant	6. E.A.=0000 0000 1100 0000 (RD)	The transfer JXR = PH-2.FARWD.NJXKD has no significance. The instructions are not involved with registers in PH1.	Sh 41	
				7. E.A.=0000 0000 1110 XX00 (RD)	PH1.FARWDO/1XXX.LG09 presets DXSTATUS for status bit → D in PH2.	45	
				8. E.A.=0000 0000 1111 XX00 (RD)	PH1.FURDO/1XXX = S/RW/2 - this presets JXD for a D → J transfer in PH2 along with RWRITE.	68, 34	
					PH1.FARWDO/111X.LG12 enables R/II or PH1.FARWDO/111X.LG13 enables R/EI in PH2.	29	
		RA-00111	-	-	9. E.A.=0000 0000 1000 xxxx (WD)	PH1.FARWDO/111X.LG11.LG12 enables S/II or PH1.FARWDO/111X.LG11.LG13 enables S/EI in PH2.	
						This instruction transfers the contents of (A) which is the memory protect information that singles out a specific page (256 locations) to protect in memory. E.A. bits H12-H15 are the address of the MP-register where this information is loaded. The transfer of (A) to the MP-register is controlled by MPXR.S/MPXR = PH1.	
						FARWDO/1XXX.FUWD.NLG09. This presets the transfer of (A) to a MP-register. PH1.FARWDO/1XXX, NLG09 = S/PH7/1	23
		RA-00111	-	-	10. E.A.=0000 0000 1101 0000 (WD)	This instruction is not involved with registers. PH1.FUWDO/1101.NLG12 = S/WAIT presets a WAIT condition in PH2.	26
		RA-00111	-	-	11. E.A.=0000 0000 1101 1000 (WD)	This instruction is not involved with registers. PH1.FUWDO/1101.LG12.NEXIT = S/EXIT presets EXIT in PH2.	26
		RA-00111	-	-	12. E.A.=0000 0000 1110 XX00 (WD)	These instructions are not involved with registers; they control status-bit F.F..PH1.FARWDO/111X.LG12 = R/II or PH1.FARWDO/111X.LG13 = R/EI in PH2.	29
				13. E.A.=0000 0000 1111 XX00 (WD)	PH1.FARWDO/111X.LG11.LG12 = S/II or PH1.FARWDO/111X.LG11.LG13 = S/EI in PH2.		
	RA-00111	-	Contents of (A) (Determines which interrupt levels of a specific group will be operated upon)	14. E.A.=0001 0XXX 0000 0000 (WD)	Contents of (A) is transferred to J by JXR = PH1=-. FARWD.NJXKD.	41	
					FUWD1.PH1 = DXJ presets a J → D transfer in PH2.	45	
					FUWD1.NRAUC/2.(PH1+2+4+5) = NINTCLEN/M1 inhibits the 1st interrupt clock (CLGAT) in PH2.	6 1	
	RA-00111	-	Contents of (A)	16. E.A.=Mode & Device Address (DIORWD)	The contents of (A) were previously loaded for data output to a device at the DIO interface for a DIORWD = (WD). JXR=PH1-2.FARWD.NJXKD transferred (A) → J.	41	
					FAIODIOEPH1 presets the transfer of zeros, DIOAX and DIODBX to the DIO add.-and DIO data registers at CL-9 in PH2. With DIODBX high, DIORWD is low indicating an RD instruction. FAIODIOEPH1 presets a DXJ and JXD for a J → D → J transfer in PH2.	83 84 45, 34	

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 2 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
2048 (PH2)	RA=nnnnn	-	Contents of (n) for transfer to (A)	1. E.A.=0000 0000 000n nnnn (RD)	The contents of (n) are transferred to J by JXR = S/RW/2 = PH2.FARWDO/GR.NLA10 and the address of (n) was preset in PH1. S/RW/2 presets DXJ and JXD for a J → D → J transfer in PH7. S/RW/2 also presets RWRITE for PH7. The address of (A) is preset by: S/RW/2 = RAX07 = S/RA4 = RAX06/1 = S/RA3 = S/RA2. S/RW/2 = S/PH7/1.	Sh 41,68 45,34 68 63-66 23
	RA=nnnnn	-	Contents of (n) for transfer back to (n) after bit 0 is reset.	2. E.A.=0000 0000 010n nnnn (WD)	The contents of (n) are transferred to J by JXR=S/RW/2 =PH2.FARWDO/GR.NLA10 and the address of (n) was preset in PH1. S/RW/2 presets DXJ and JXD for a J → D → J transfer in PH7. S/RW/2 also presets RWRITE for PH7. PH2.FARWDO/GR.NLAIO.LG09=NJ00. PH2.FARWDO/GR.NLA10.LG09.R00=S/OF presets the setting of the overflow indicator in PH7 if bit 0 of (n) = 1. S/RW/2 = S/PH7/1.	41 45,34 68 42 30 23
	RA=nnnnn	Contents of J-bus in PH1 (contents of A for transfer to (n))	Contents of D-register	3. E.A.=0000 0000 000n nnnn (WD)	The J → D → J transfer was preset in PH1. The address of (n) and RWRITE were preset in PH1 for writing into (n) in PH7. S/PH7=PH2.FARWDO/GR.NLA10.	23
	-	-	-	5. E.A.=0000 0000 1100 0000 (WD)	The transfer of J08,10,11,14 and 15 to set the appropriate status-bits were preset in PH1. S/PH7 = PH2.FARWDO.LG08.N(S/FLAGM-1).	23
	-	Status-bit configuration in area of byte 1, byte 0 = zero	Contents of D-register	6. E.A.=0000 0000 1100 0000 (RD)	The transfer of status → D → J was preset in PH1.	23
	-	-	Contents of D-register	7. E.A.=0000 0000 1110 XX00 (RD)	The transfer of status → D → J was preset in PH1.	
	-	-	Contents of D-register	8. E.A.=0000 0000 1111 XX00 (RD)	The transfer of status → D → J was preset in PH1. Status-bit II or EI is set after the transfer to J. S/PH7 = PH2.FARWDO.LG08.N(S/FLAGM-1).	23
	-	-	-	10. E.A.=0000 0000 1101 0000 (WD)	WAIT F.F. is set - CPU in WAIT condition. S/PH7 = PH2.FARWDO.LG08.N(S/FLAGM-1).	23
	-	-	-	11. E.A.=0000 0000 1101 1000 (WD)	EXIT F.F. is set. S/PH7 = PH2.FARWDO.LG08.N(S/FLAGM-1).	23
	-	-	-	12. E.A.=0000 0000 1110 XX00 (WD)	Status bits II or EI are reset.	23
	-	-	-	13. E.A.=0000 0000 1111 XX00 (WD)	Status bits II or EI are set. S/PH7 = PH2.FARWDO.LG08.N(S/FLAGM-1).	

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 3 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
2048	RA=00111	Contents of J in PH1 (contents of A)	-	14. E.A.=0001 0XXX 0000 0000 (WD)	<p>The data transferred to D is also present at the Interrupt Priority Control logic to operate on the interrupt levels according to the code of E.A. bits H05, 06 and 07. The 1st interrupt control clock (CLGAT) in PH2, however, is inhibited (refer to PH1). FUWD1.N(S/PH7).PH2 = RAUC/2. This sets the RA-reg. to up-count. RAUC/2 also inhibits phase advance (PHADV). The present value in RA=00111 (the add. of A).</p> <p>At the 1st CLock in PH2, RA=01000 and the inhibit is lifted from CLGAT allowing the 1st interrupt clock in PH2.</p> <p>If the code (E.A. bits H05, 06, and 07) ≠ 0, a 2nd CLGAT is available at the 2nd CLock in PH2 (RA=01001). If the code = zero, RADDR1.FUWD1.INTCODE0.PH2-1 = NINTCLEN/M1, and CLGAT is inhibited at the 2nd CL in PH2. S/PH7 = RADDR1.RA4(RA=01001).FUWD1.PH2. This drops RAUC/2 in as much as N(S/PH7). FUWD1.PH2=RAUC/2. With NRAUC/2 high, phase advance is enabled, however, FUWD1.NRAUC/2. (PH1+2+4+5) = NINTCLEN/M1, and further interrupt clocks (CLGAT) are inhibited until ENDE/1.</p>	Sh 62 23 6,7 6 23 6,7
	RA=00111	Contents of the I-reg. (Interrupt levels status)	-	15. E.A.=0001 0XXX 0000 ---- (RD)	<p>The state of the interrupt levels are read into the D-reg. from the I-reg. under the control of the code (E.A. bits H05, 06, 07). FURD1.PH2-2 = DXI = DXI/0 = DXFRL/1 controls the transfer of byte 0 of I-reg. to byte 0 of D-reg. as well as byte 1 of I-reg. to byte 1 of D-reg. via the FRL-register.</p>	44
			Contents of J in PH1 (contents of A) → Contents of D-register		16. E.A. = Mode & Device Address (DIORWD)	<p>The transfer of J → D → J was preset in PH1. FAIODIOEPH2 = DIOAXH presets the H (mode & add.) to DIO add-register transfer at CL-9 (PH3) for both RD and WD. FAIODIOEPH2.FUWD = DIODBXJ presets the J → DIO data-register transfer at CL-9 (PH3) for a WD, only.</p>

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 4 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
10XX	RA=00111	Contents of I-register	-	15. E.A.=0001 0XXX 0000 ---- (RD)	The D-register holds the interrupt levels status for transfer to J. FURD1.PH3 = S/JXD and S/RW, presets a D → J and RWRITE for PH7. FARWD1.PH3 = S/PH7.	Sh 35 68 23
	-	-	-	16. E.A. = Mode & Device Address (DIORWD - Mode 2-F)	The DIO address-register holds the mode and device address (E.A. from H-reg.). If it is a WD, the DIO data-register holds the data from the J-bus. The function strobe, DIOFS, is preset in PH4. FAIODIOEPH3 = HX/1 and S/JXHP1 presets clearing H and H-Plus-1 to J in PH4 when the H-register is used as a counter (interface timer, IT) if the condition for presetting DIOFS is not present. FAIODIOEPH3.NIOPON = S/SW1/2 presets an SW1 in PH4. SW1 identifies the FADIO phases.	84 52, 32 27 26
0889 (PH4)	-	-	-	16. H-register is cleared to zero upon entering PH4. If there is a delay in dropping DIOFSA from an immediately preceding DIORWD instruction, a value of one is added to H each clock thereafter in PH4 while DIOFSA remains high.	FARWD.SW1.PH4 = FADIOPH4. FADIOPH4.NDIOFSA = R/NDIOFS = DIOFS If NDIOFSA is low when entering PH4, FADIOPH4.DIOFSA.NCFIT = HXJ/2. HXJ/2 inhibits phase advance and enables the interface timer (IT), H-plus-1 to J to H for each clock in PH4 until DIOFSA drops or the IT runs out (CFIT). CFIT = (PH1+2+4+5).FARWD.SW1.NIO.H08 - Bit H08 = decimal value 128 or 42 microseconds. When NDIOFSA occurs, DIOFS is high at the interface, and the instruction advances to PH5 to await the return of DIOFSA. An H-plus-1 for PH5 is preset by FAIODIOEPH4 for timing the return of DIOFSA. If the IT should run out before the FS can be sent, the instruction advances to PH5 at which time DIOFSA is forced high by CFIT, inhibiting FS and advancing the instruction on through the final phases (aborting the instruction). FADIOPH4 = S/SW1/2 presets an SW1 in PH5. SW1 identifies the FADIOPH.	26 84 52 23 3 32 84 27 26
0489 (PH5)	-	-	-	16. The H-register is still zero or contains some value less than H08 (timer runoff). A value of one is added each clock in PH5 while waiting for the return of DIOFSA	FARWD.SW1.PH5 = FADIOPH5. While the NDIOFSA state remains, FADIOPH5.NDIOFSA = HXJ/2 inhibits phase advance (PHADV) and enables the interface timer condition (H-plus-1 to J to H) for each clock in PH5 until DIOFSA or CFIT. If DIOFSA occurs before timer runoff (CFIT), the instruction proceeds to PH6. If CFIT occurs first, the instruction is aborted (refer to PH4 comments). If an I/O service call is acknowledged, IOASC = R/NIOASC = IO:SC.IOPTION.(FARWD.SW1), before the return of DIOFSA, the instruction remains in PH5 while the IO is completed. During this time the IT continues counting, however, IO overrides CFIT and	26 52 23 77 3

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 5 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
0489 (PH5, Contd)	-	-	-	16. (PH5 continued)	the value in H has no significance. When the IOSC is completed (IOPH17) the H-register is cleared to zero. If DIOFSA is still absent, FADIOPH5.N(S/IO/1) = S/JXHP1 and FADIOPH5.NDIOFSA = HXJ continues to enable the interface timer and the count starts over again from zero. The instruction is aborted at CFIT if DIOFSA does not appear in 42 usec. FADIOPH5.N(S/IO/1) = S/SW1/1 presets an SW1 in PH6 to identify the next FADIOPH.	Sh 52 32 52
0289 (PH6)	-	Data from the DIO interface (RD)	-	16.	FARWD.SW1.PH6 = FADIOPH6. If the instruction has been successful, two status bits are received from the device in PH6, DIO:CC3 and DIO:CC4. These bits of information control the setting of the overflow (OF) and/or carry (CF) indicators. The address of the accumulator is preset by FADIOPH6 = RAX07 = S/RA4 = RAX06/1 = S/RA3, S/RA2. FURD.FADIOPH06 = S/RW presets a RWRITE. Data on the DIO:DB lines is transferred to the D-reg. by: PH6-1.FURD = DXDIO:DB = DXI/O DXDIO:DB-1. Byte 0 in D is transferred via the I-register by DXI/O and byte 1 is transferred directly by DXDIO:DB-1. This data is for storage in (A). FADIOPH6.S/JXD presets a D → J in PH7.	88 30, 31 63 65, 66 68
0148 (PH7)	RA=00111	-	Contents of D-register (Data from the DIO interface)	16.	For the Read Direct DIORWD, a RWRITE was preset in PH6 as well as the address of (A). The final transfer of data from the device to (A) is at the arrival of the next CK(PRE1). RWRITE.CK = K/R0007 = K/R0815 This enables the transfer of data on the J-bus to the addressed register. The transfer of D → J was preset in PH6.	69
	RA=00111	Contents of J from PH2	Contents of D-register	1. E.A.=0000 0000 000n nnnn (RD)	The transfer of the contents of (n) from J → D → J was preset in PH2 as well as RWRITE and the address of (A). NOTE: If the address of (n) was the P-reg. an SXJ preset in PH2 also transfers J → S as the next instruction address: PH2.S/PH7.RA0003EZ.RA4/1 = SXJ. The final step of the J → (A) is accomplished at the next CK (PRE1). RWRITE.CK = K/R0007 = K/R0815 This enables the transfer of data on the J-bus to the addressed fast memory register.	56 69

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 6 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
0148 (PH7, Contd)	RA=nnnn	Contents of J in PH2	→ Contents of D-register	2. E.A.=0000 0000 010n nnnn (WD)	The transfer of the altered contents of (n) from J → D → J was preset in PH2 as well as RWRITE. The address of (n) was high in PH2. The final step of the J → (n) transfer is accomplished at the next CK (PRE1). RWRITE.CK = K/R0007 = K/R0815 This enables the transfer of data on the J-bus to the addressed fast memory address.	Sh 69
	RA=nnnn	-	-	3. E.A. = 0000 0000 000n nnnn (WD)	Fast memory register (n) contains the contents of the J-bus. The address of (n) was set in PH2 as well as RWRITE. The final step was accomplished at the arrival of the next CK (PH7). RWRITE.CK = K/R0007 = K/R0815 This enabled the transfer of data on the J-bus to the register (n). NOTE: If the address of (n) was the P-reg., an SXJ preset in PH2 also transfers the J-bus to S as the next instruction address.	69
	RA=00111	Contents of J in PH1	→ Contents of D-register	4. E.A.=0000 0000 1000 0000 (RD)	The transfer of J → D → J was preset in PH1 as well as RWRITE and the address of (A). The final step of J → (A) transfer is accomplished at the next CK (PRE1). RWRITE.CK = K/R0007 = K/R0815 This enables the transfer of data on the J-bus to (A)	69
	-	-	-	5. E.A.=0000 0000 1100 0000 (WD)	Status bits are set to the new configuration dictated by the contents of (A).	
	RA=00111	-	-	6. E.A.=0000 0000 1100 0000 (RD)	The accumulator (A) contains the status-bit configuration (for all three instructions 6,7 and 8) transferred to J in PH2. The final step was accomplished at the arrival of CK for PH7. RWRITE.CK = K/R0007 = K/R0815 This enabled the transfer of data on the J-bus to the accumulator (a).	69
	RA=00111	-	-	7. E.A.=0000 0000 1100 XX00 (RD)	The II or EI status-bit is reset.	
	RA=00111	-	-	8. E.A.=0000 0000 1110 XX00 (RD)	The II or EI status-bit is set.	

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 7 of 8)

DISPLAY (Ref. Figure 3-9)					Comments	Figure Ref. 3-12
PH	O(RA)	D-Register	J-Bus	H-Register		
0148 (PH7, Contd)	RA=00111	-	-	9. E.A.=0000 0000 1000 XXXX (WD)	The transfer of (A) to a memory protect register was preset in PH1 as well as the address of (A). The address of the memory protect register was selected by E.A. bits H12-H15. The final step of (A) → MP is accomplished at the next CK (PRE1) MPXR.CK = K/MP07A = K/MP07B This enables the transfer of a general register to the memory protect register.	Sh 69
	-	-	-	10. E.A.=0000 0000 1101 0000 (WD)	CPU is in a WAIT condition	
	-	-	-	11. E.A.=0000 0000 1101 1000 (WD)	EXIT is high for the generation of the FUEXIT instruction (FULDX.EXIT)	
	-	-	-	12. E.A.=0000 0000 1110 XX00 (WD)	The II or EI status bit is reset	
	-	-	-	13. E.A.=0000 0000 1111 XX00 (WD)	The II or EI status bit is set	
	-	-	-	14. E.A.=0001 0XXX 0000 0000 (WD)	The interrupt priority levels status have been altered.	
	RA=00111	Contents of I-register (transferred in PH2)	Contents of D-register	15. E.A.=0001 0XXX 0000 ---- (RD)	The D → J transfer was preset in PH3 as well as RWRITE. The contents of the D-register (status of interrupt levels) is loaded from the J-bus into (A) at the next CK (PRE1). RWRITE.CK = K/R0007 = K/R0815. This enables the transfer of data on the J-bus to (A).	

Table 3-6. Direct Control Instruction Execution Sequence Control (sheet 8 of 8)

PLEASE NOTE!

THE FOLLOWING LOGIC DIAGRAMS ARE INCLUDED FOR INFORMATION PURPOSES ONLY AND WILL NOT BE UPDATED. FOR CURRENT TECHNICAL INFORMATION REFER TO THE LATEST ISSUE OF ENGINEERING SUPPORT MANUAL, SIGMA 3 CENTRAL PROCESSOR, MODEL 8101, ASSEMBLY NO. 153253, DOCUMENT NO. 902400.

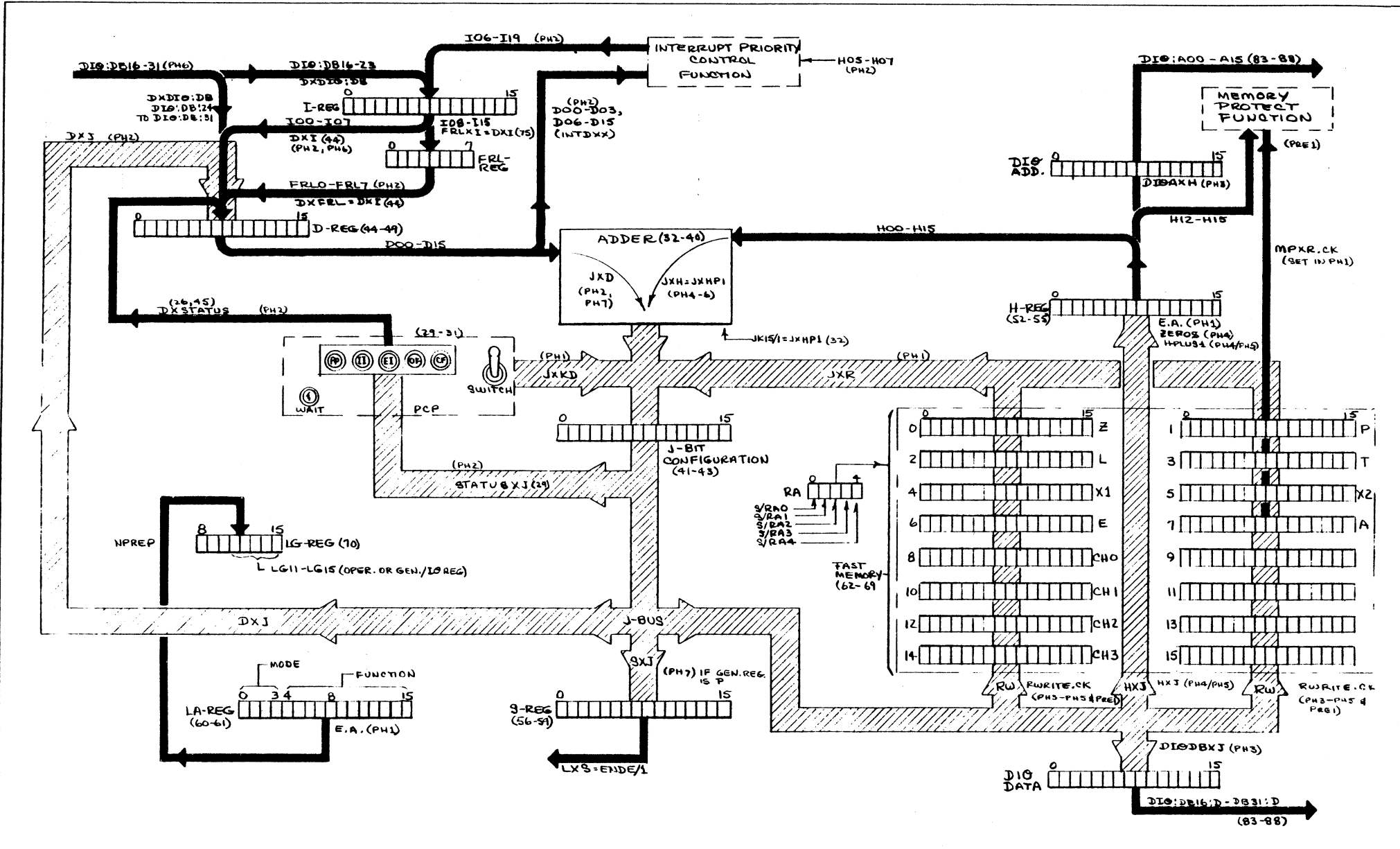


Figure 3-9. Direr Control Instruction Execution - Data F¹ (Exclud. I/O RD)

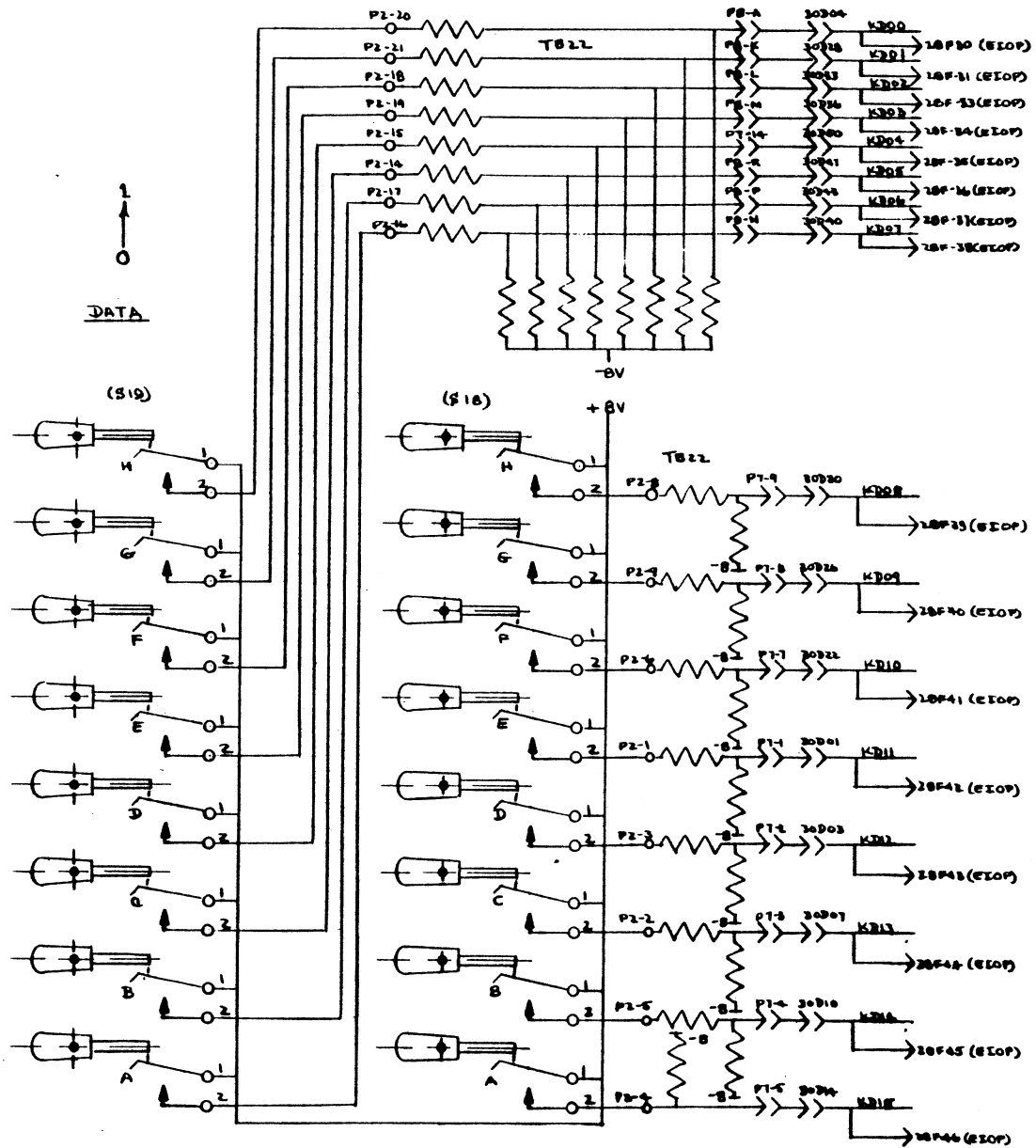
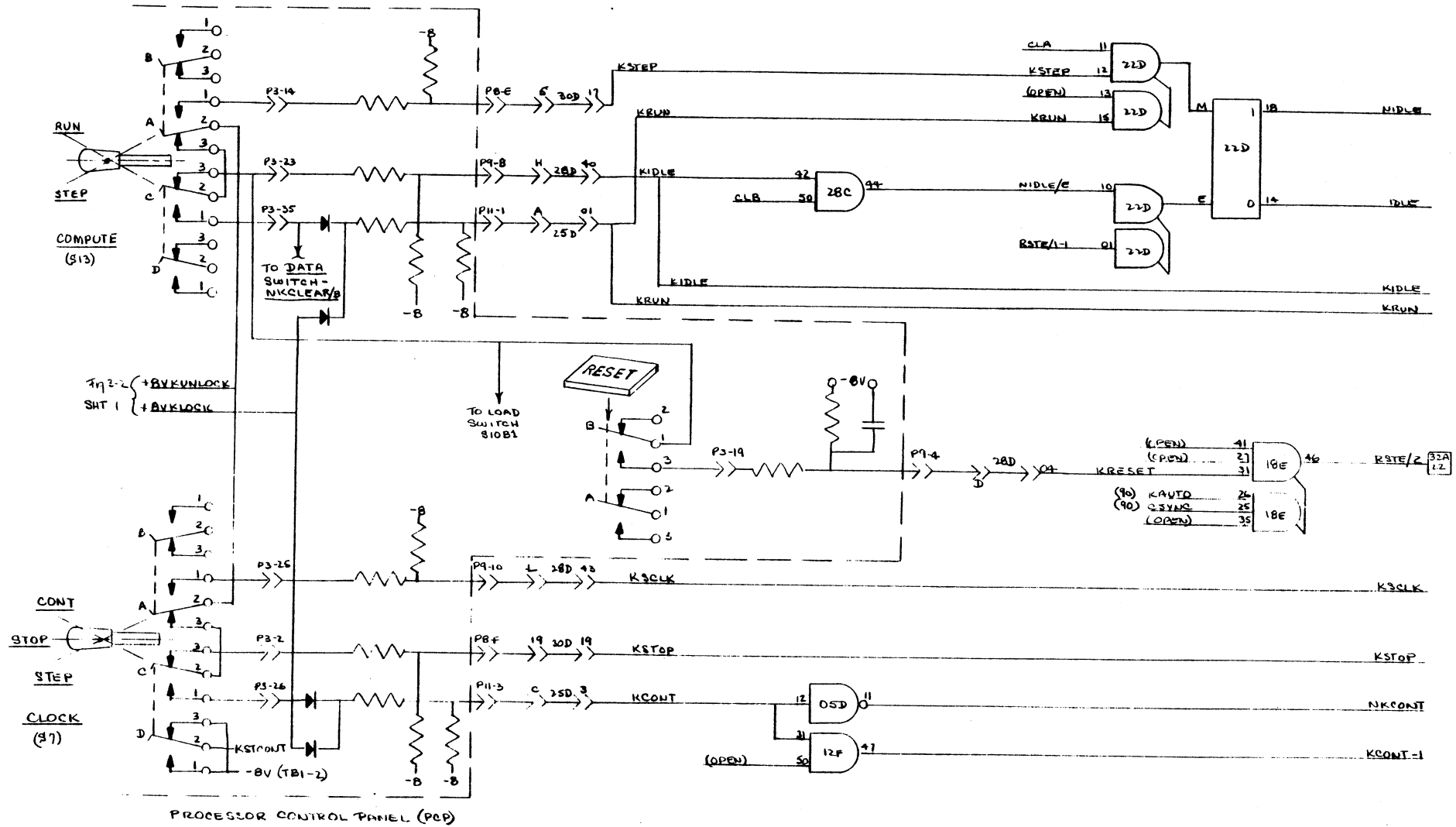


Figure 3-12. Logic Diagrams (sheet 1 of 91)



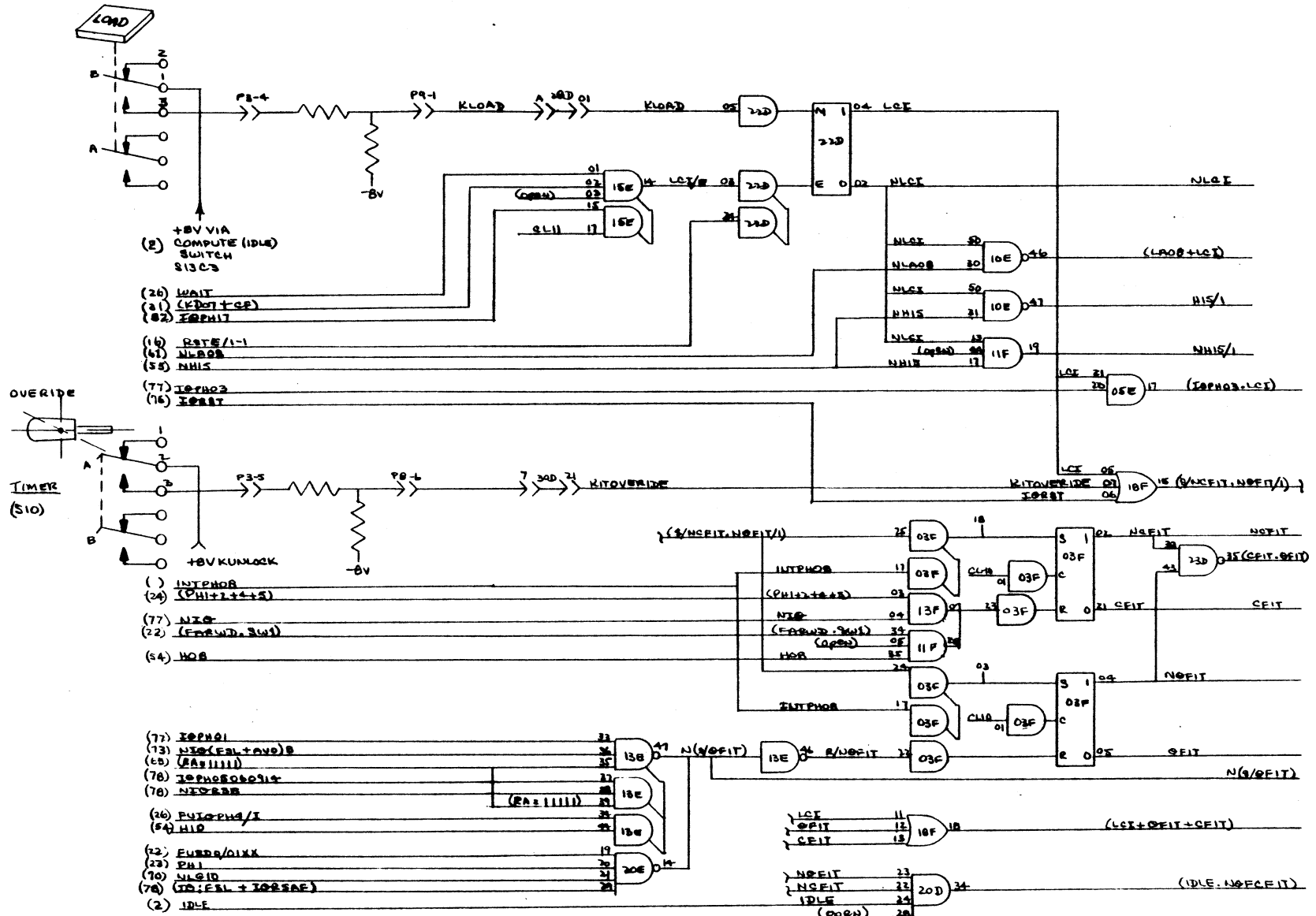


Figure 3-12. Logic Diagrams (sheet 3 of 91)

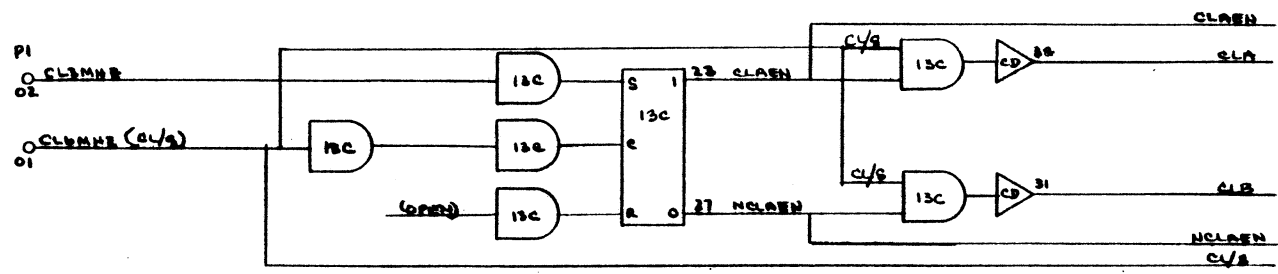
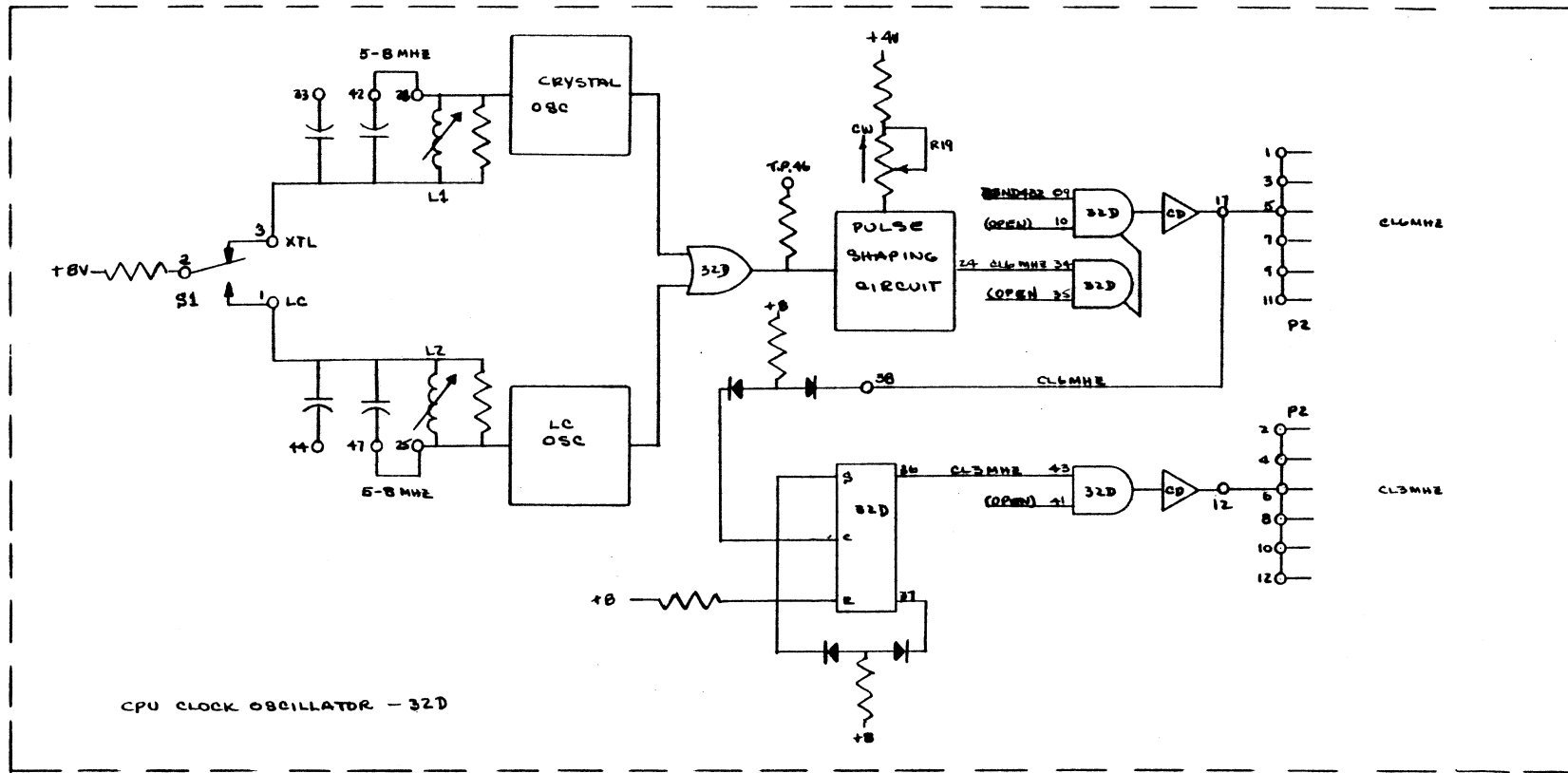


Figure 3-12. Logic Diagrams (sheet 4 of 91)

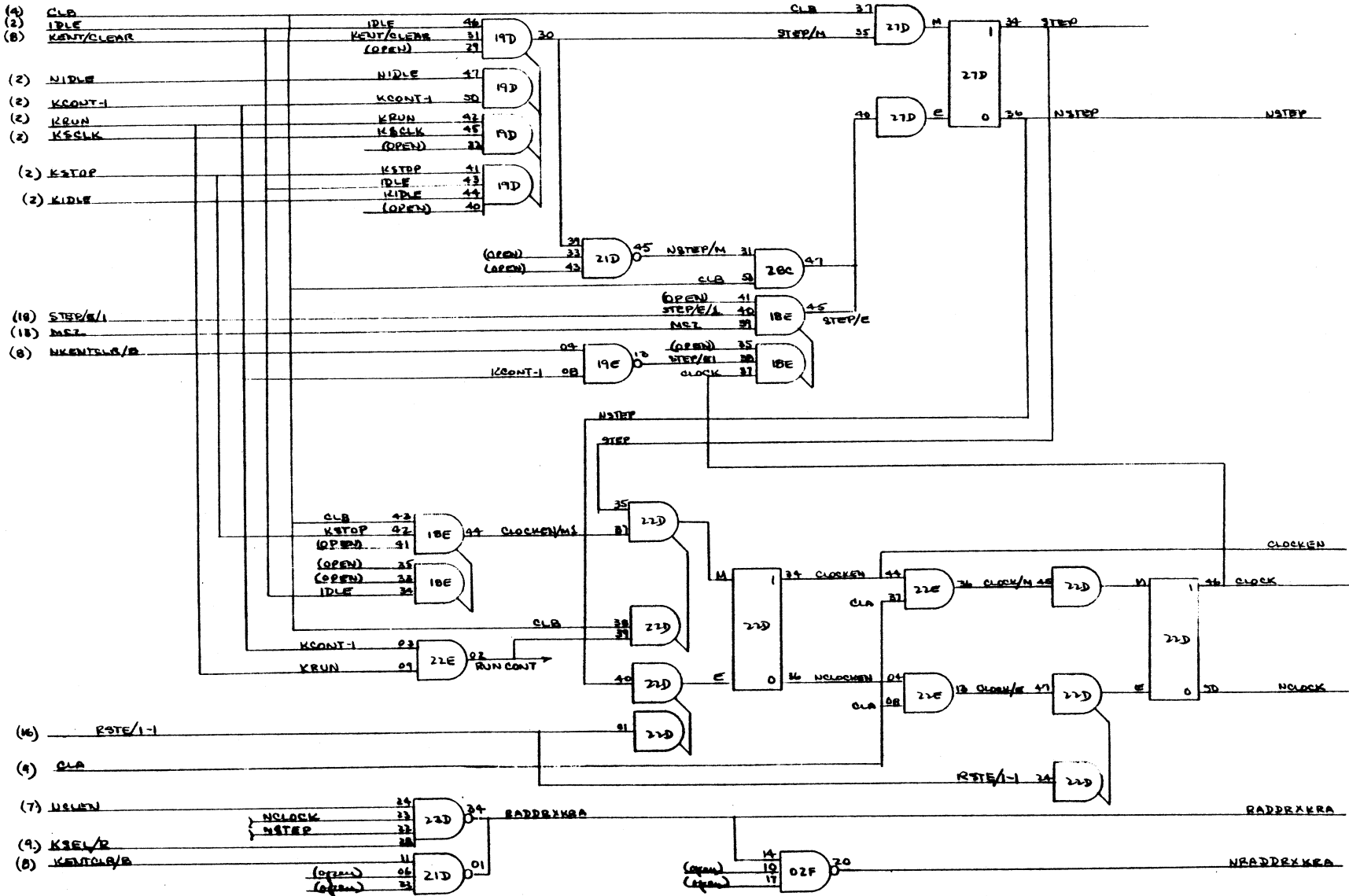


Figure 3-12. Logic Diagrams (sheet 5 of 91)

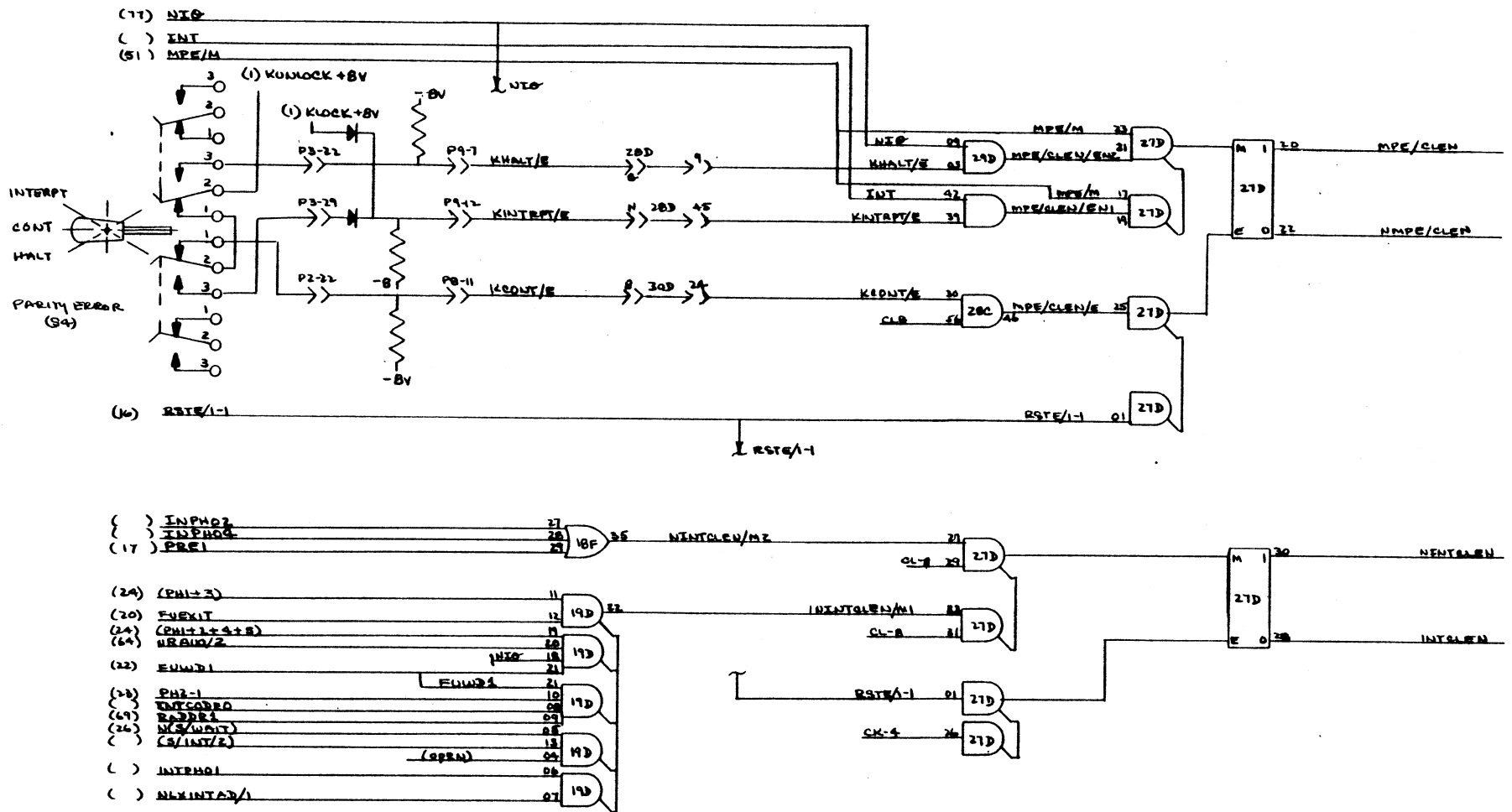


Figure 3-12. Logic Diagrams (sheet 6 of 91)

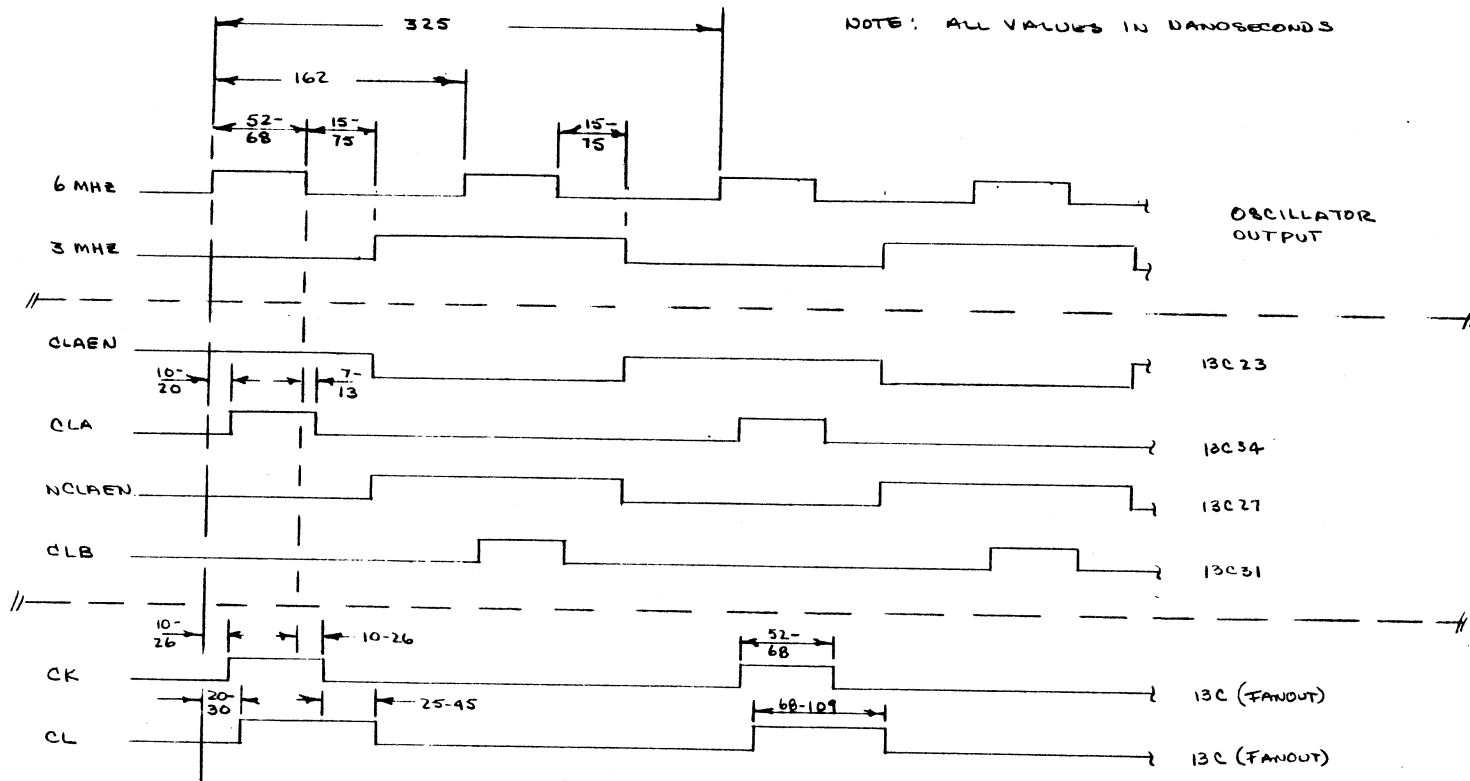
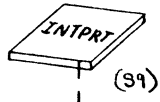


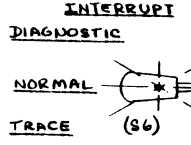
Figure 3-12. Logic Diagrams (Clock Timing Diagrams - sheet 6A of 91)

(22) (S/FLAAM)-1



(91) I11S

(17) PREP-1



- (1) KUNLOCK +BY
- (71) NKD
- (51) MPE/M
- NINTRPY/E

() IDGP

- (71) I1LP
- (91) I1LN
- (29) NIT
- (16) B3TE/I-1
- (25) TRAP/E

(3) (CPIT+QFIT)

- (20) NFADEX
- () NMPV

- (14) ENDE
- (17) ENDP-1

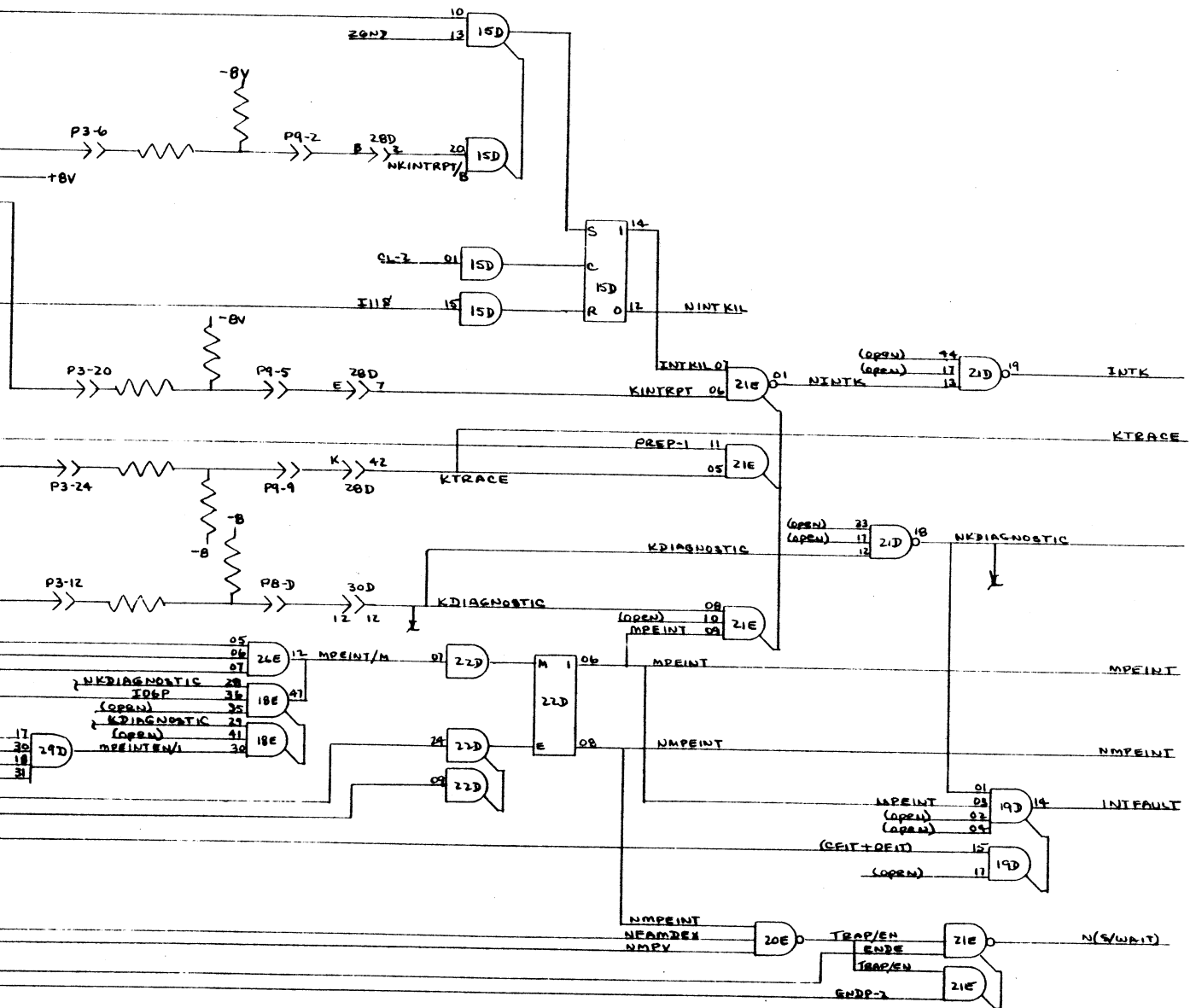


Figure 3-12. Logic Diagrams (sheet 7A of 91)

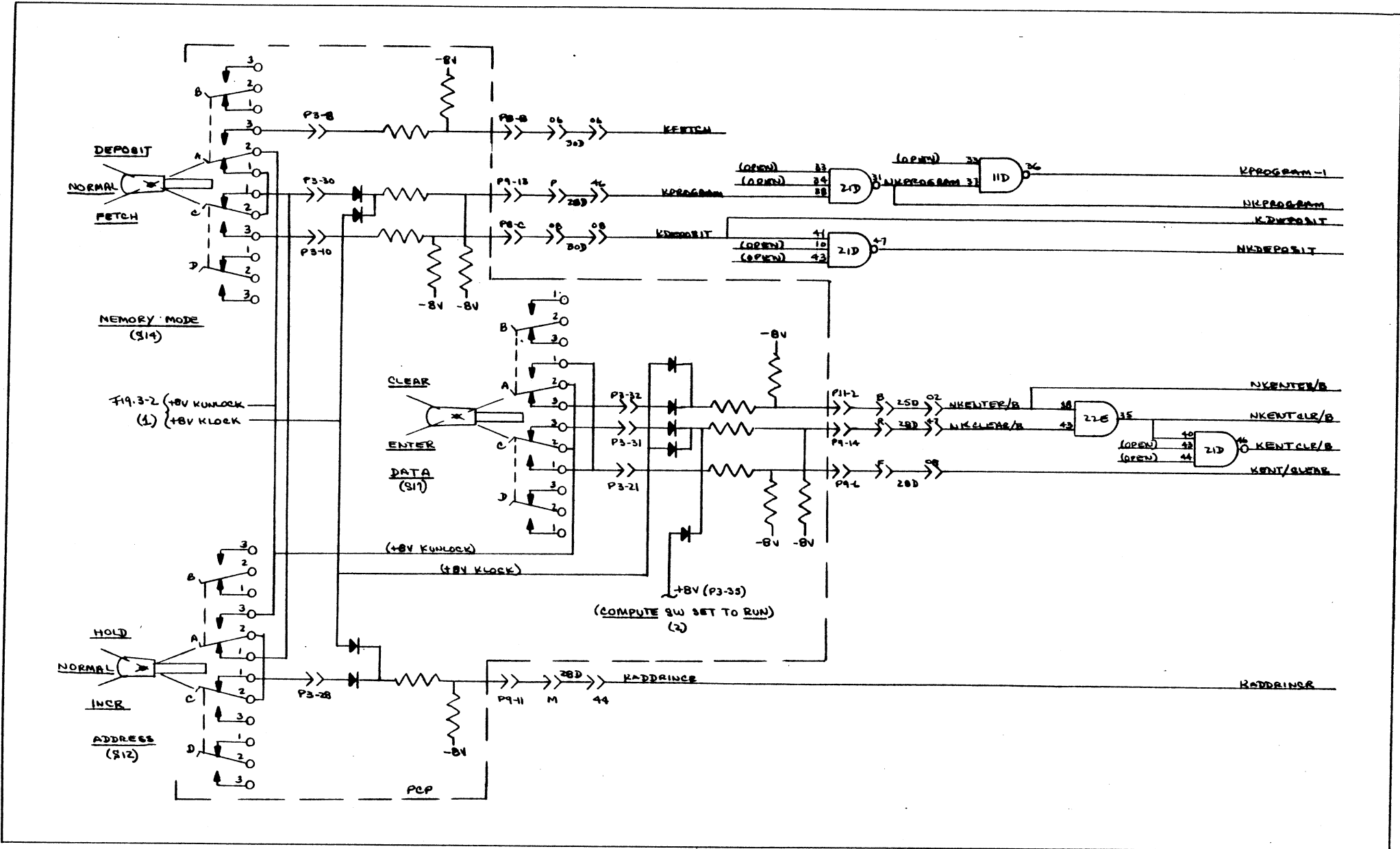


Figure 3-12. Logic Diagrams (sheet 8 of)

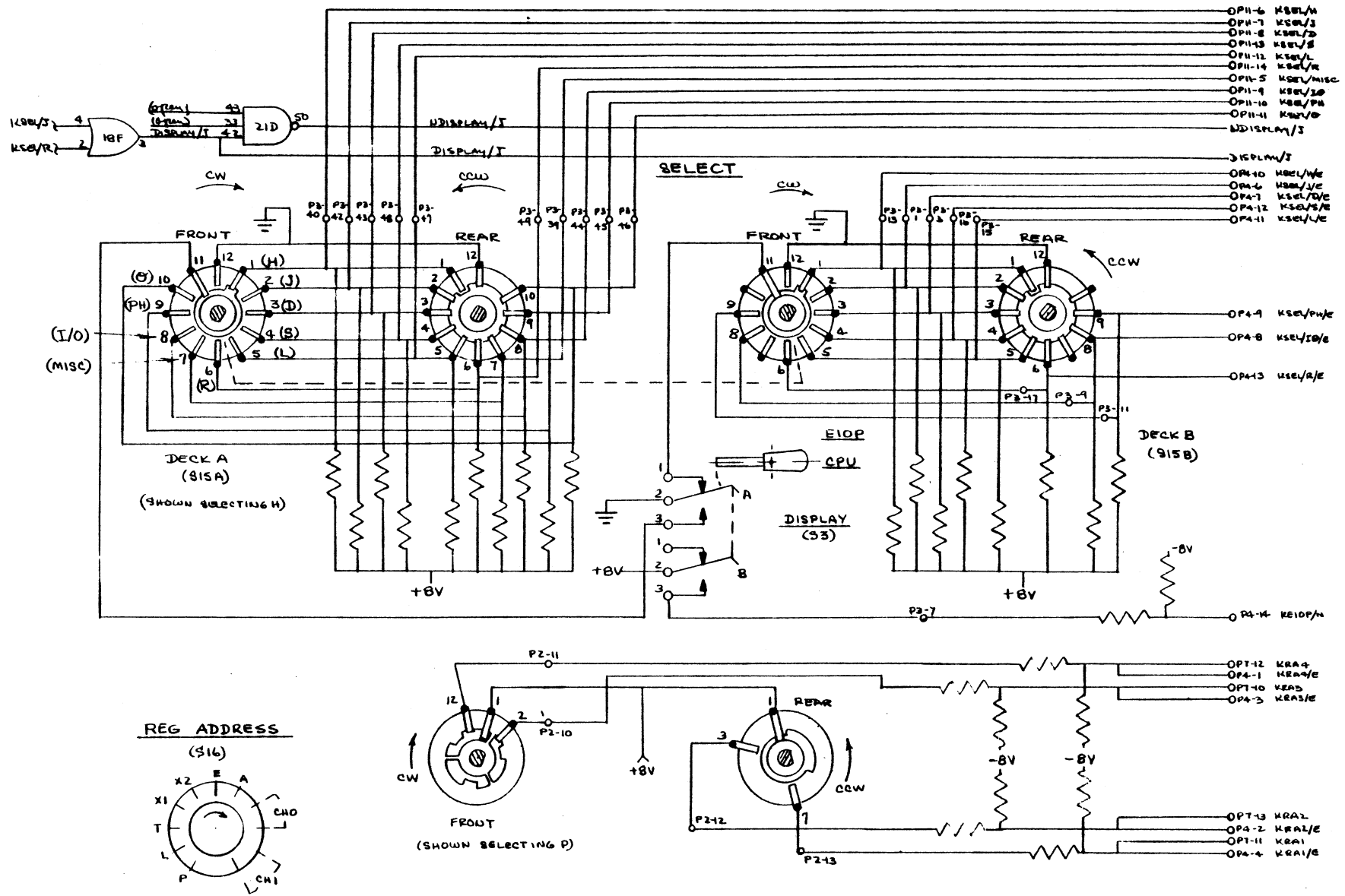


Figure 3-12. Logic Diagrams (sheet 9 of 91)

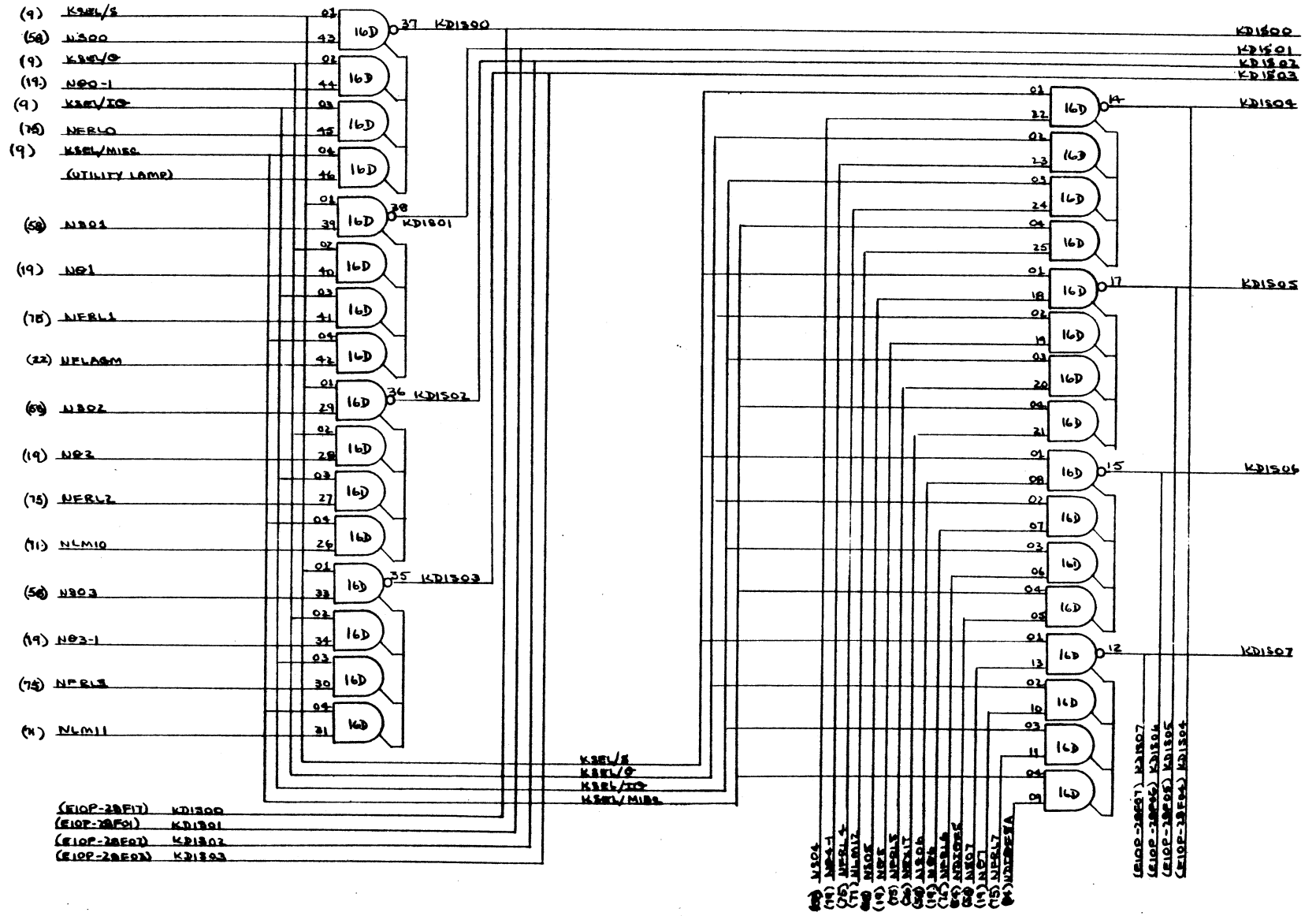


Figure 3-12. Logic Diagrams (sheet 10 of 21)

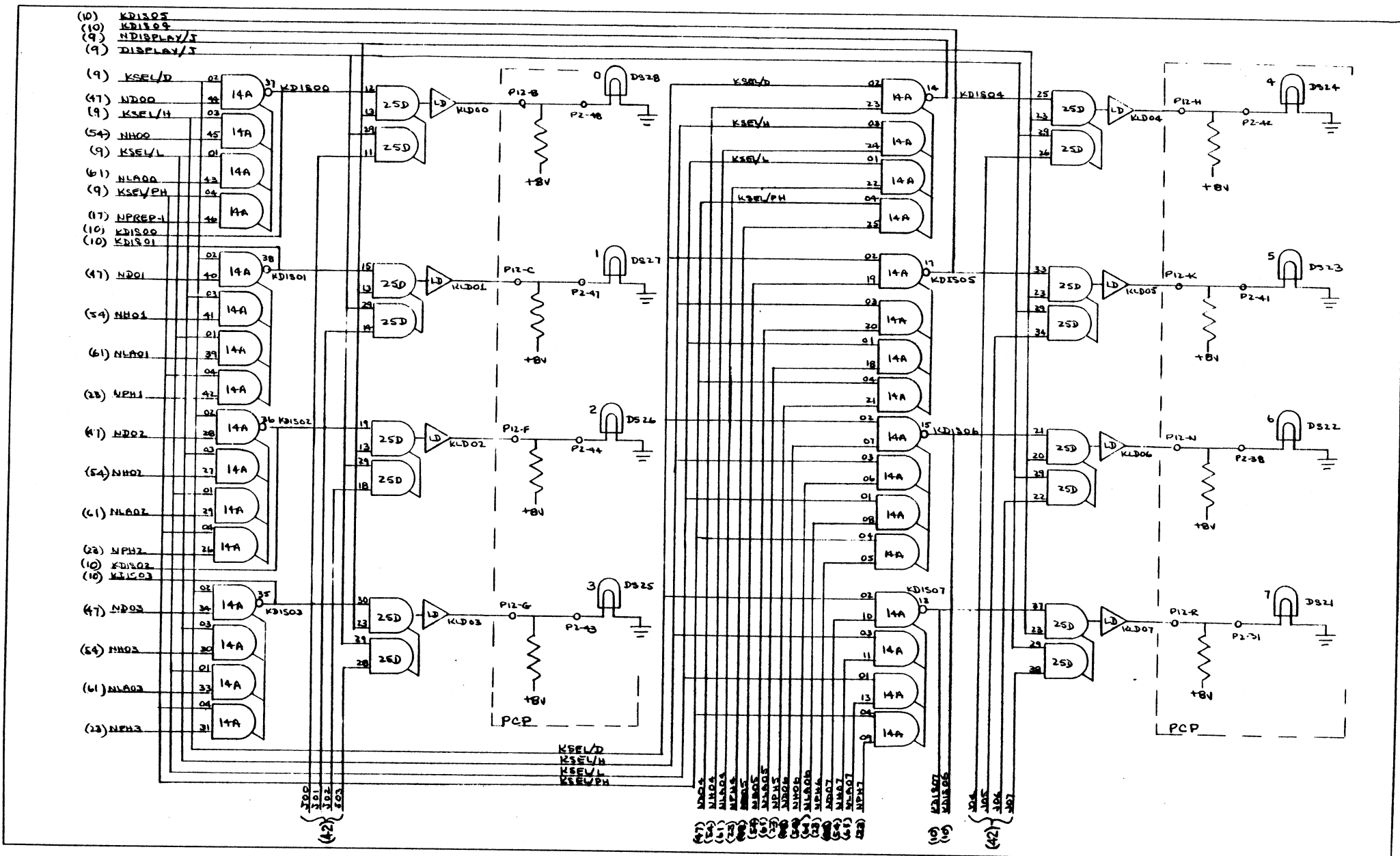


Figure 3-12. Logic Diagrams (sheet 11 of 91)

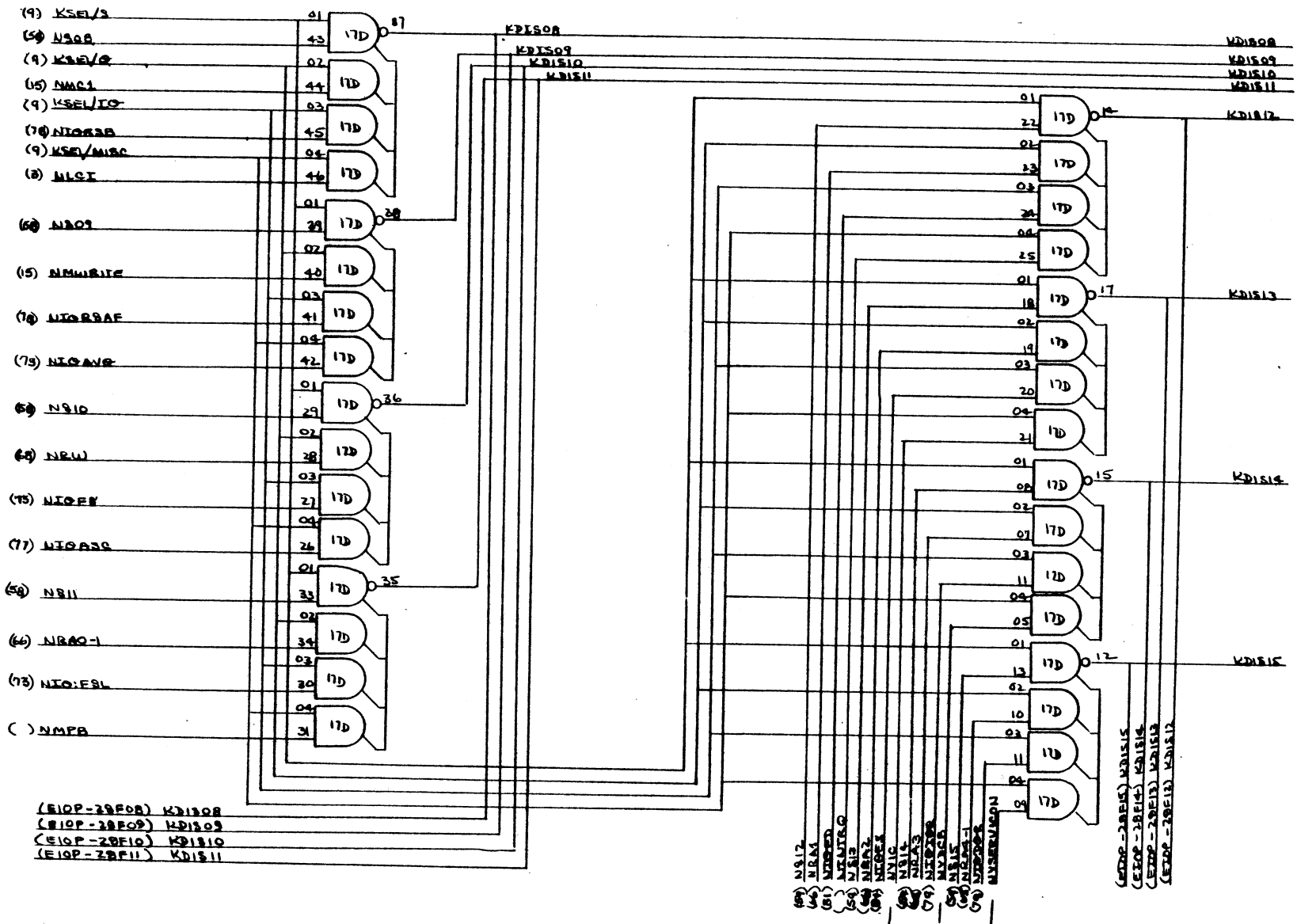


Figure 3-12. Logic Diagrams (sheet 12 of 91)

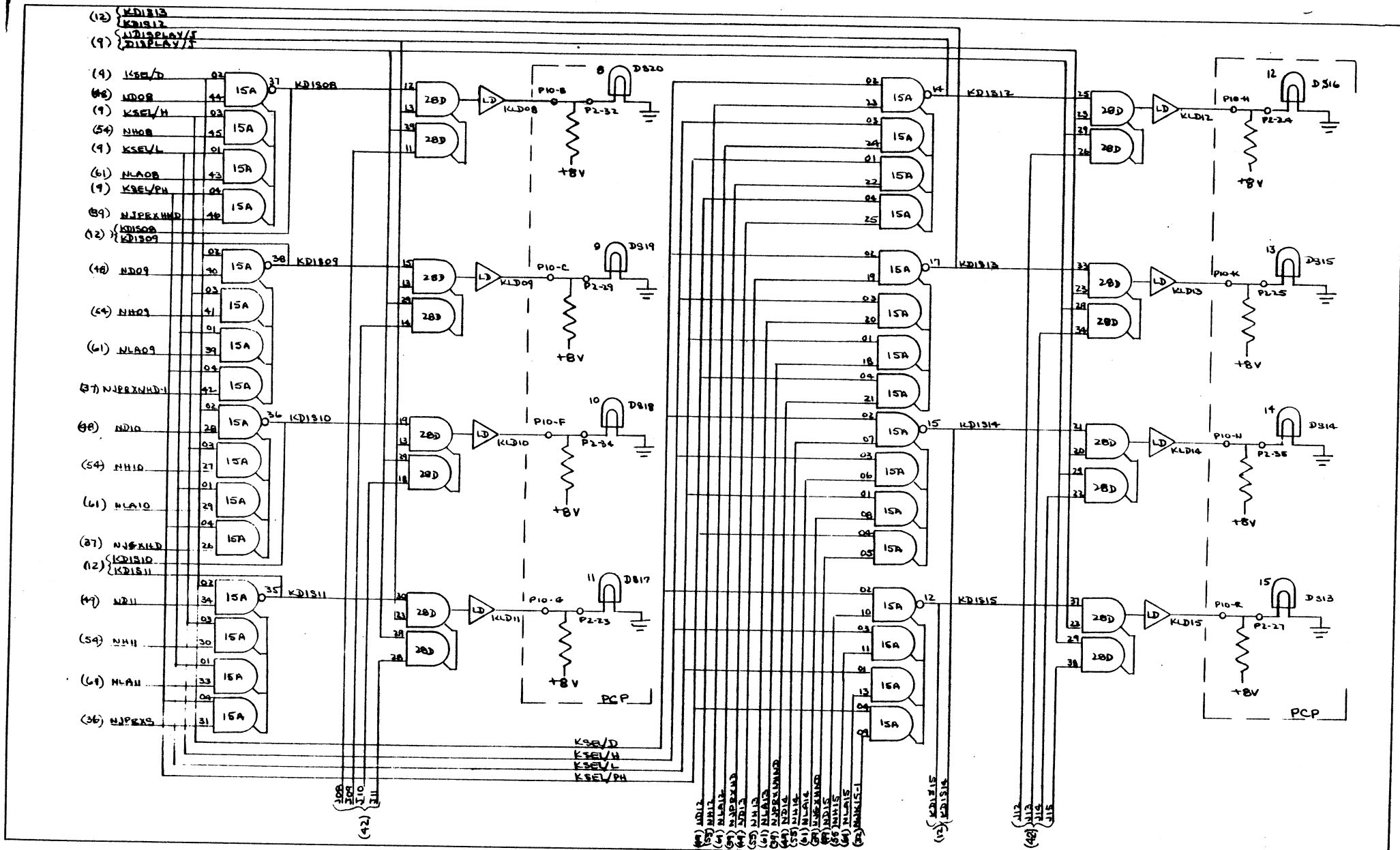


Figure 3-12. Logic Diagrams (sheet 13 of 91)

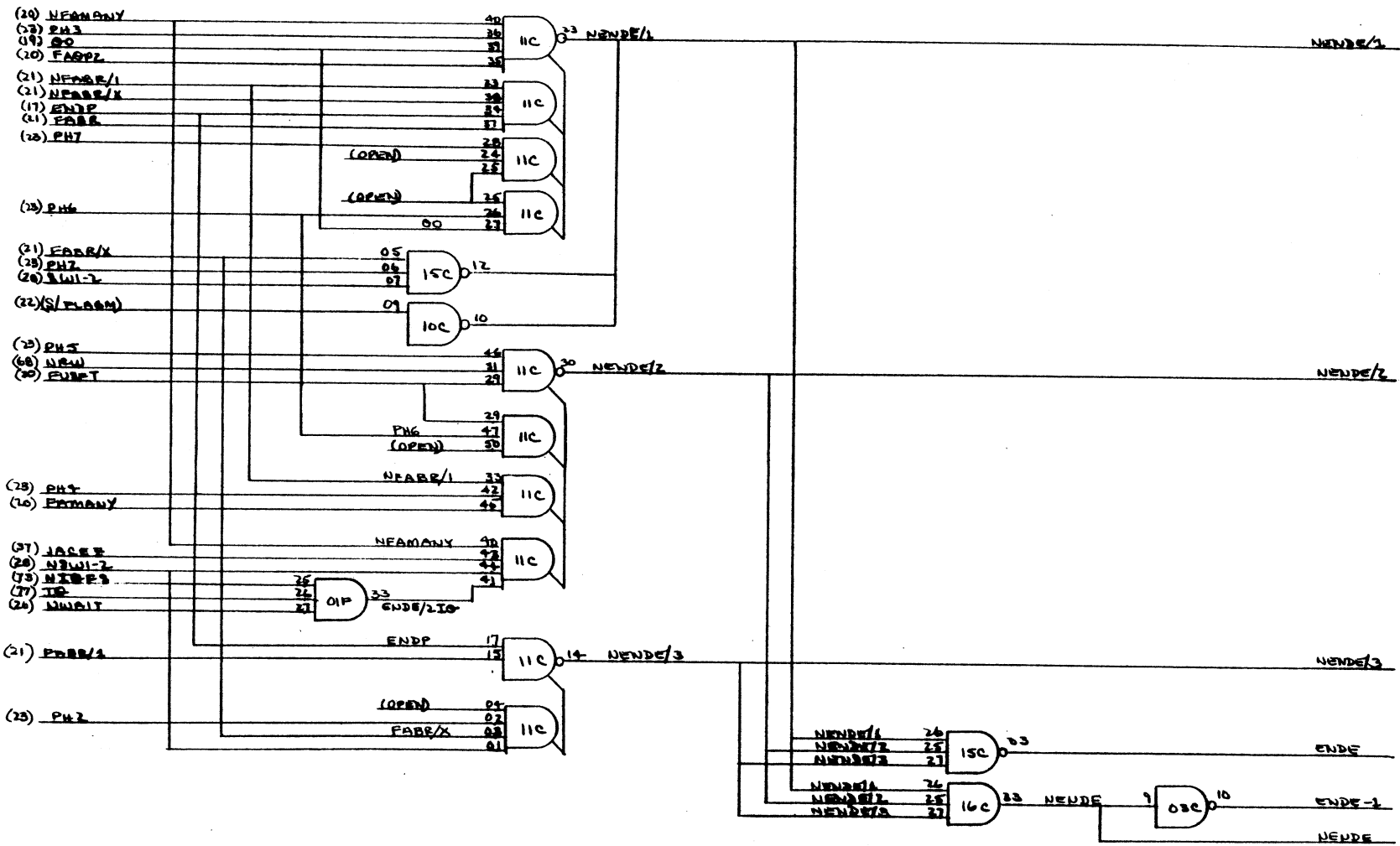


Figure 3-12. Logic Diagrams (sheet 14 of 21)

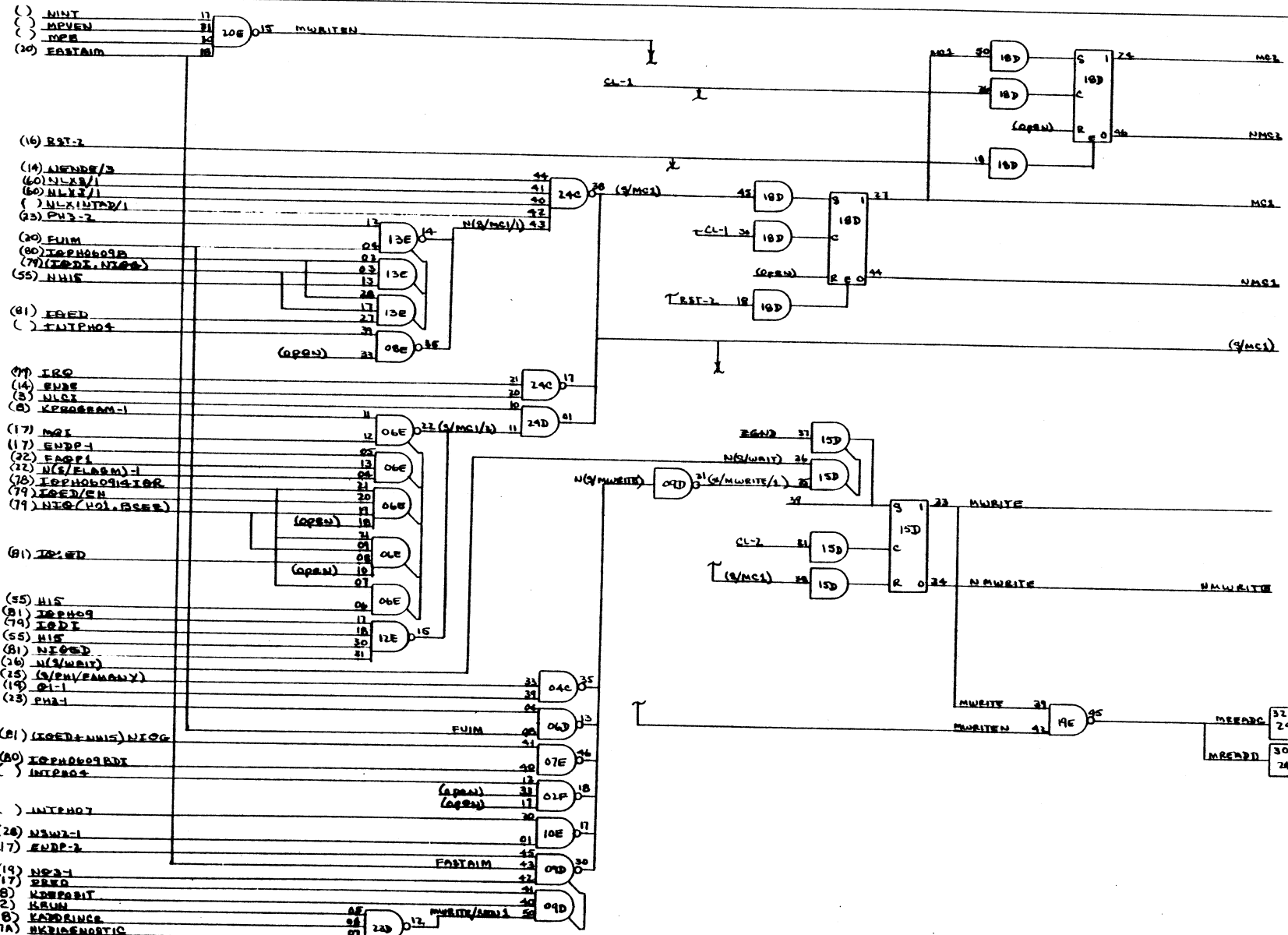


Figure 3-12. Logic Diagrams (sheet 15 of 91)

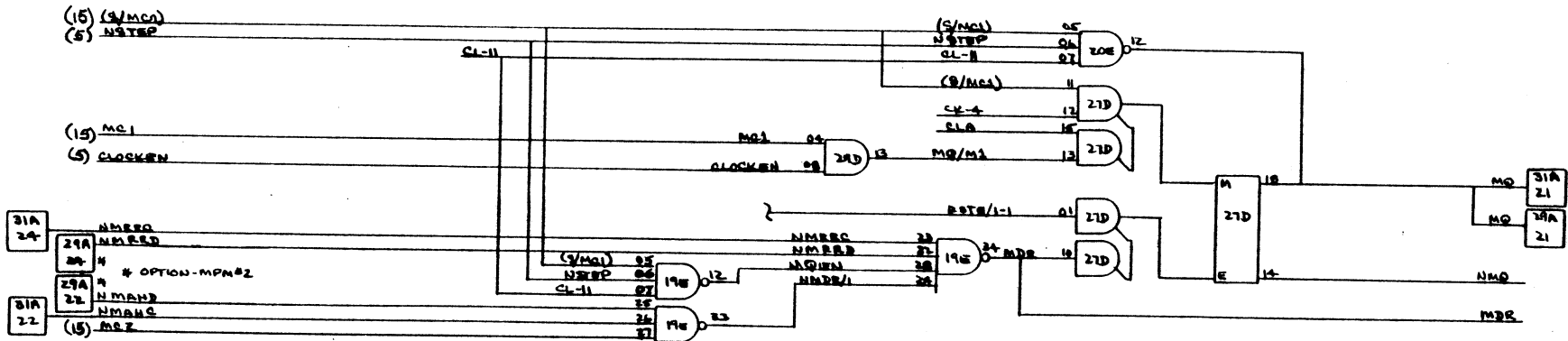
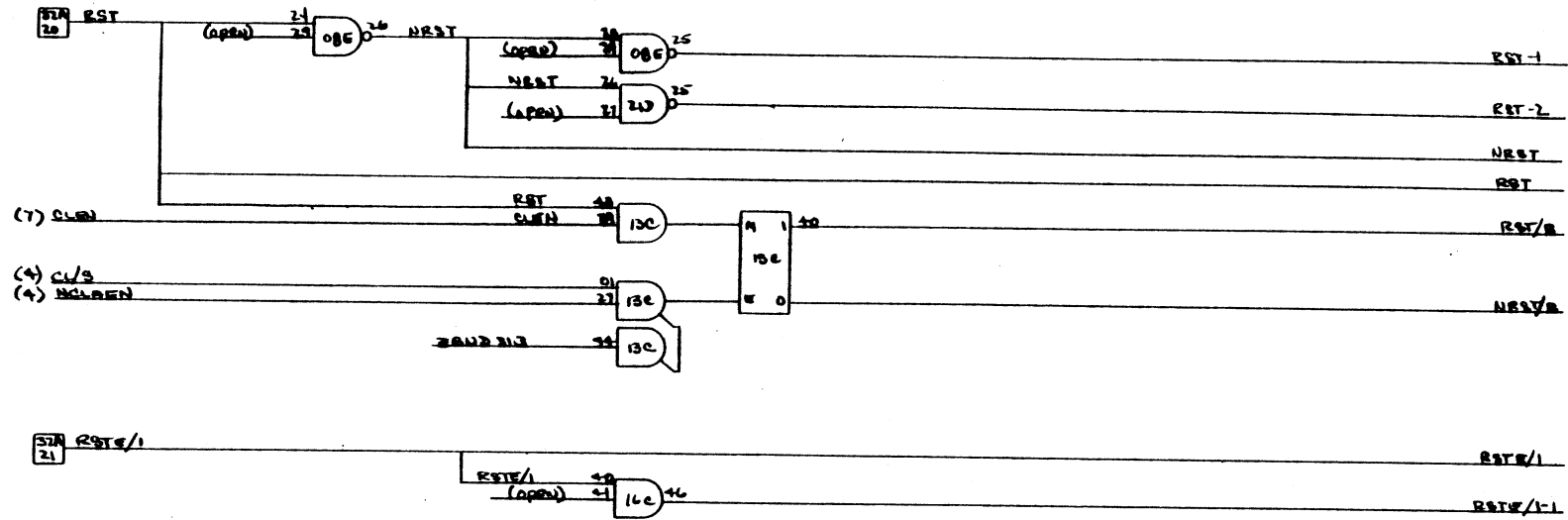


Figure 3-12. Logic Diagrams (sheet 16 of 21)

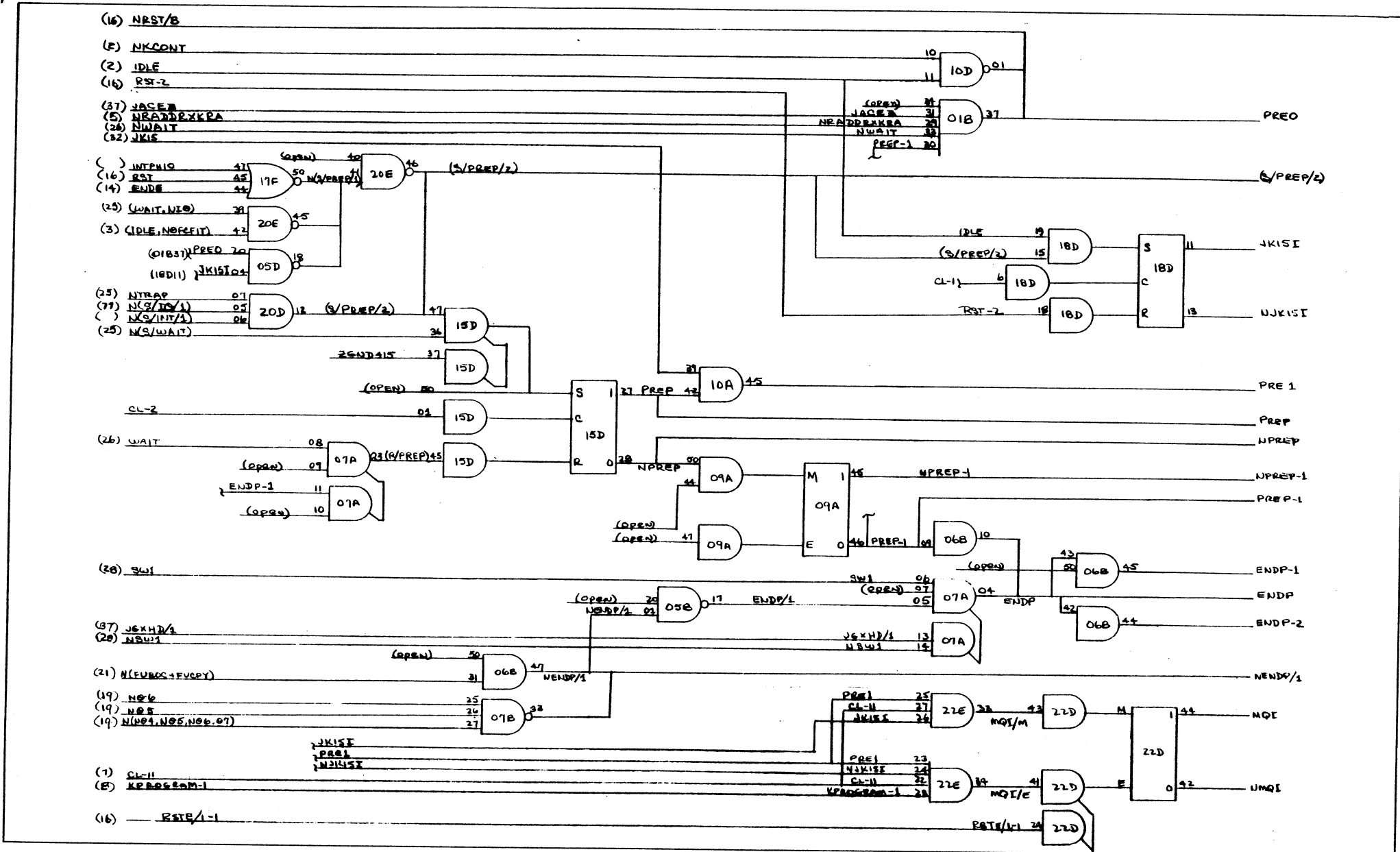


Figure 3-12. Logic Diagrams (sheet 17 of 91)

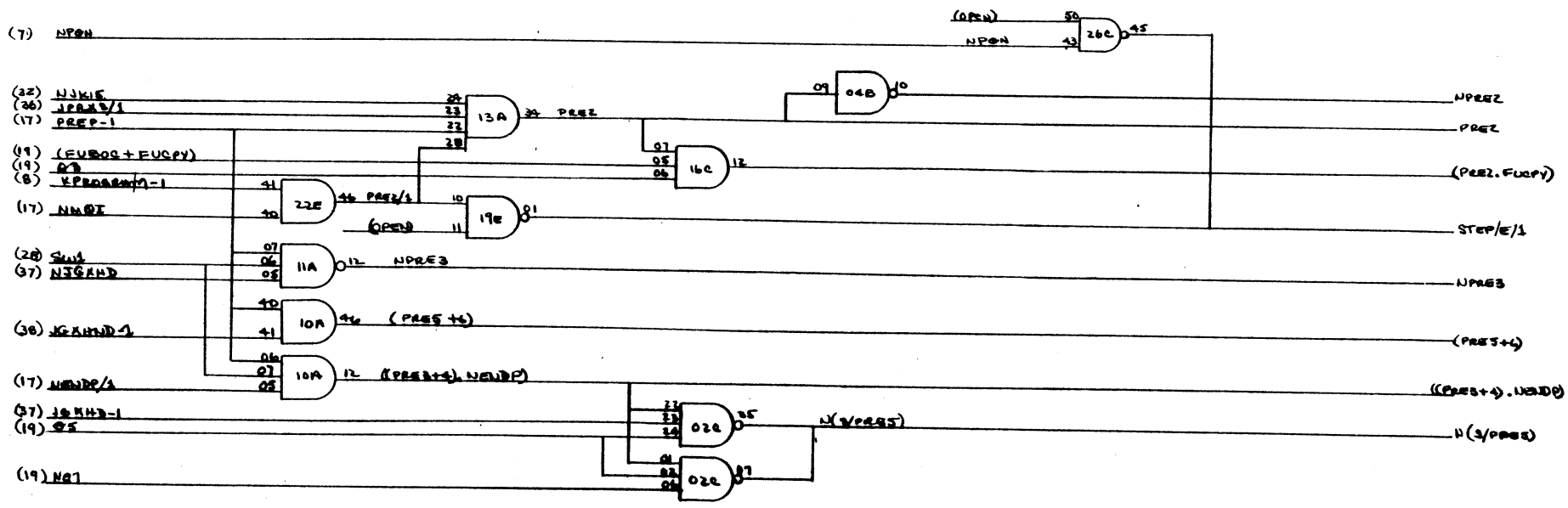
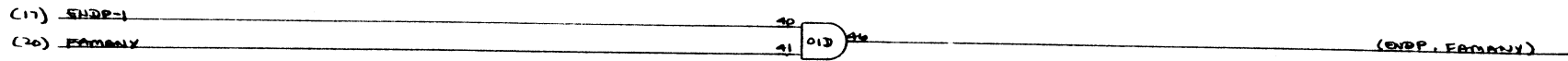


Figure 3-12. Logic Diagrams (sheet 18 of 91)

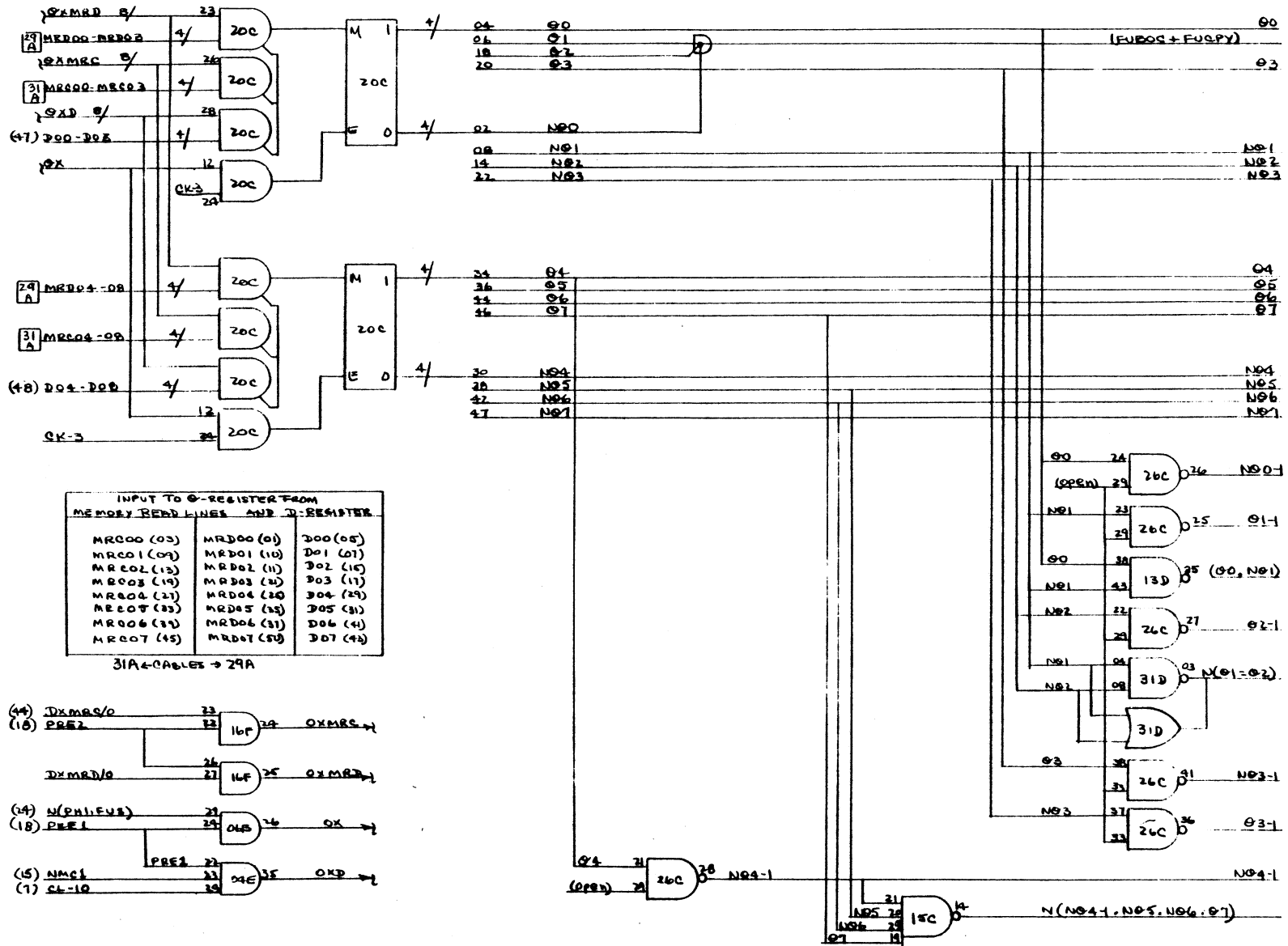


Figure 3-12. Logic Diagrams (sheet 19 of 91)

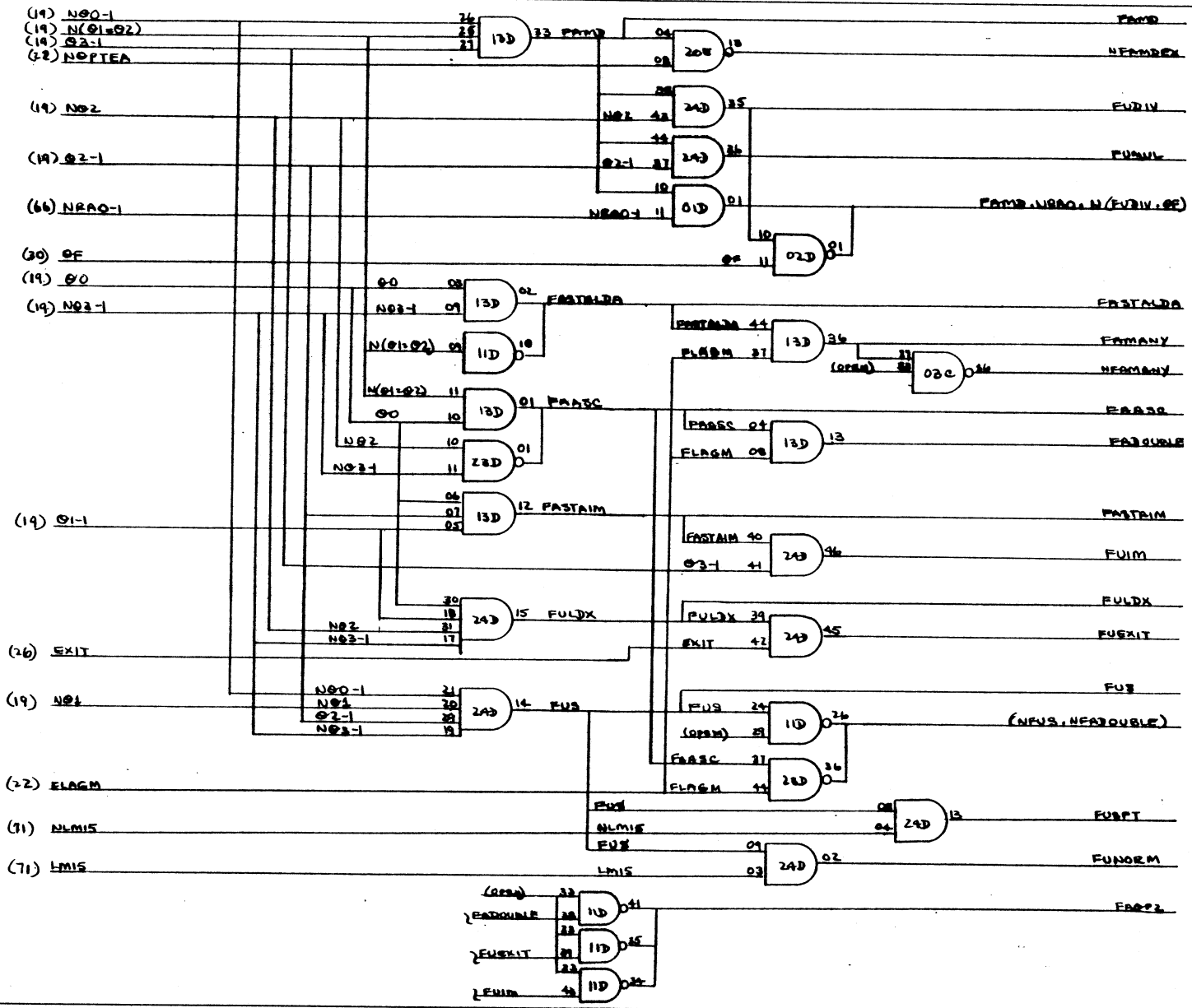


Figure 3-12. Logic Diagrams (sheet 20 of 91)

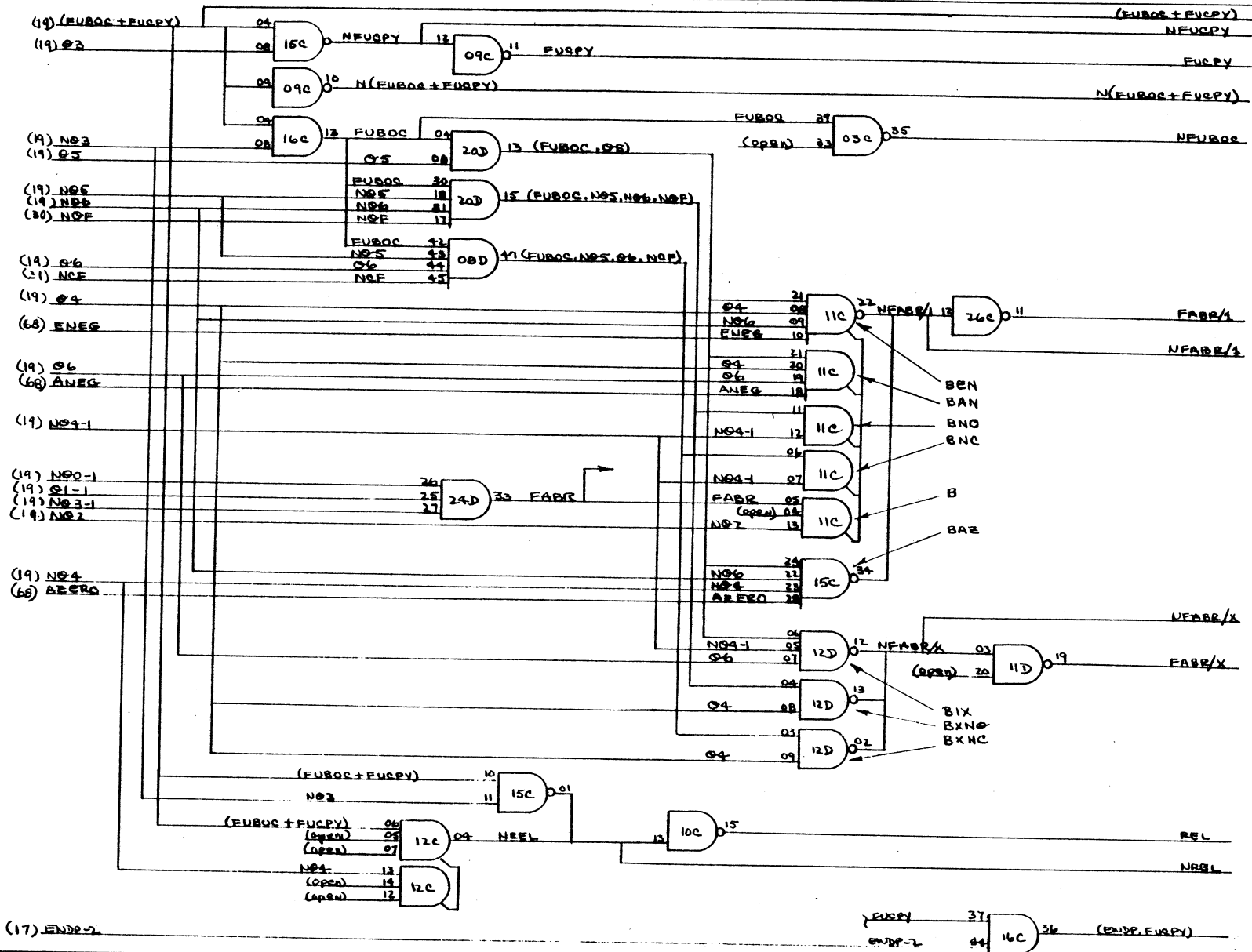


Figure 3-12. Logic Diagrams (sheet 21 of 91)

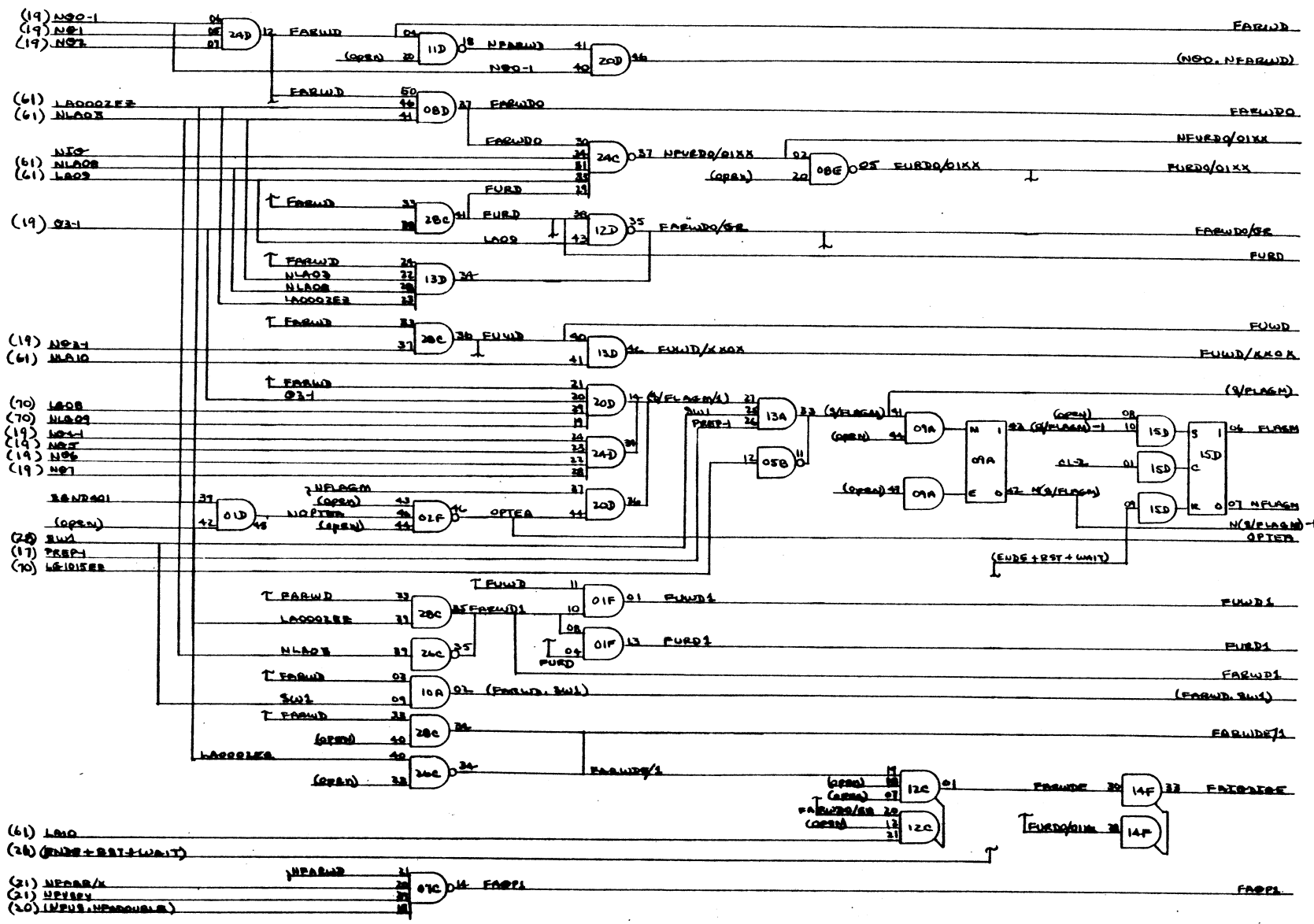


Figure 3-12. Logic Diagrams (sheet 22 of 91)

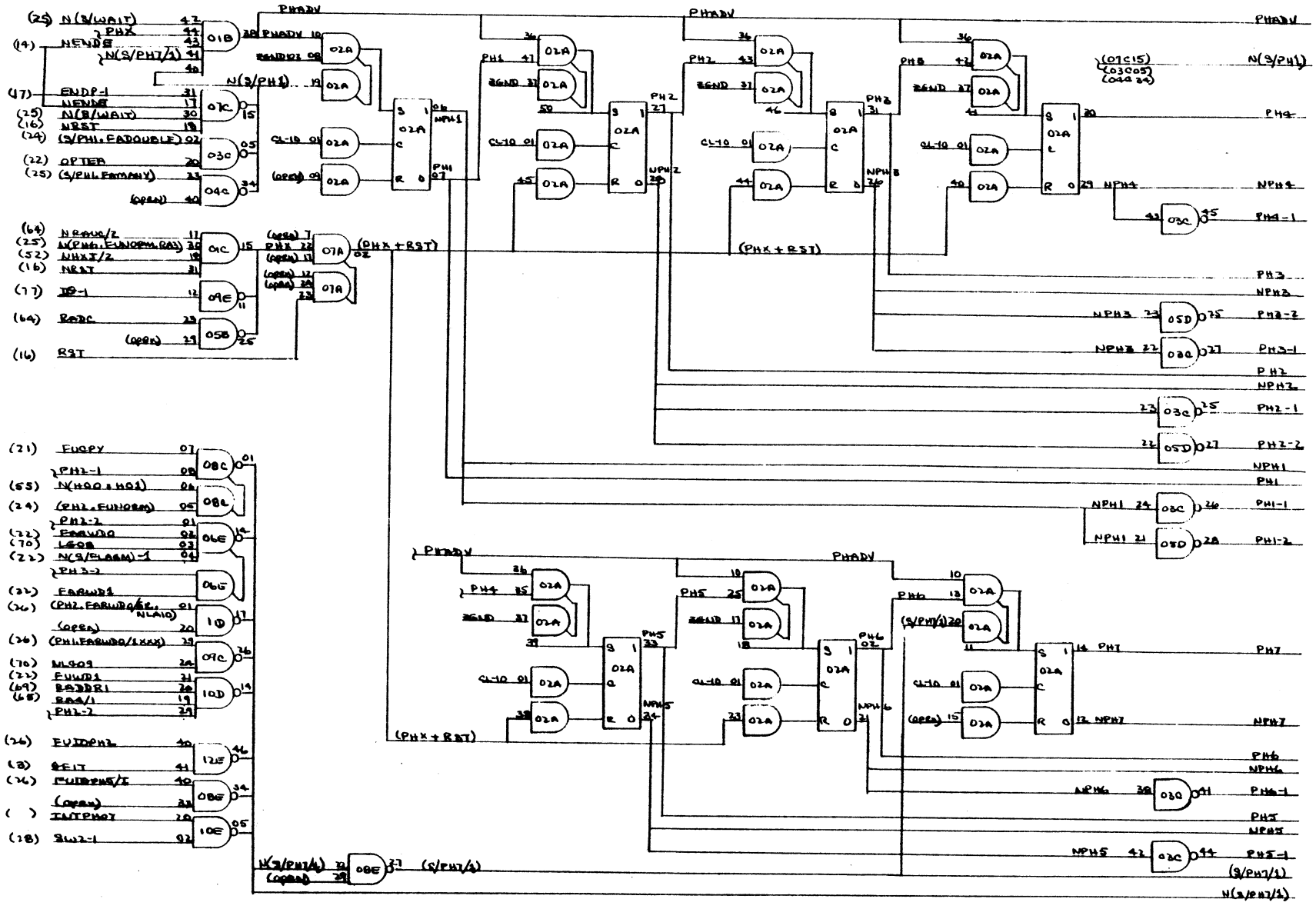


Figure 3-12. Logic Diagrams (sheet 23 of 91)

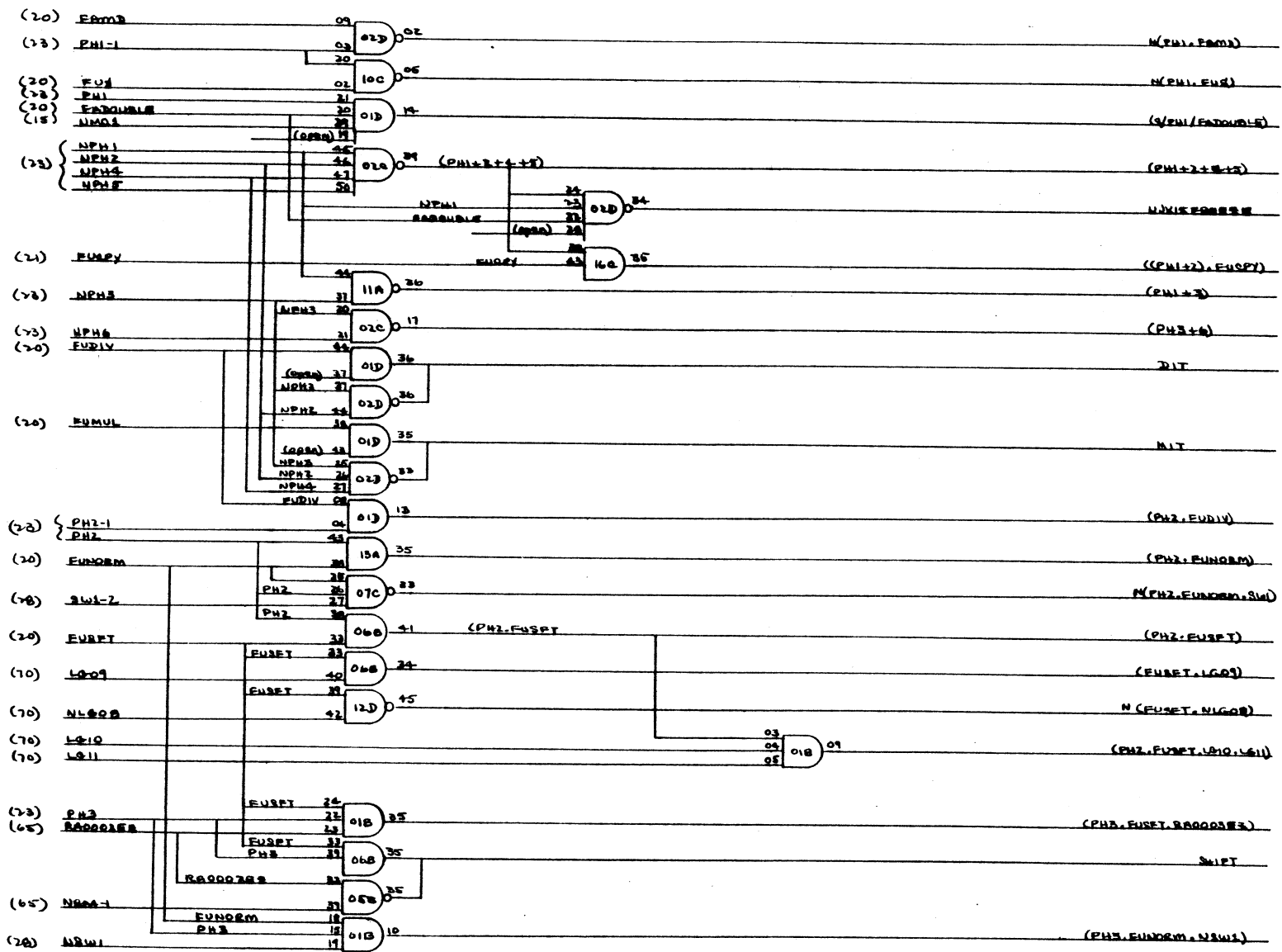


Figure 3-12. Logic Diagrams (sheet 24 of 91)

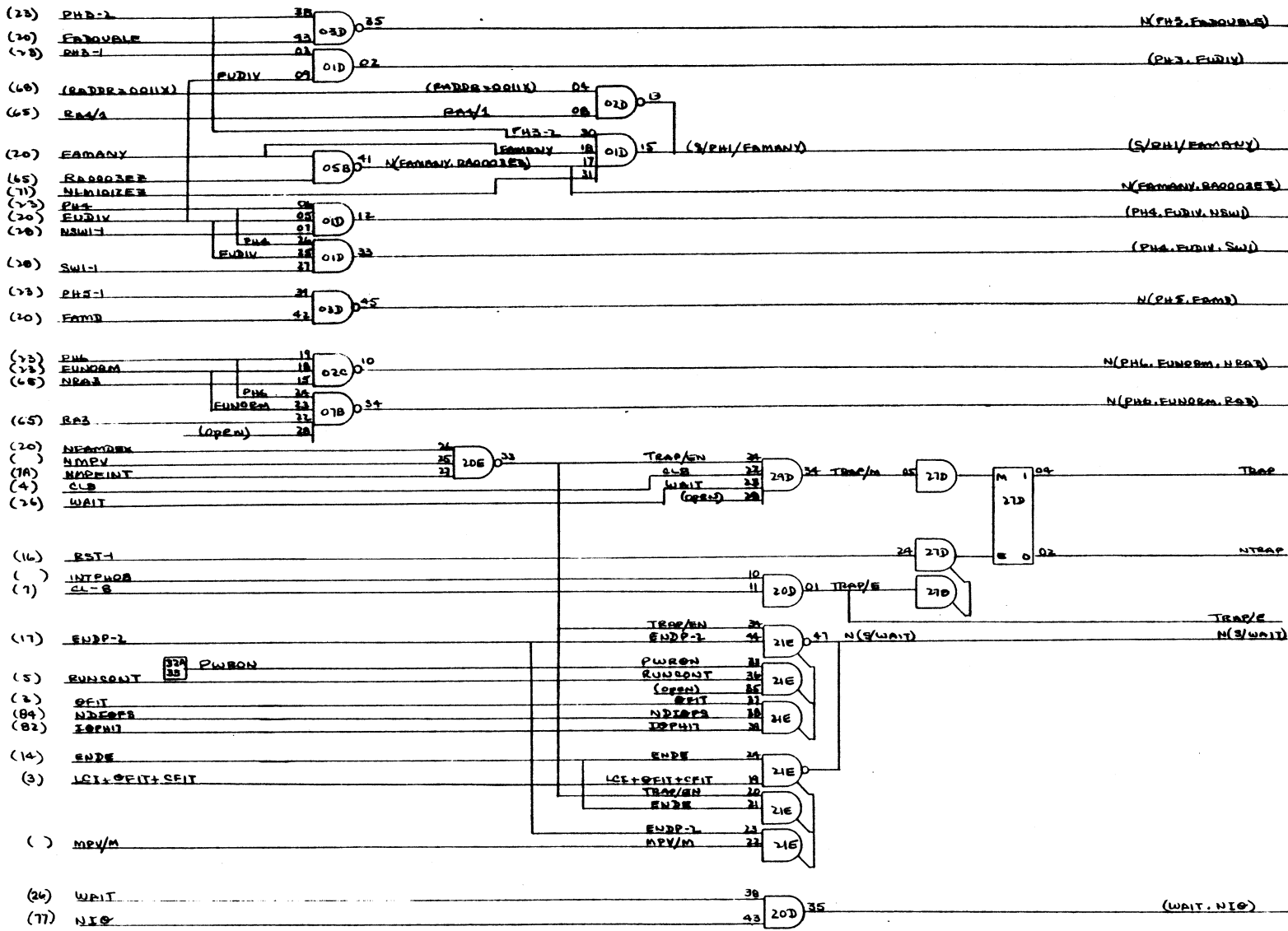


Figure 3-12. Logic Diagrams (sheet 25 of 91)

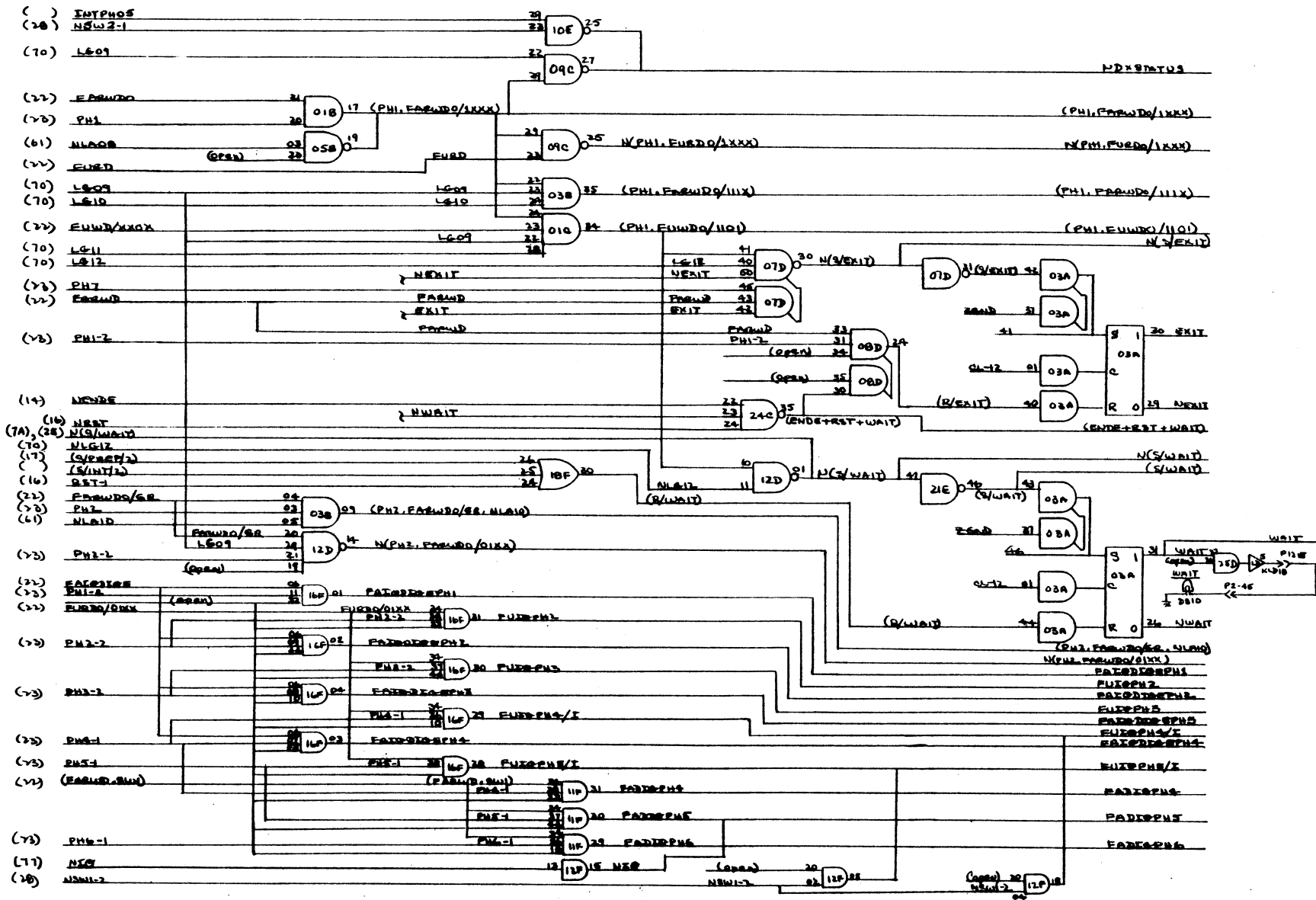


Figure 3-12. Logic Diagrams (sheet 26 of 71)

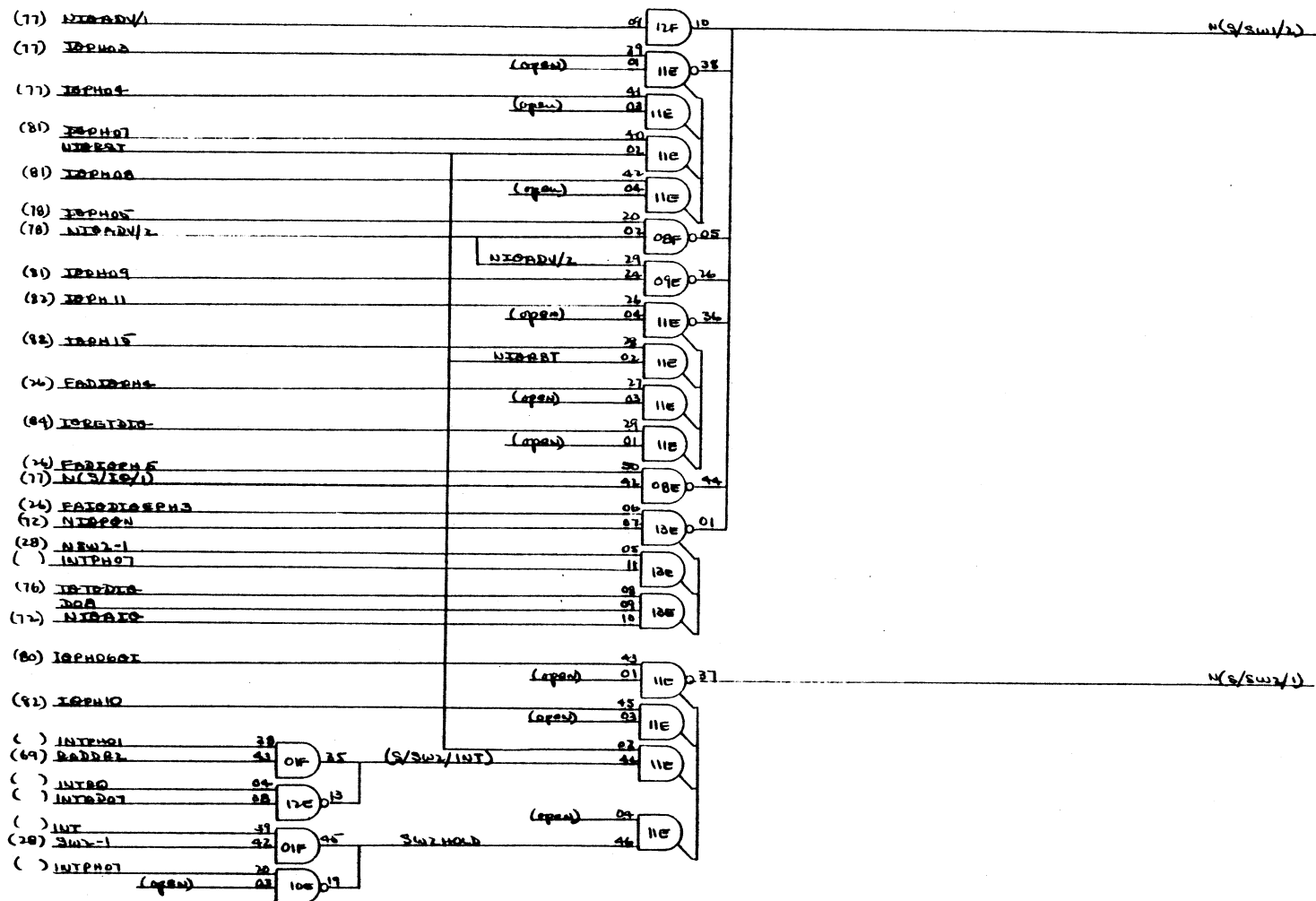


Figure 3-12. Logic Diagrams (sheet 27 of 91)

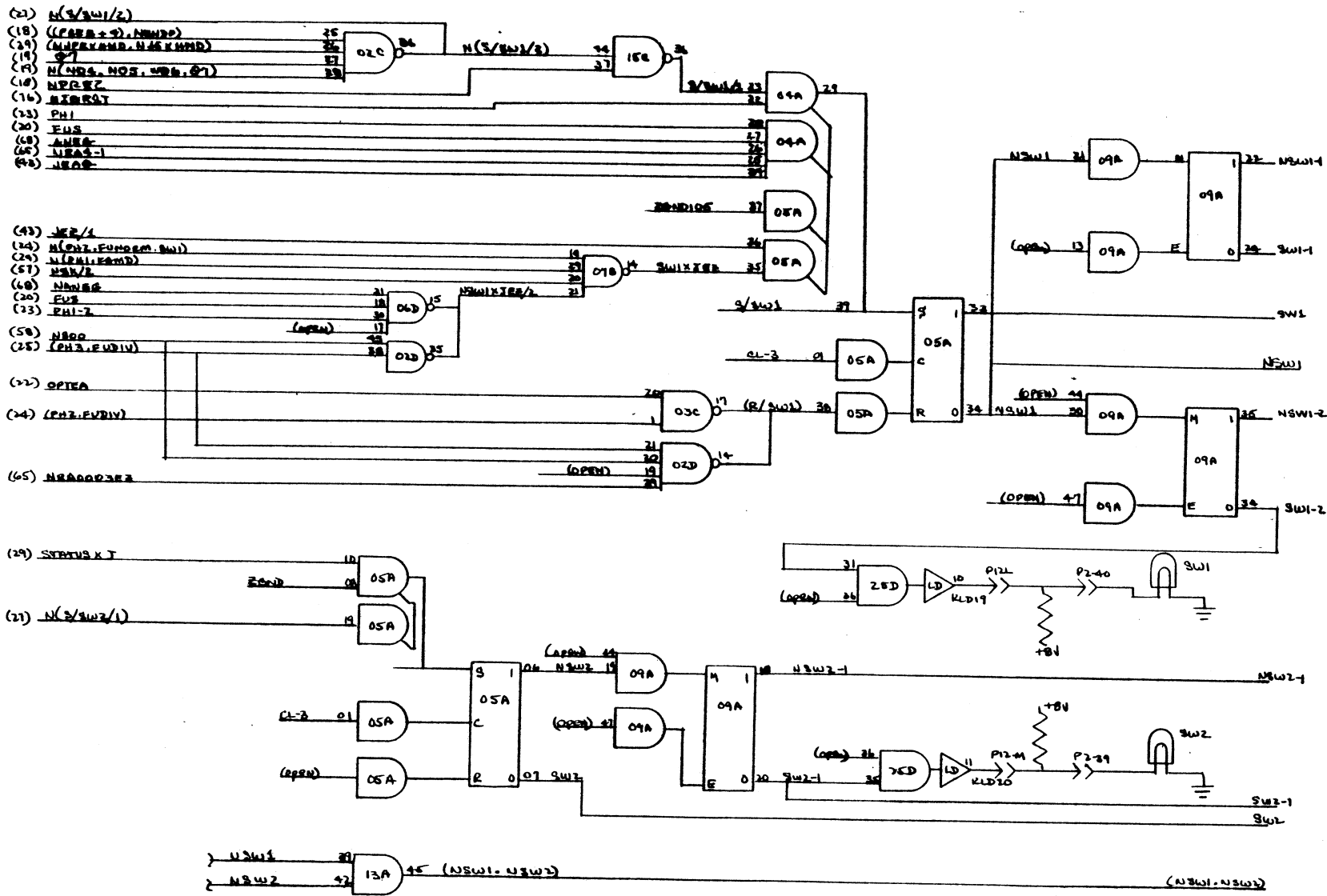


Figure 3-12. Logic Diagrams (sheet 28 of 91)

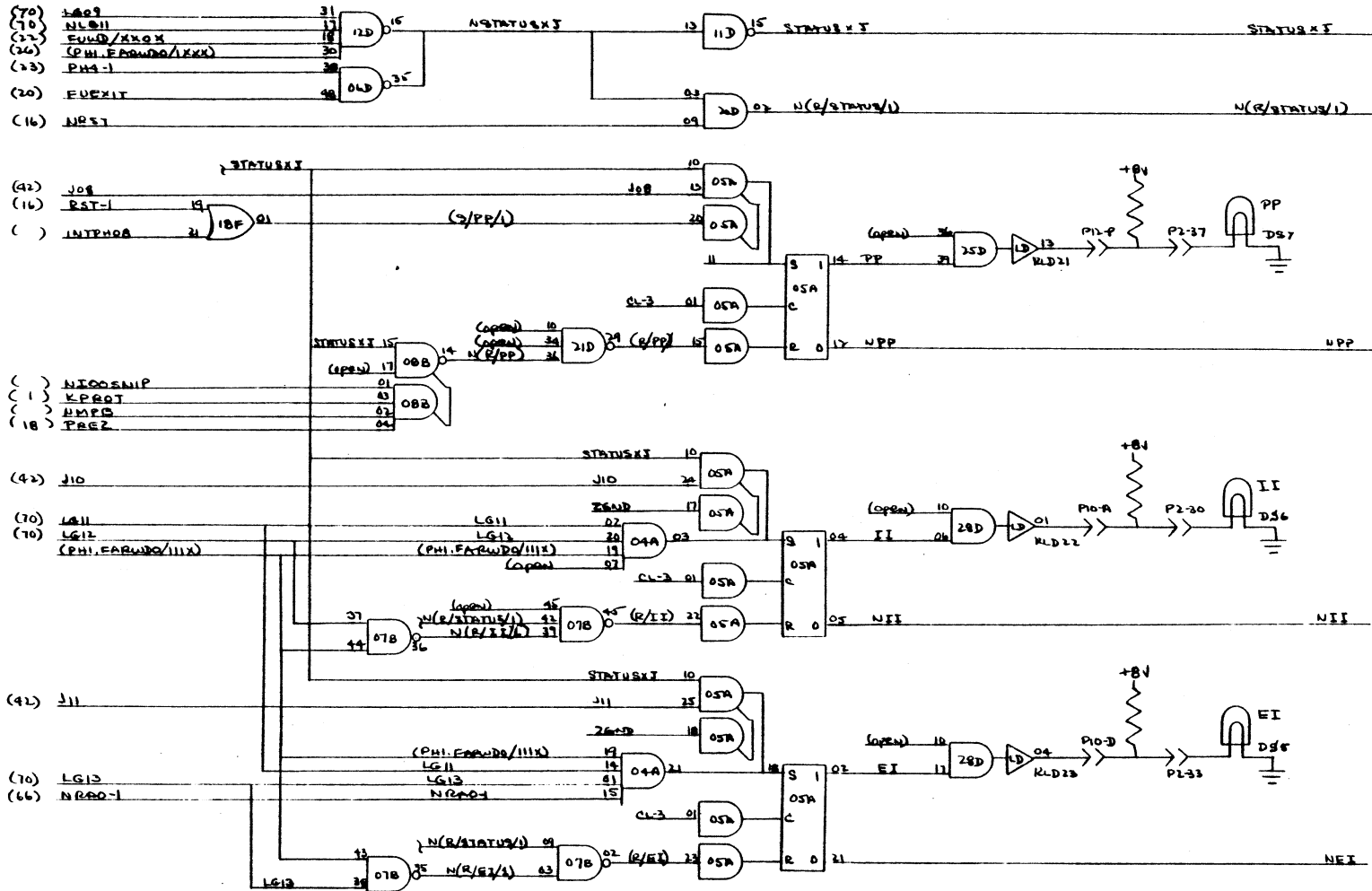


Figure 3-12. Logic Diagrams (sheet 29 of 91)

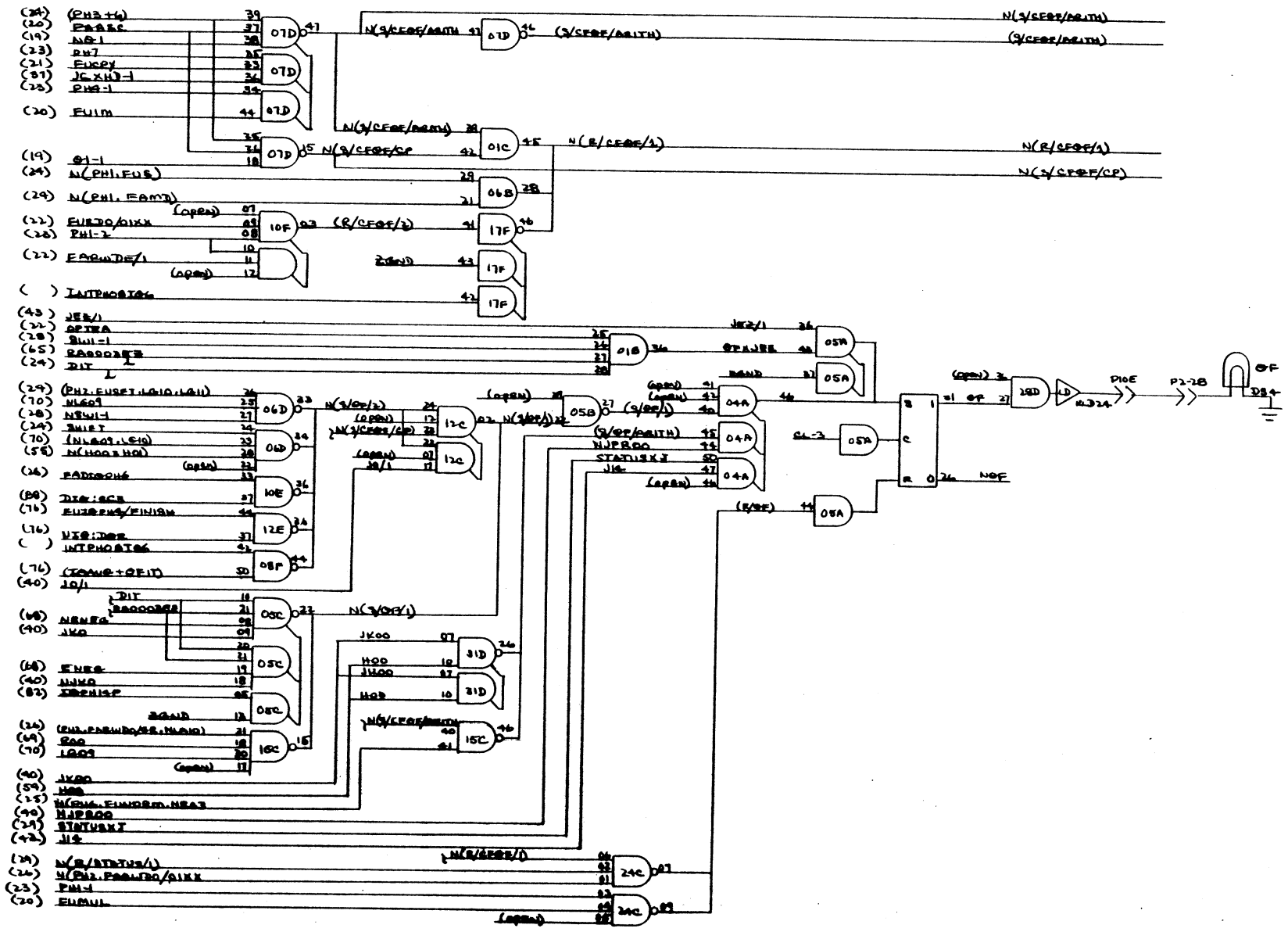


Figure 3-12. Logic Diagrams (sheet 30 of 91)

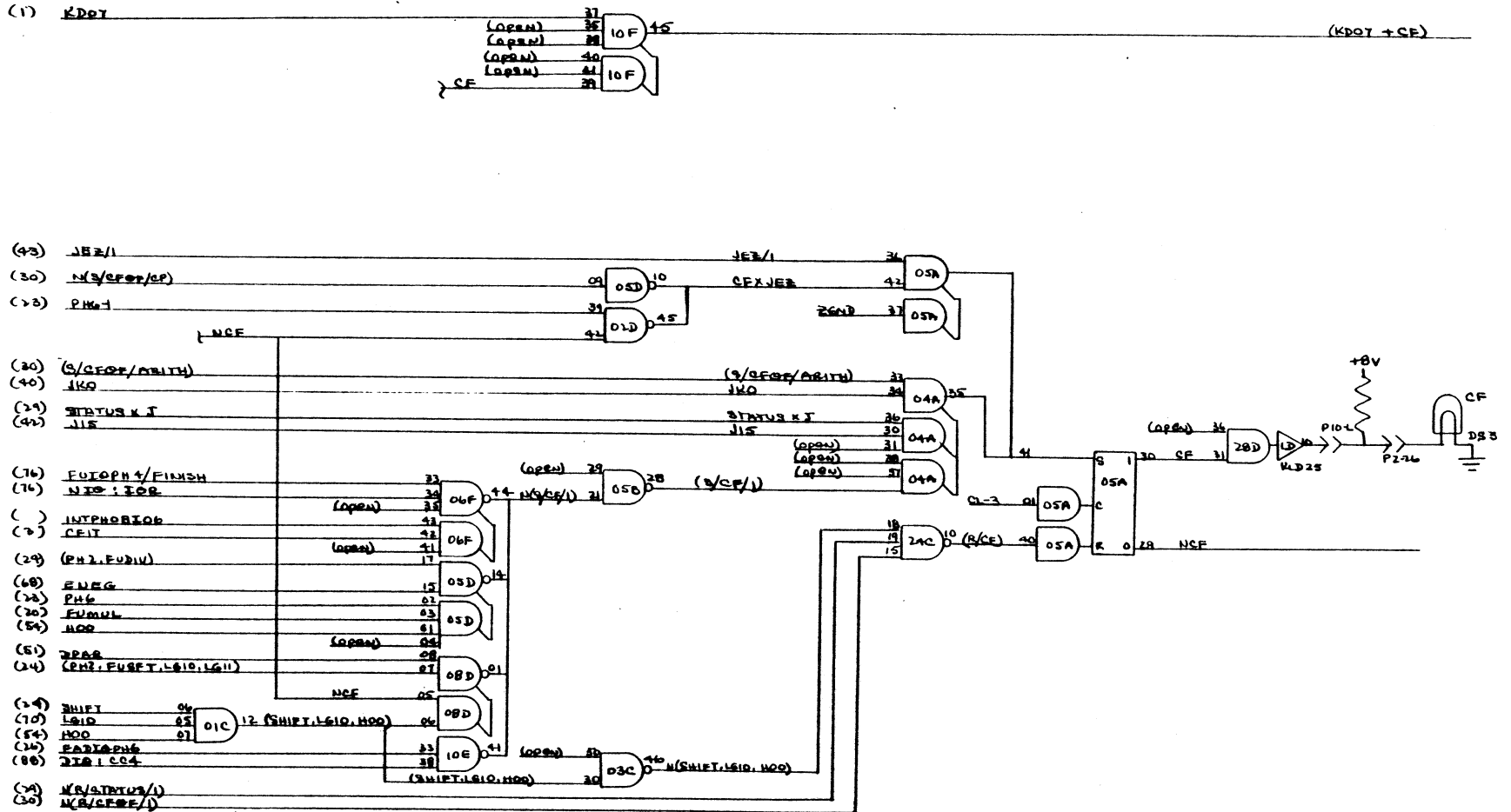


Figure 3-12. Logic Diagrams (sheet 31 of 91)

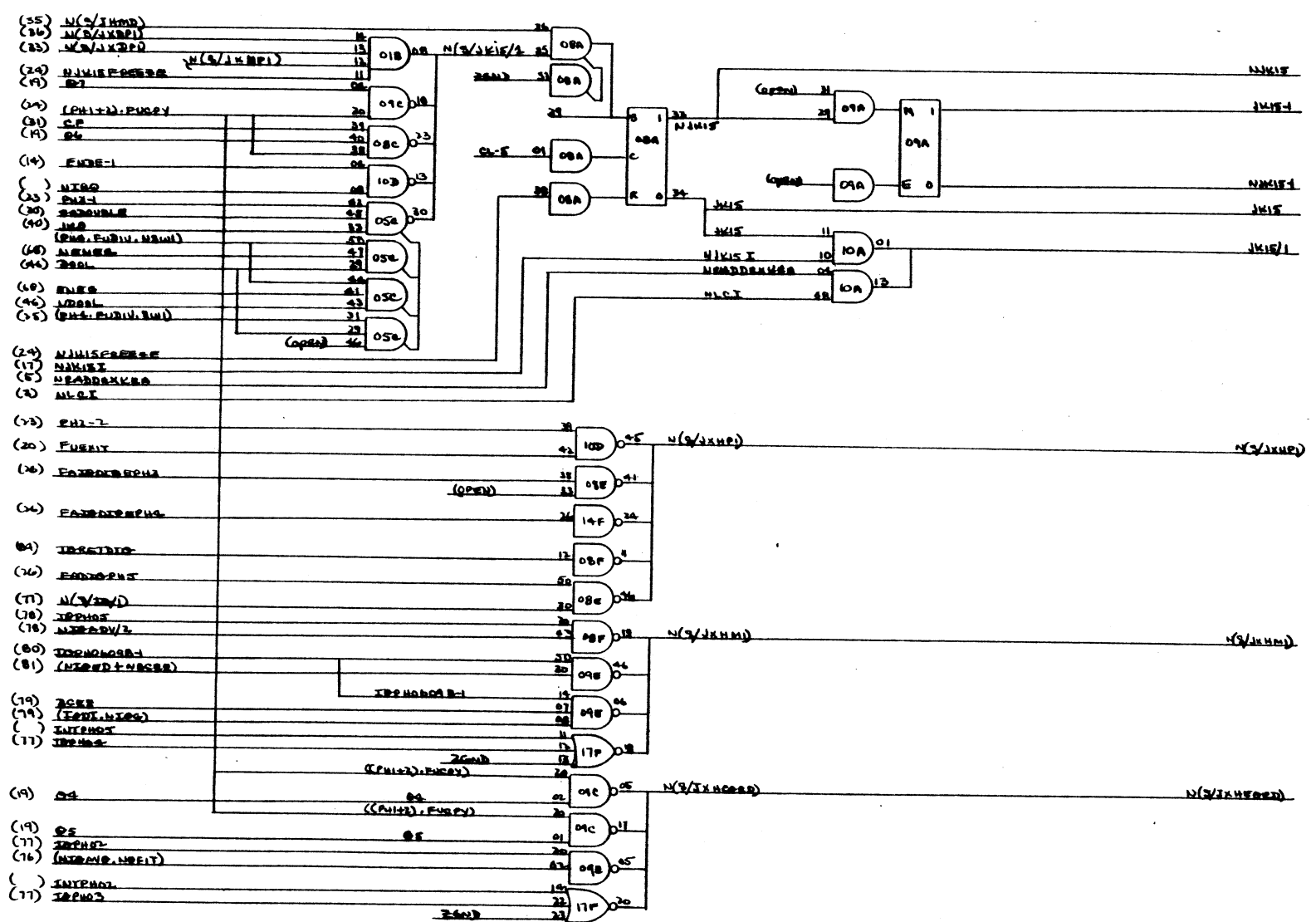


Figure 3-12. Logic Diagrams (sheet 32 of 91)

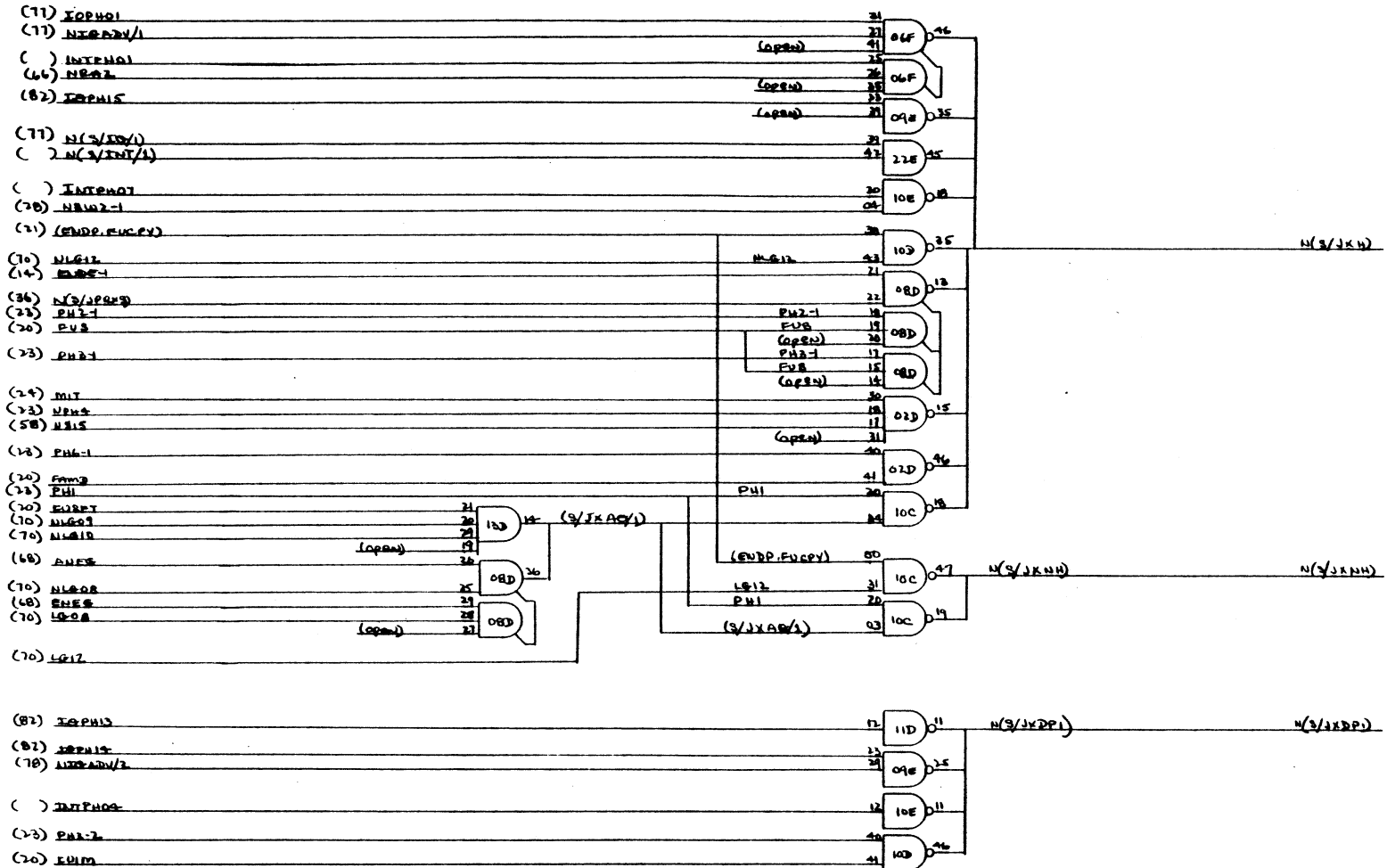


Figure 3-12. Logic Diagrams (sheet 33 of 91)

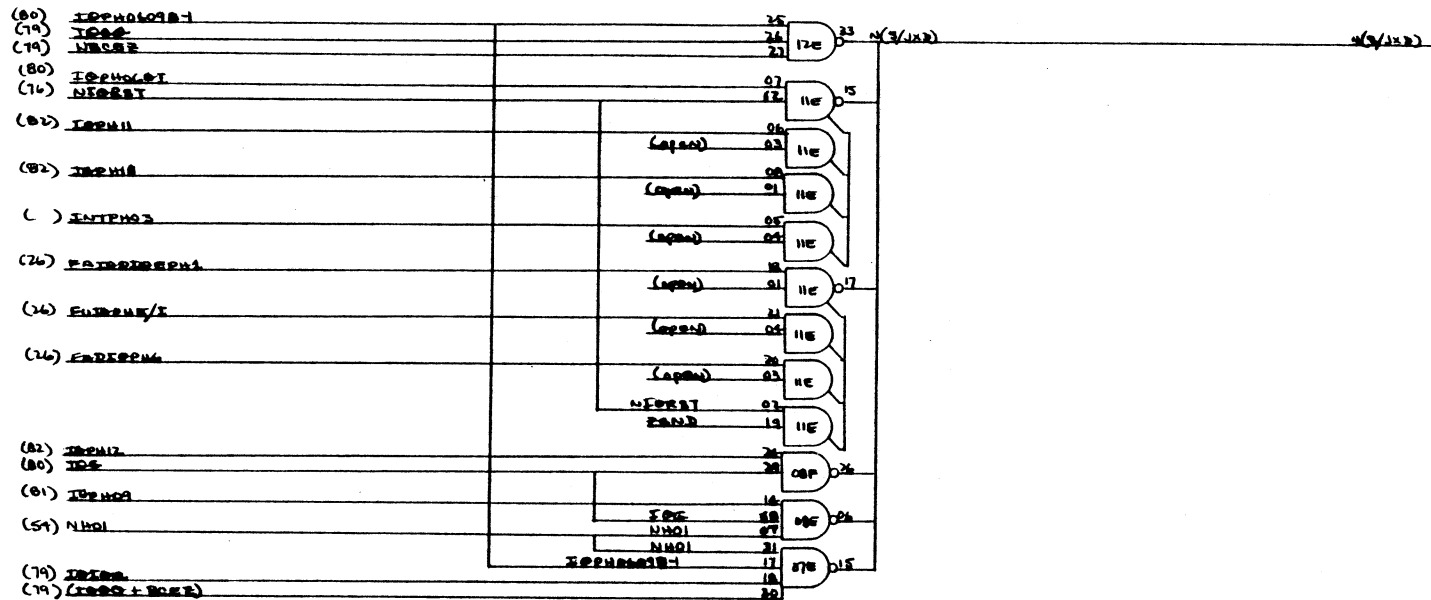


Figure 3-12. Logic Diagrams (sheet 34 of 91)

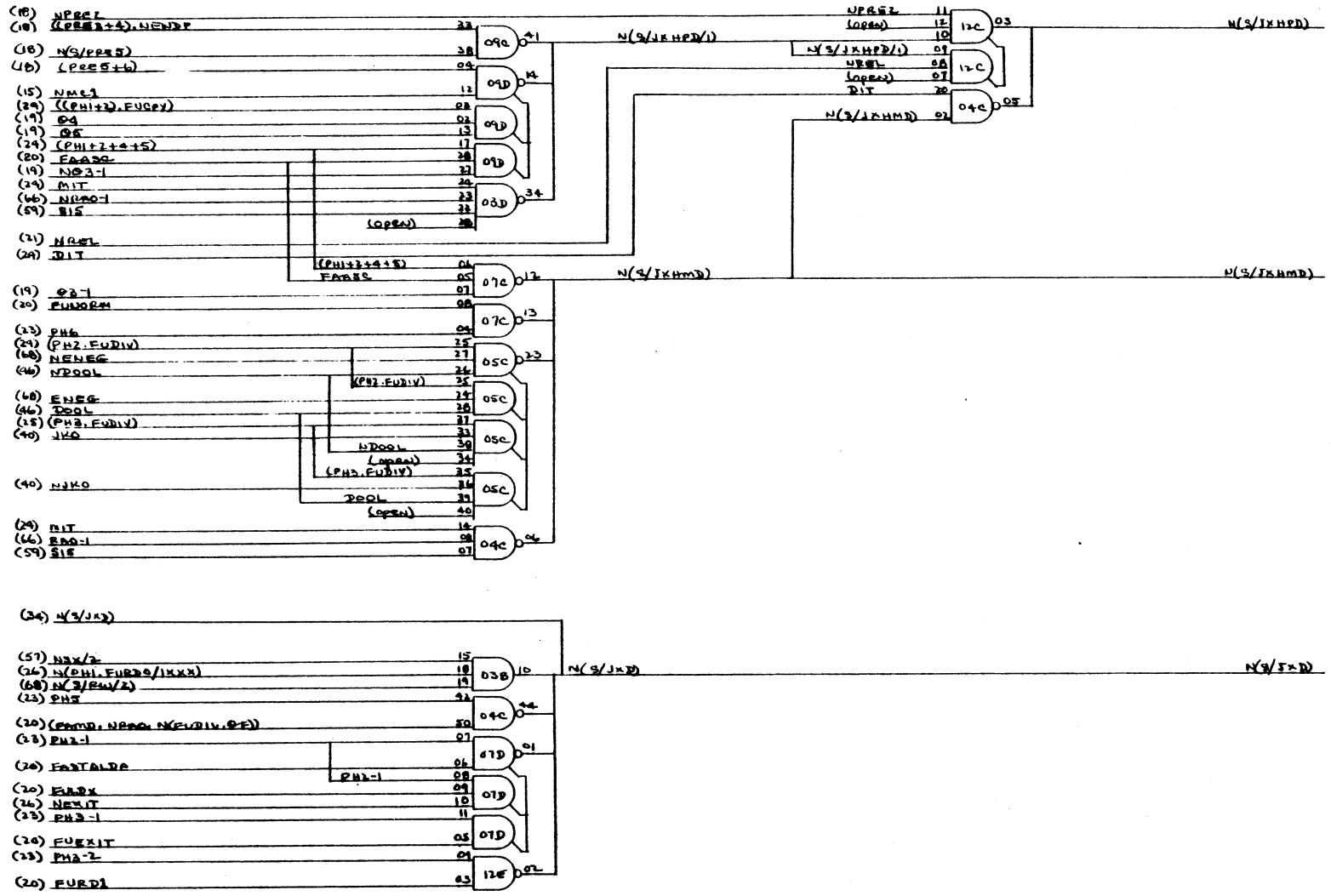


Figure 3-12. Logic Diagrams (sheet 35 of 91)

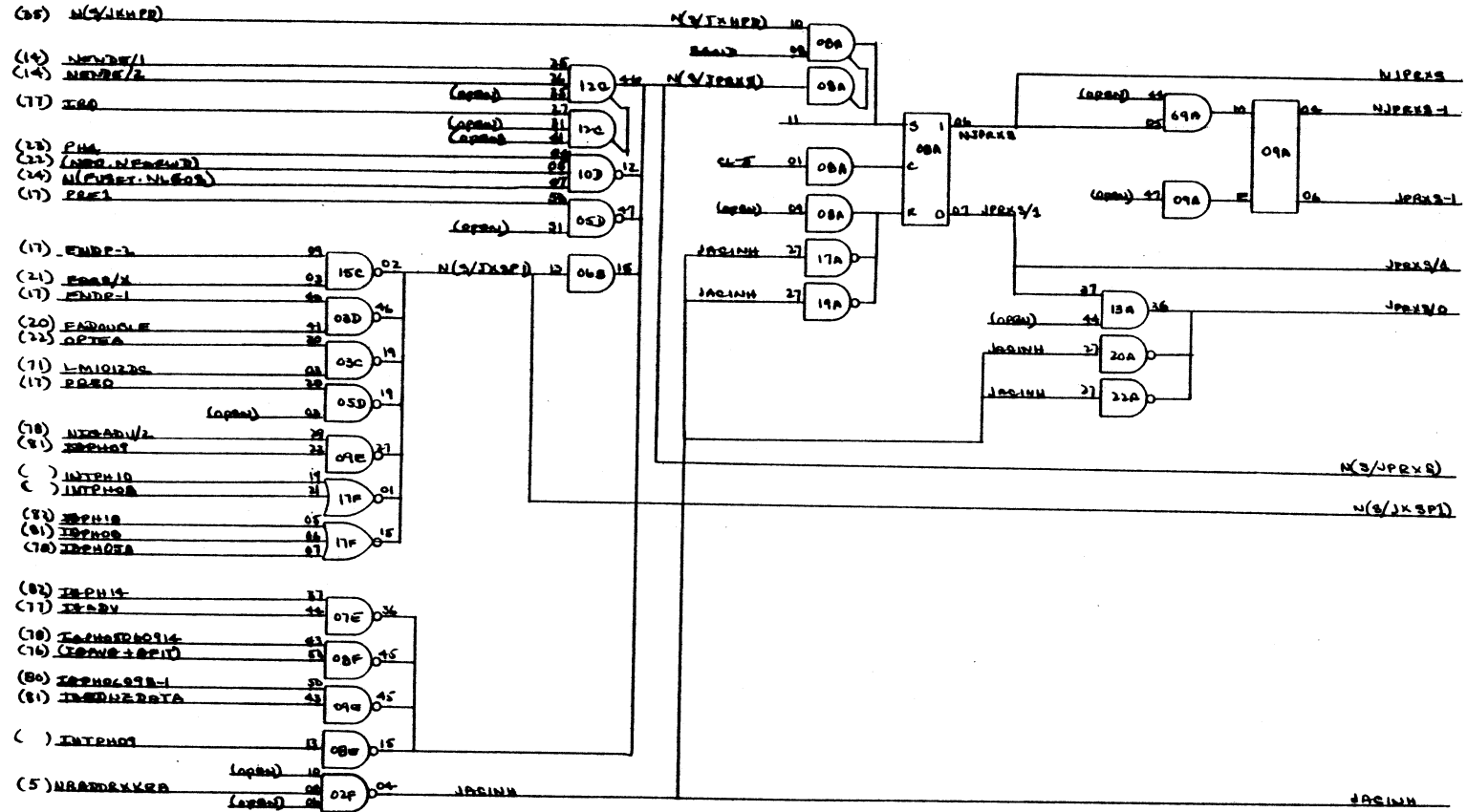


Figure 3-12. Logic Diagrams (sheet 36 of 91)

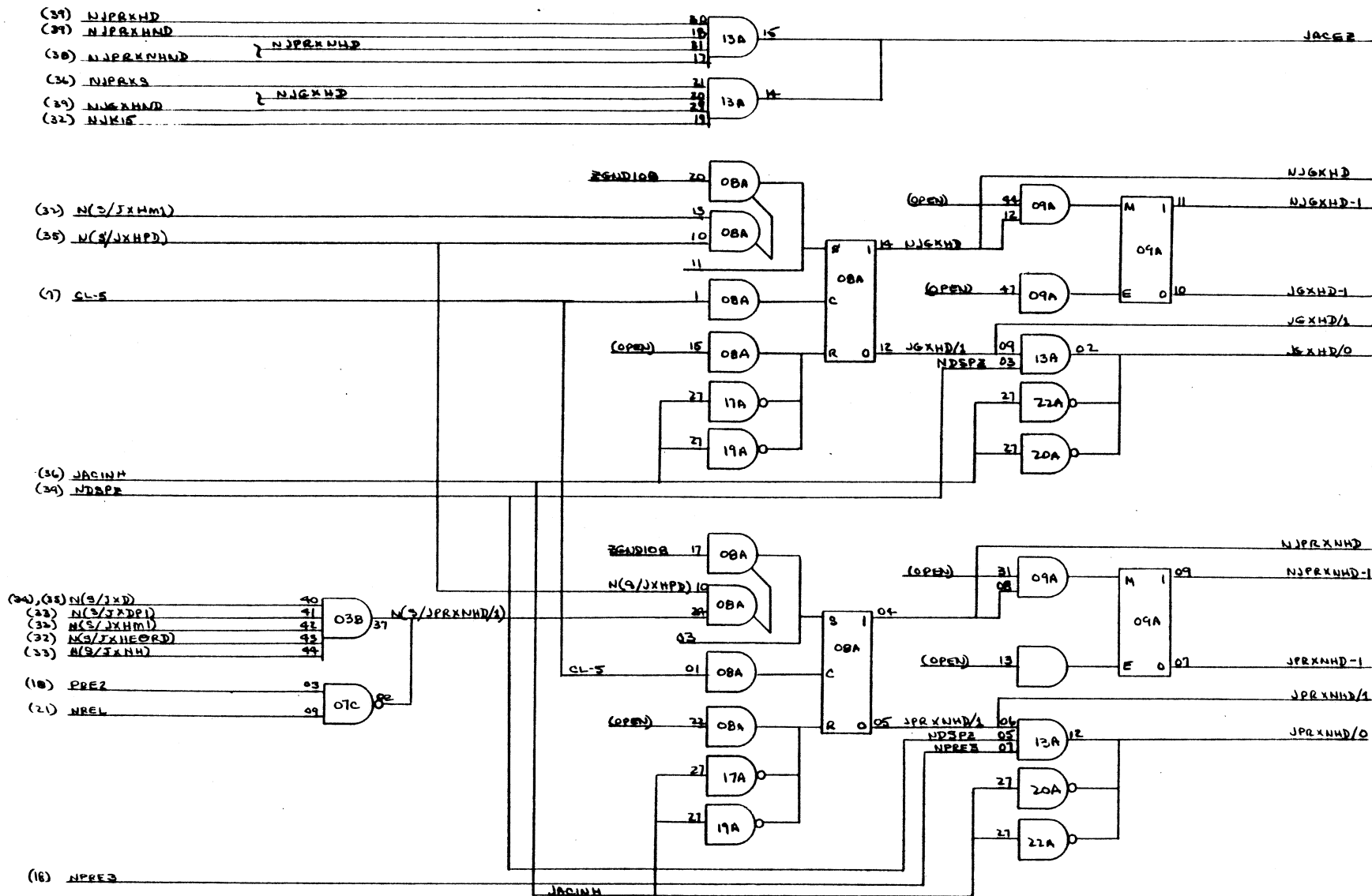
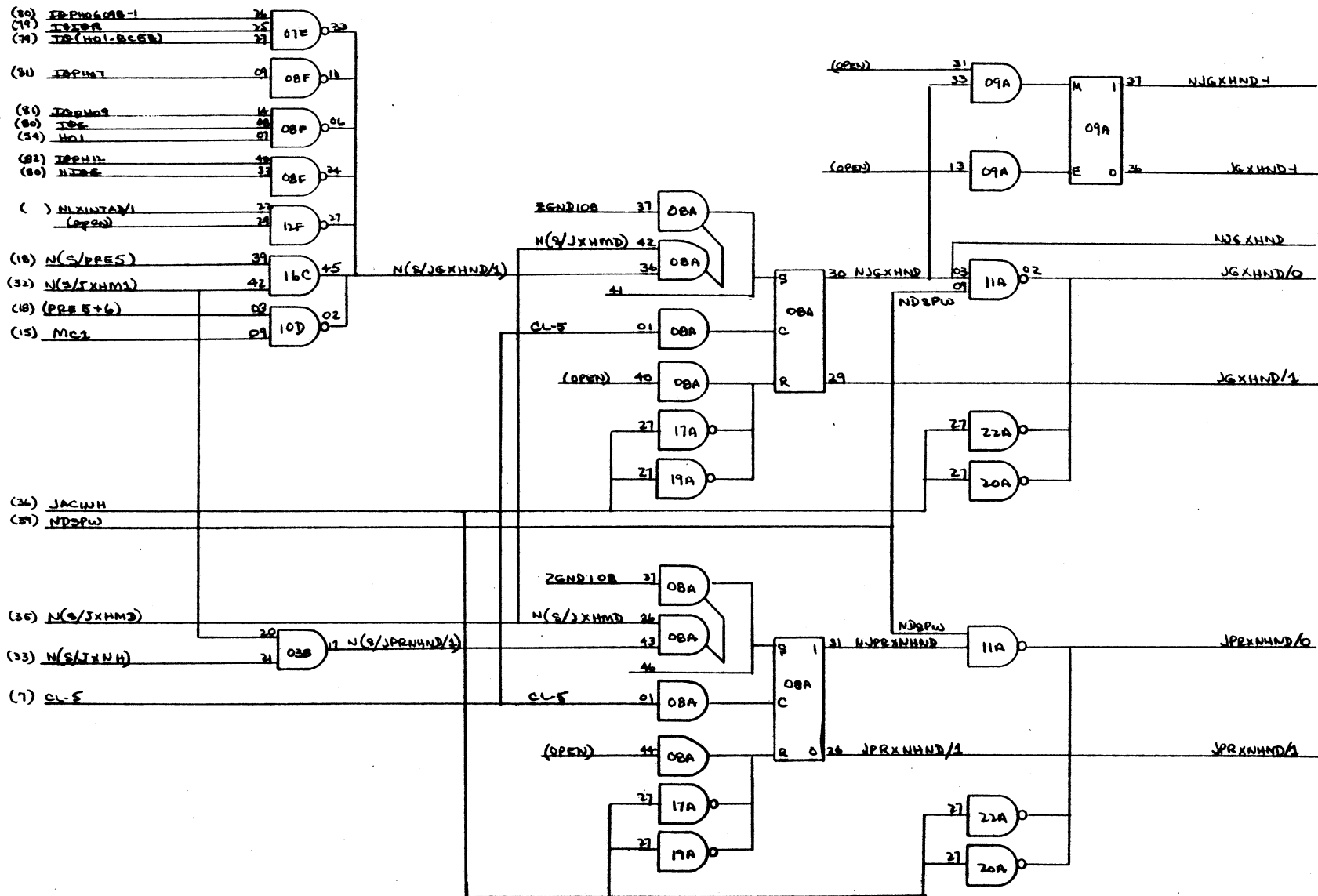


Figure 3-12. Logic Diagrams (sheet 37 of 91)



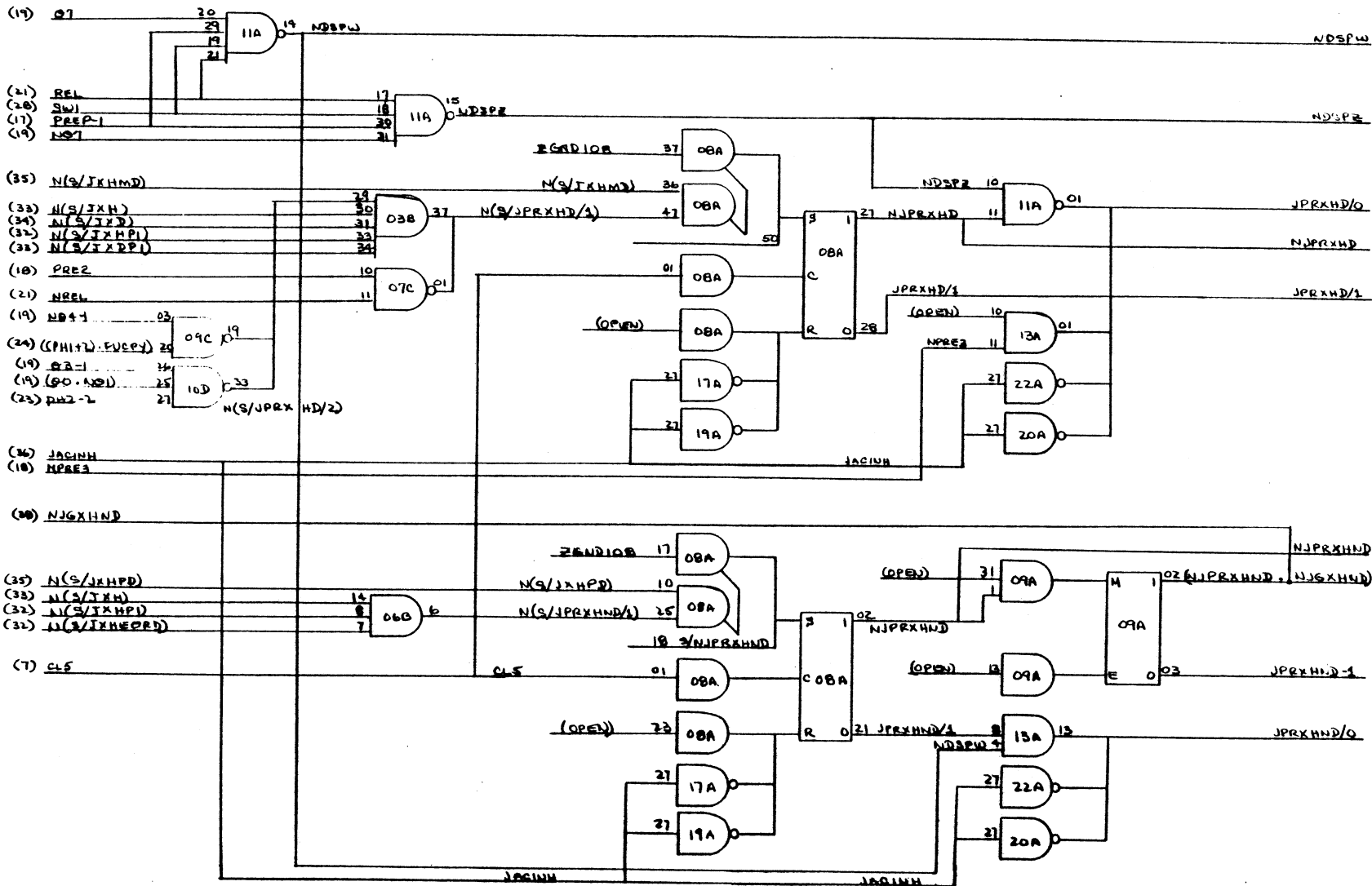


Figure 3-12. Logic Diagrams (sheet 39 of 91)

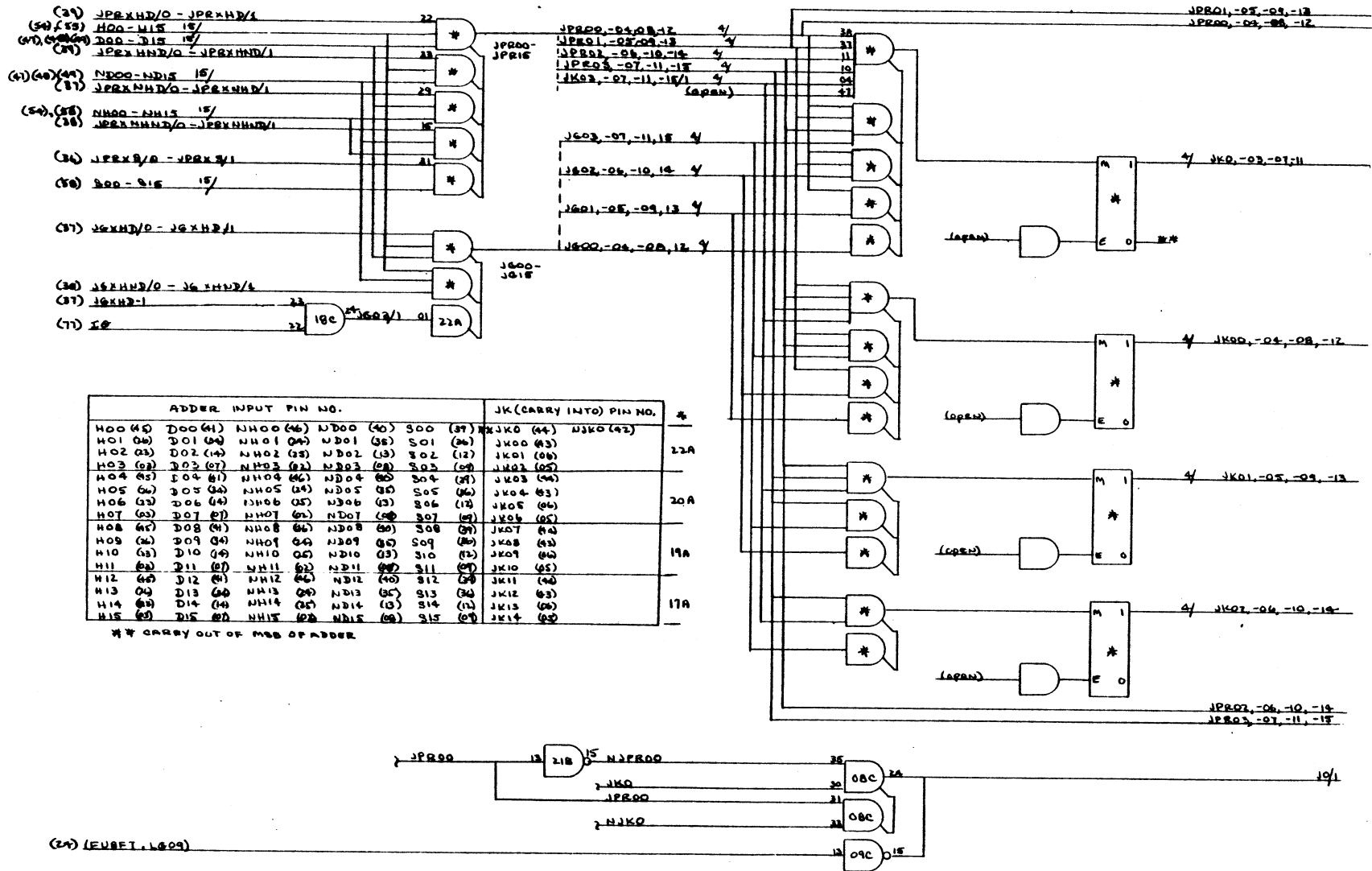


Figure 3-12. Logic Diagrams (sheet 40 of 71)

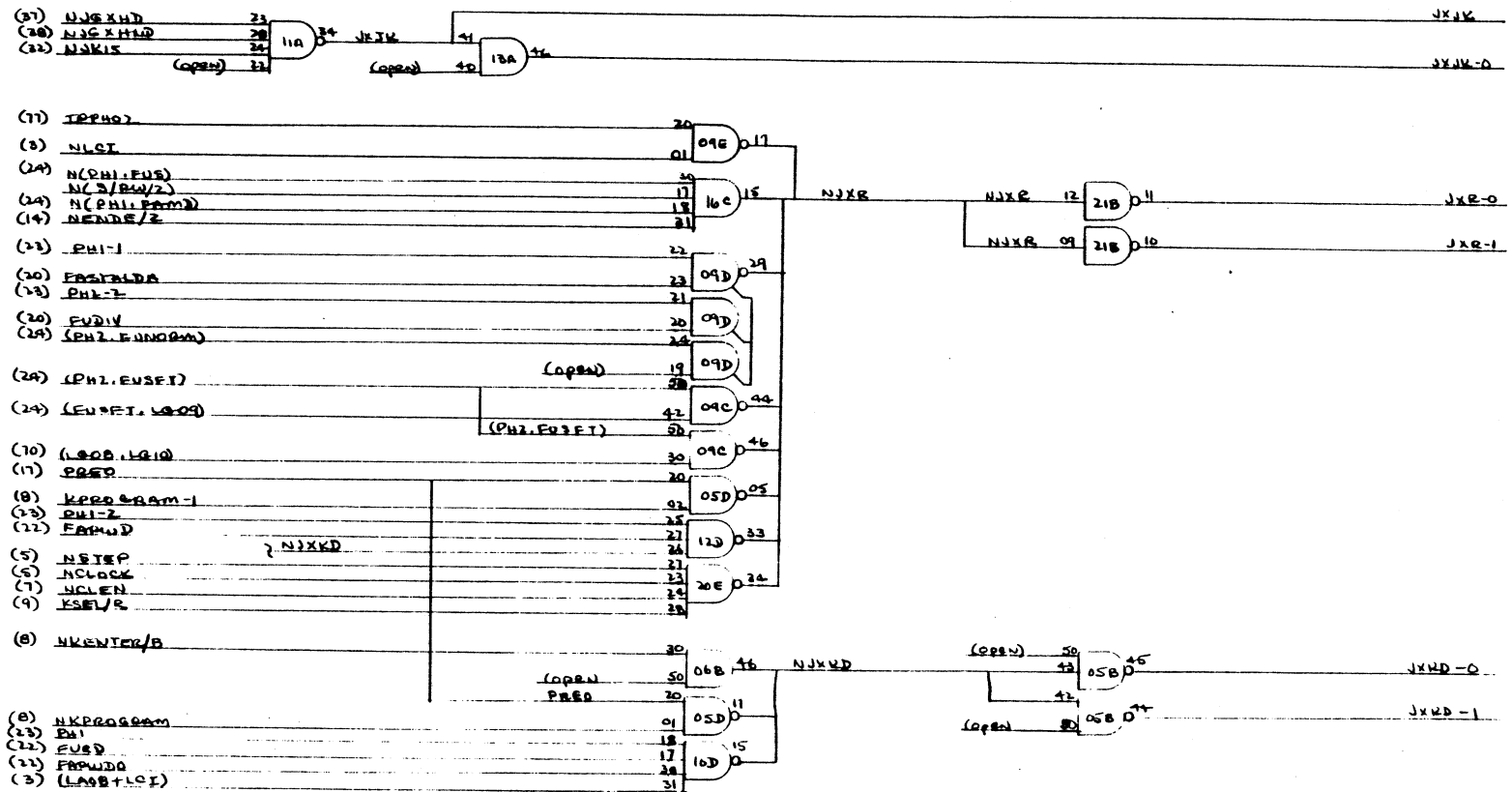
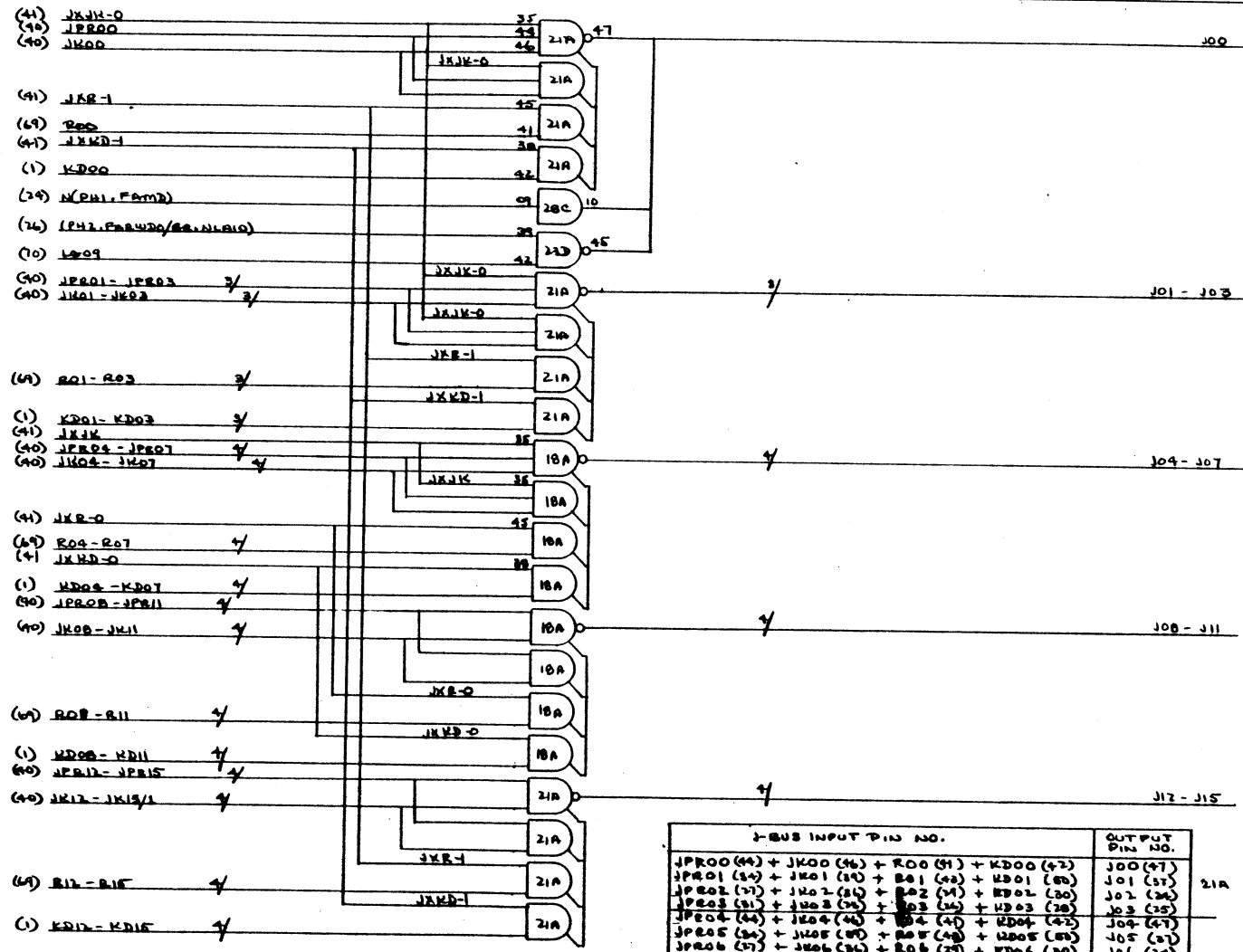


Figure 3-12. Logic Diagrams (sheet 41 of 91)



3-BUS INPUT PIN NO.	OUTPUT PIN NO.
JPR00 (44) + JK00 (46) + R00 (41) + K000 (42)	J00 (47)
JPR01 (45) + JH01 (48) + R01 (43) + K001 (49)	J01 (48)
JPR02 (51) + JH02 (54) + R02 (51) + K002 (50)	J02 (50)
JPR03 (51) + JH03 (55) + R03 (52) + K003 (58)	J03 (55)
JPR04 (44) + JH04 (45) + R04 (47) + K004 (47)	J04 (47)
JPR05 (54) + JH05 (57) + R05 (48) + K005 (58)	J05 (57)
JPR06 (57) + JH06 (56) + R06 (57) + K006 (50)	J06 (56)
JPR07 (51) + JH07 (57) + R07 (54) + K007 (58)	J07 (57)
JPR08 (12) + JH08 (08) + R08 (04) + K008 (12)	J08 (12)
JPR09 (18) + JH09 (09) + R09 (05) + K009 (11)	J09 (18)
JPR10 (11) + JH10 (07) + R10 (08) + K010 (07)	J10 (10)
JPR11 (06) + JH11 (01) + R11 (03) + K011 (08)	J11 (08)
JPR12 (12) + JH12 (09) + R12 (10) + K012 (07)	J12 (17)
JPR13 (18) + JH13 (09) + R13 (05) + K013 (01)	J13 (05)
JPR14 (11) + JH14 (07) + R14 (05) + K014 (05)	J14 (10)
JPR15 (06) + JH15 (01) + R15 (04) + K015 (08)	J15 (10)

Figure 3-12. Logic Diagrams (sheet 42 of 71)

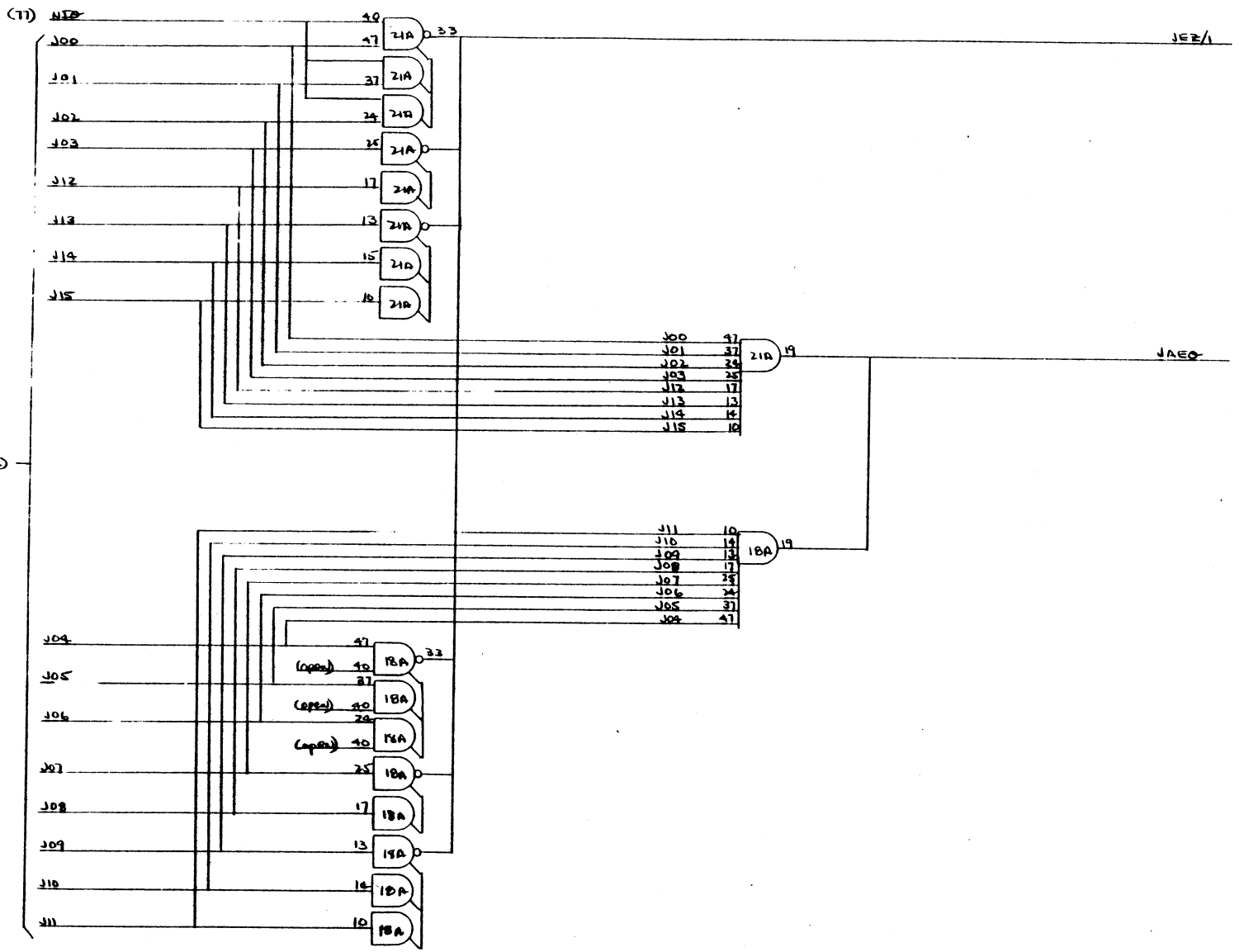


Figure 3-12. Logic Diagrams (sheet 43 of 91)

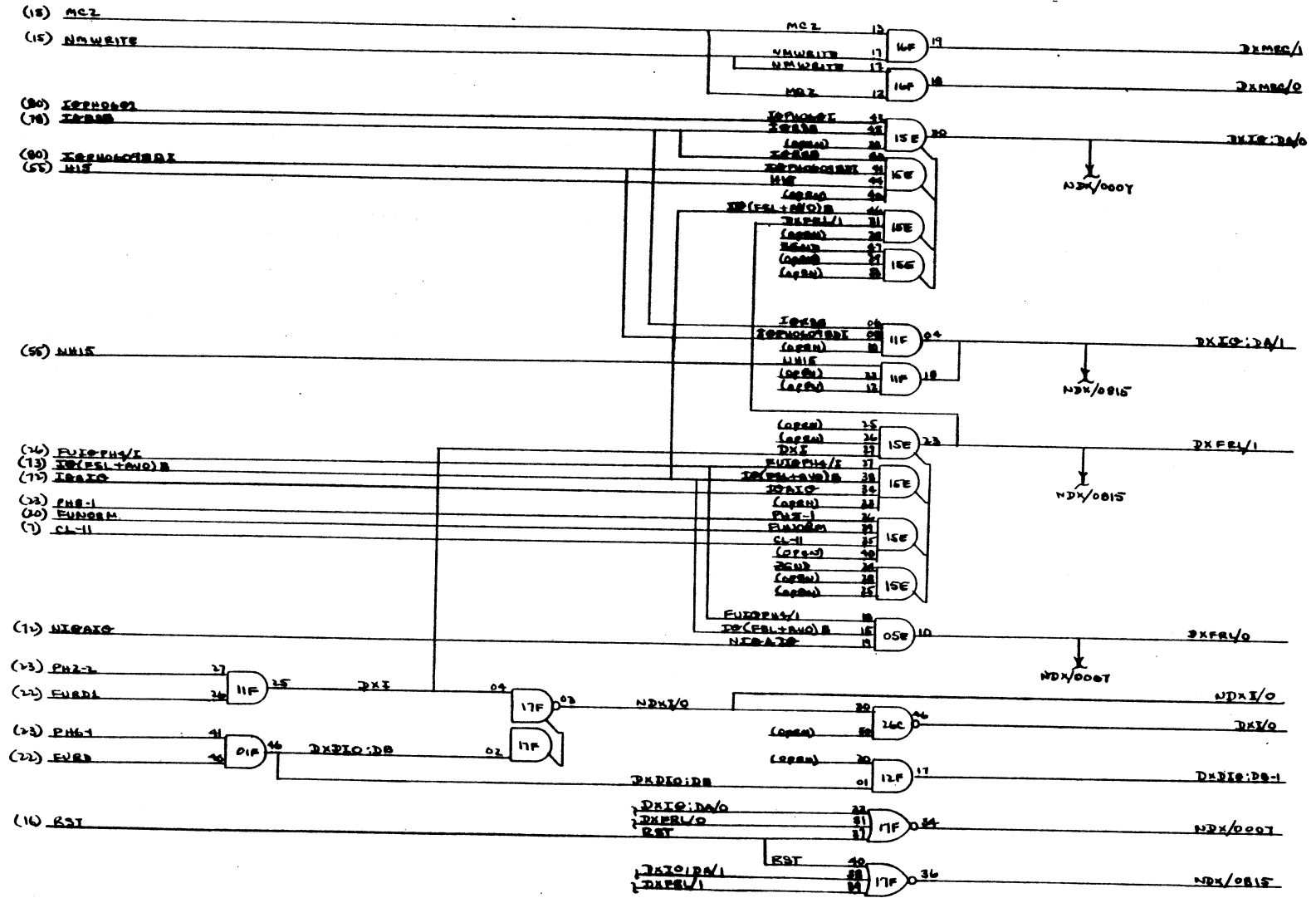


Figure 3-12. Logic Diagrams (sheet 44 of 71)

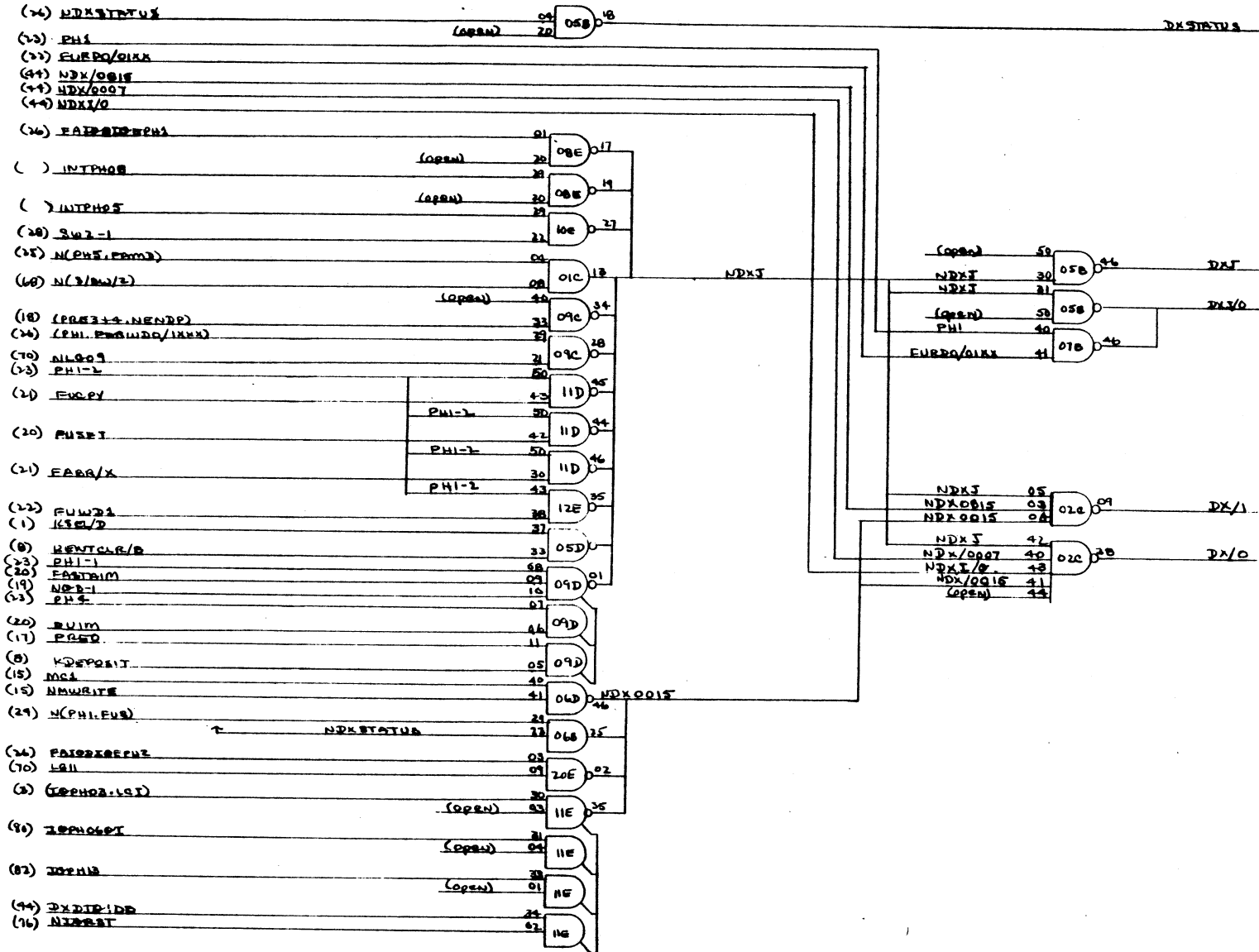


Figure 3-12. Logic Diagrams (sheet 45 of 91)

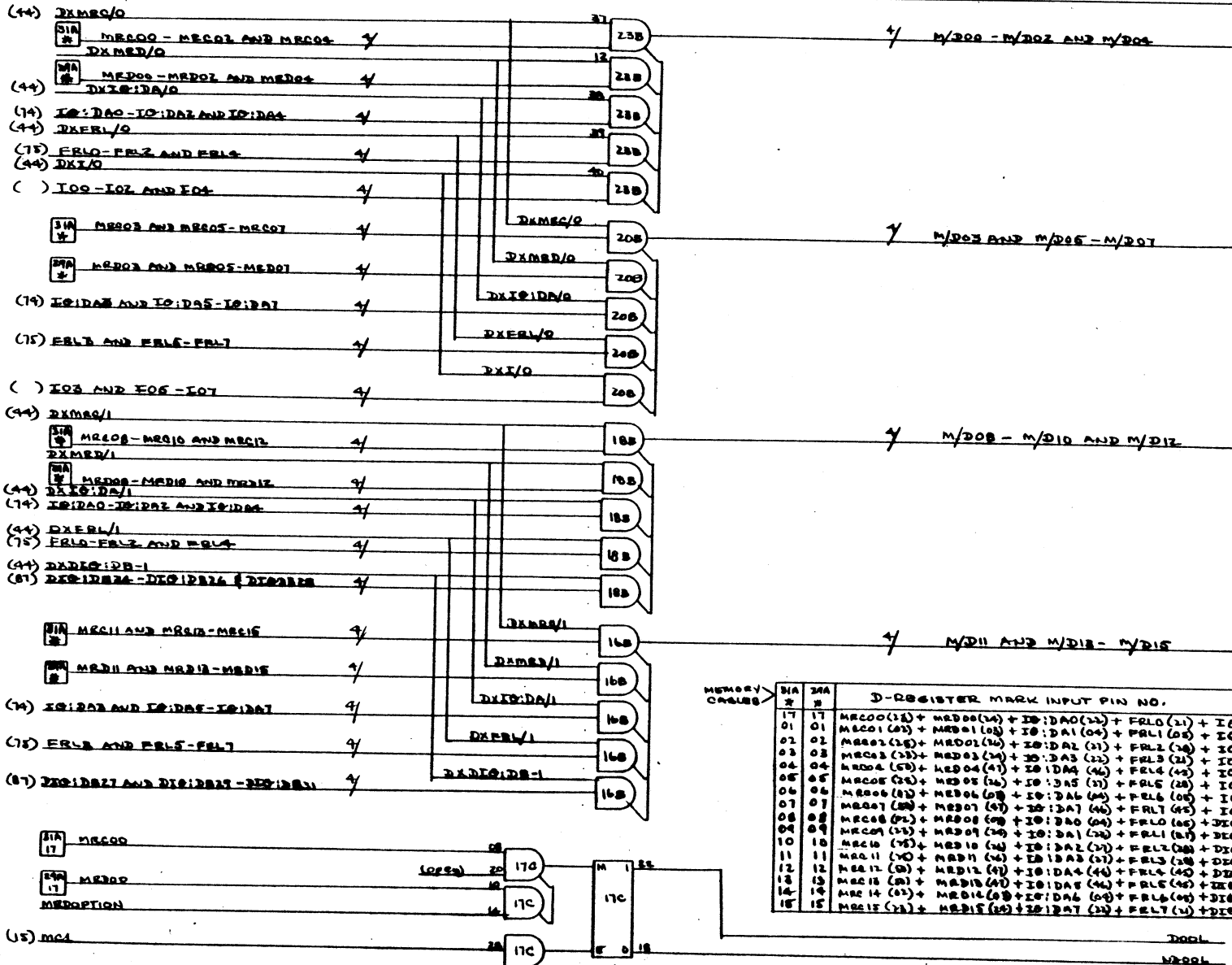


Figure 3-12. Logic Diagrams (sheet 46 of 71)

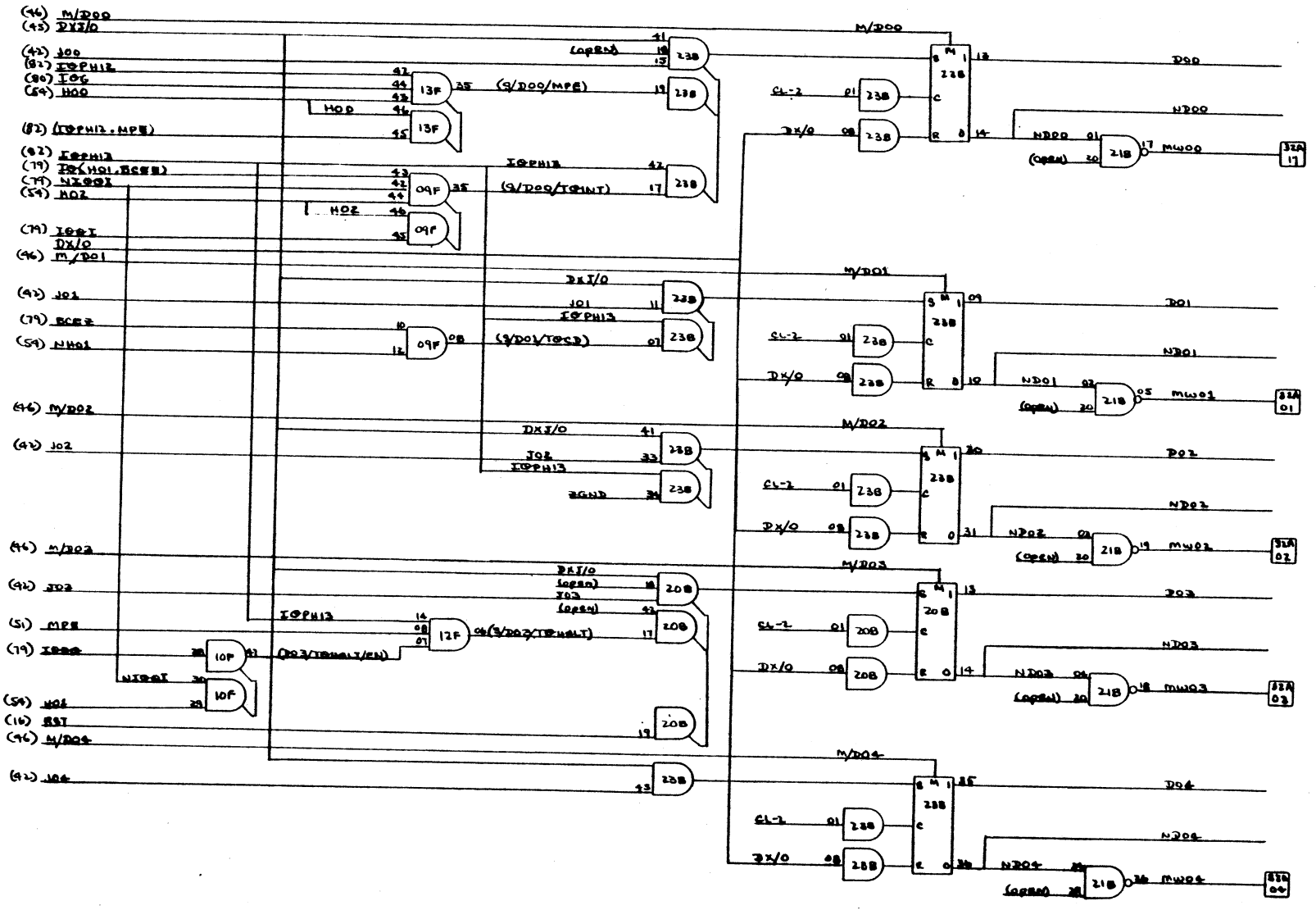


Figure 3-12. Logic Diagrams (sheet 47 of 91)

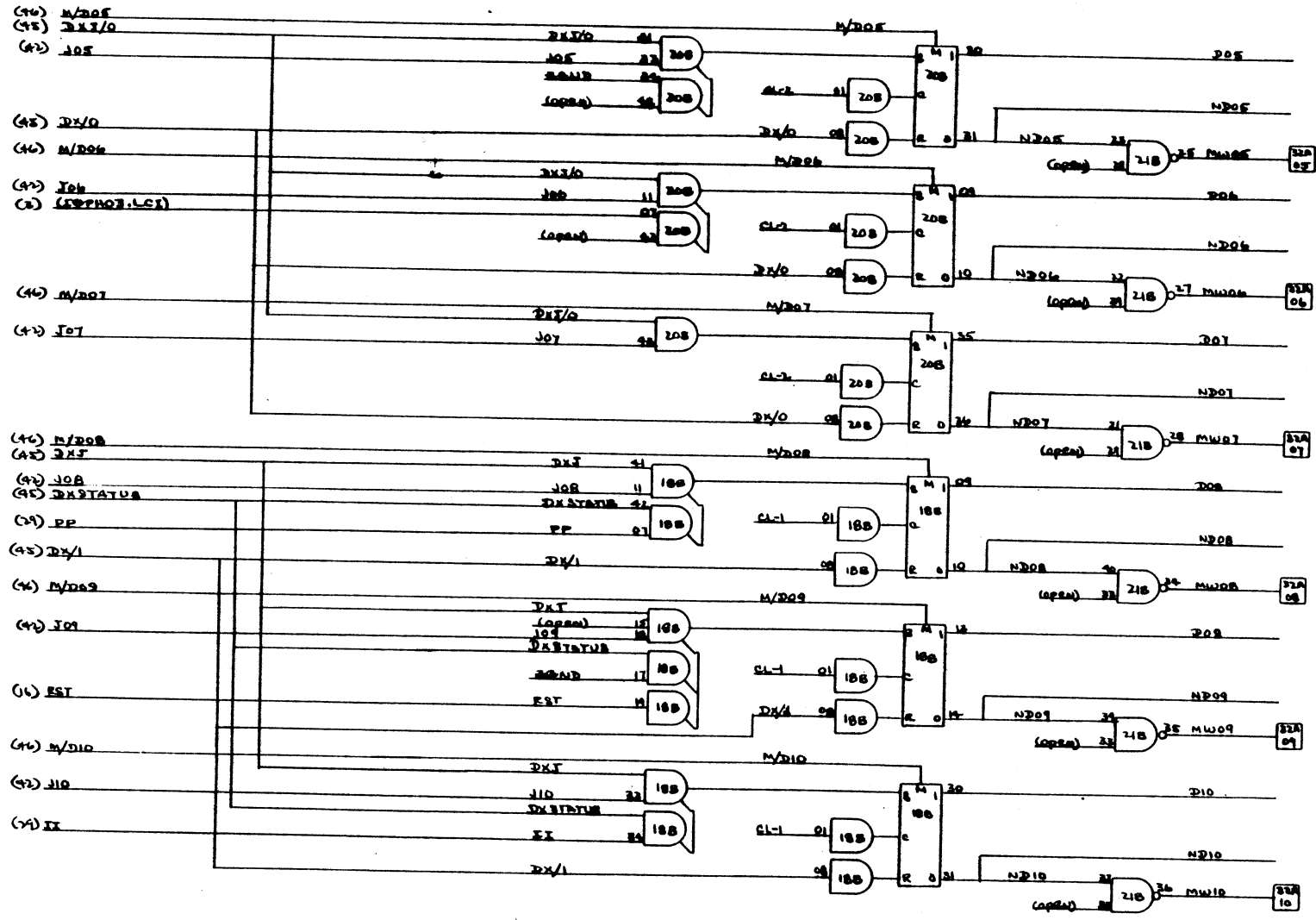


Figure 3-12. Logic Diagrams (sheet 48 of 91)

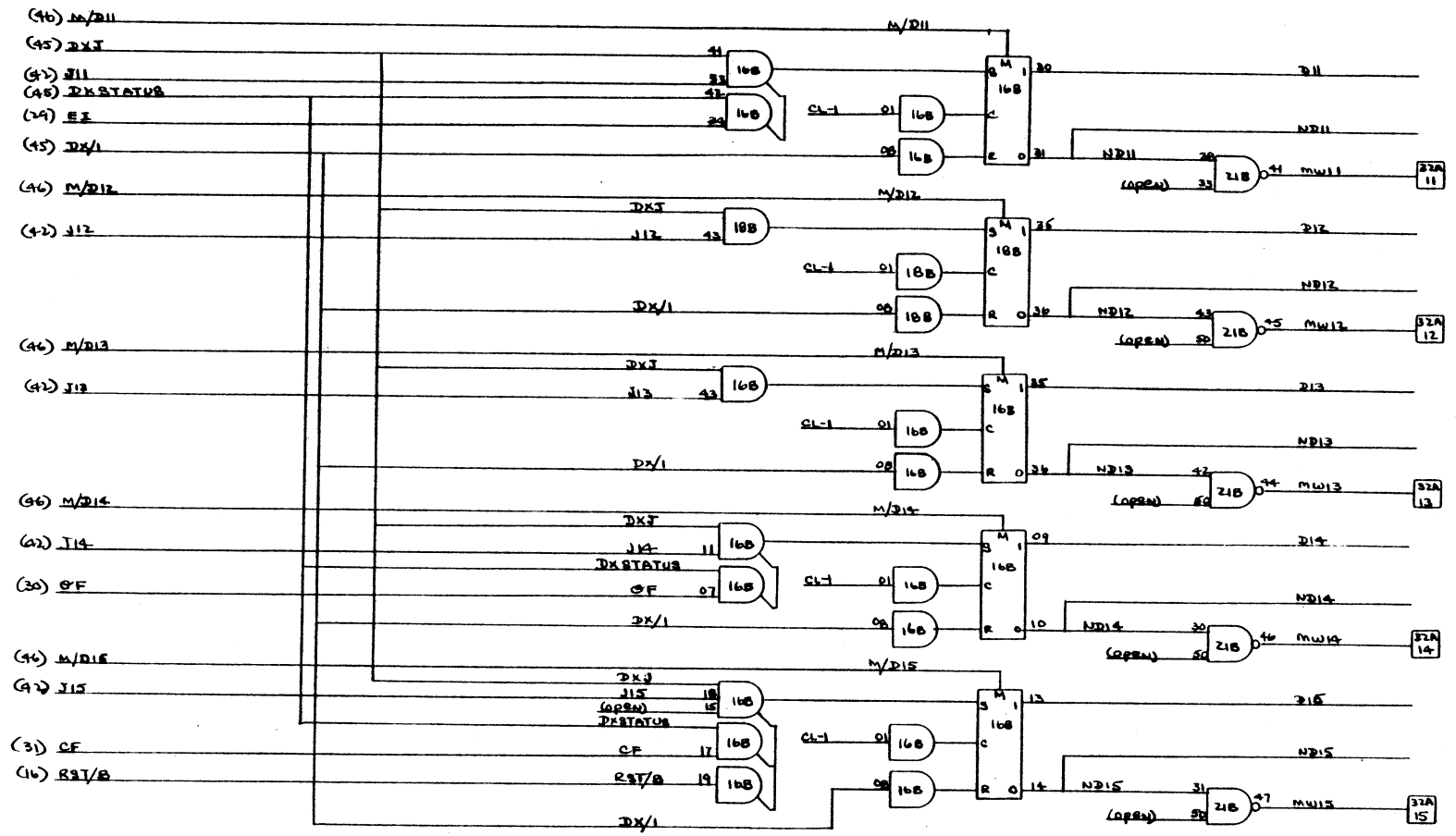


Figure 3-12. Logic Diagrams (sheet 49 of 91)

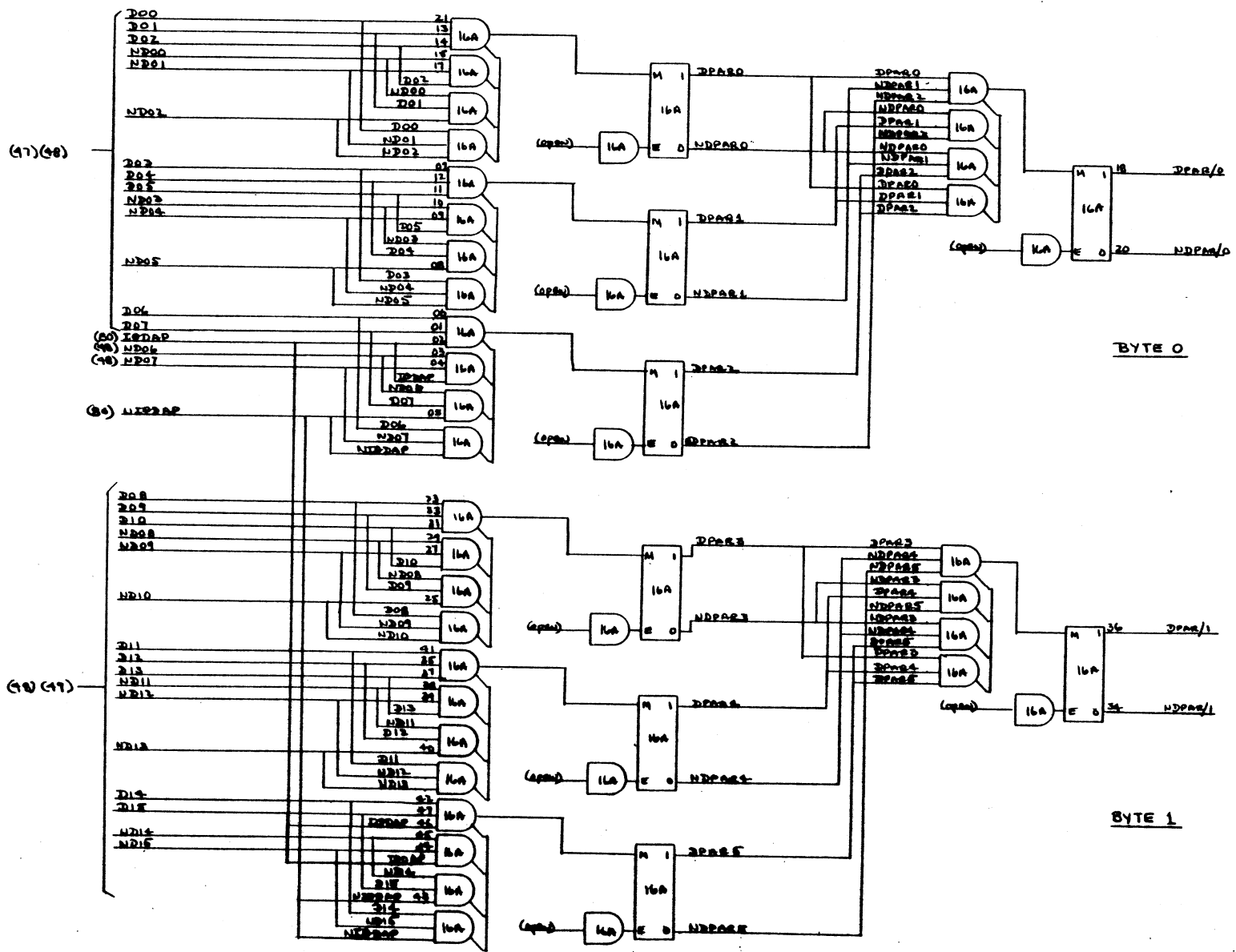


Figure 3-12. Logic Diagrams (sheet 50 of 11)

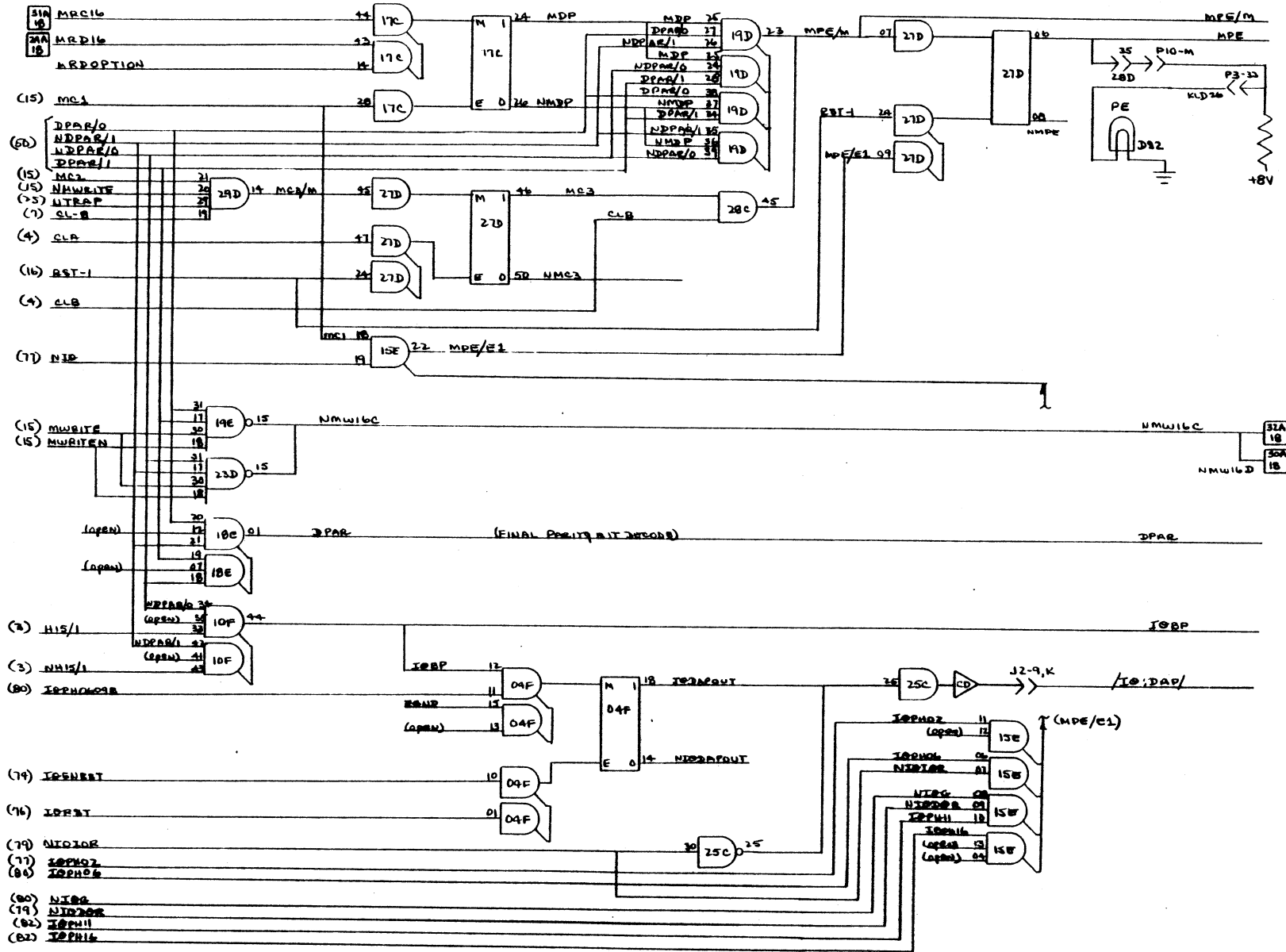


Figure 3-12. Logic Diagrams (sheet 51 of 91)

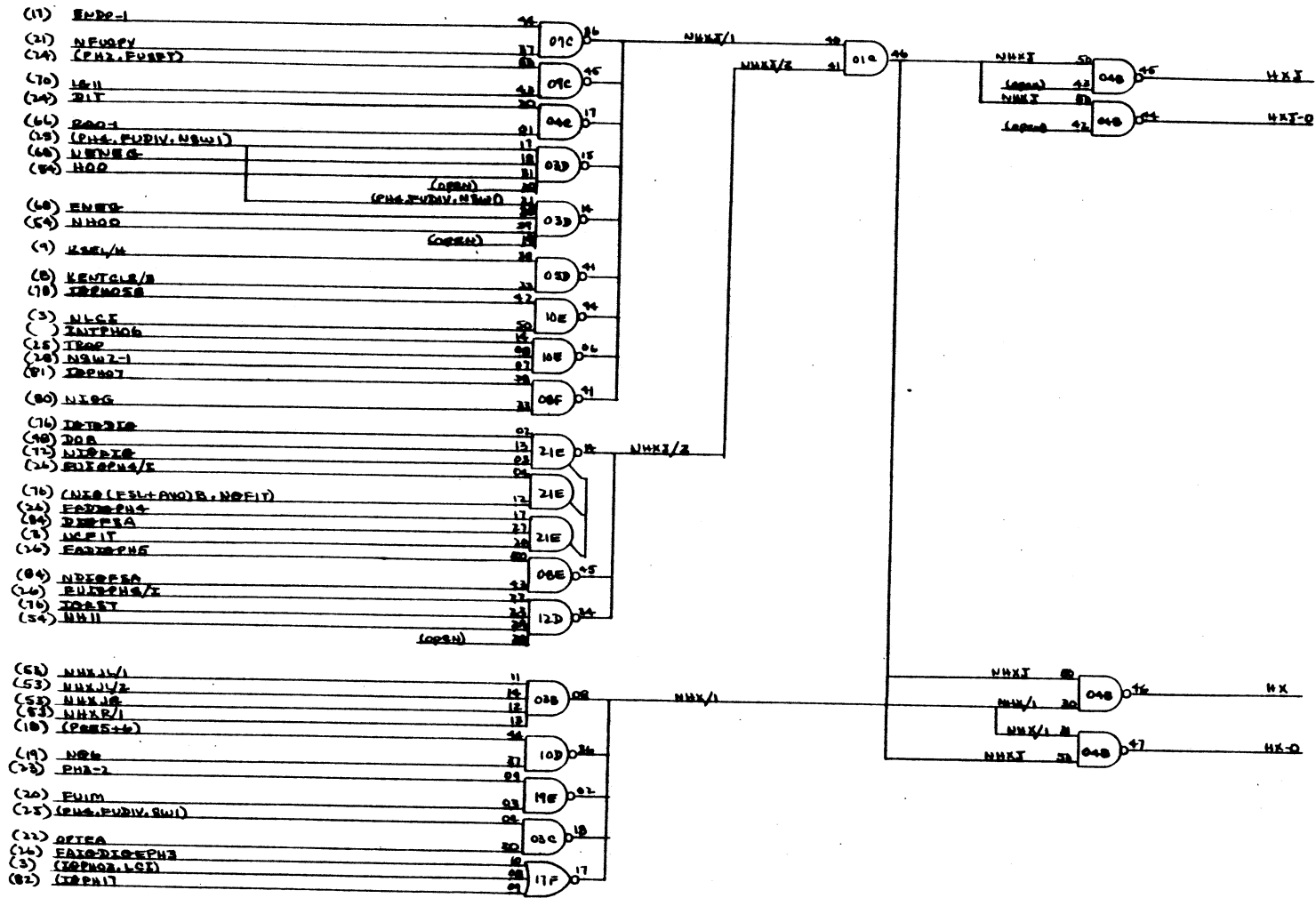


Figure 3-12. Logic Diagrams (sheet 52 of 91)

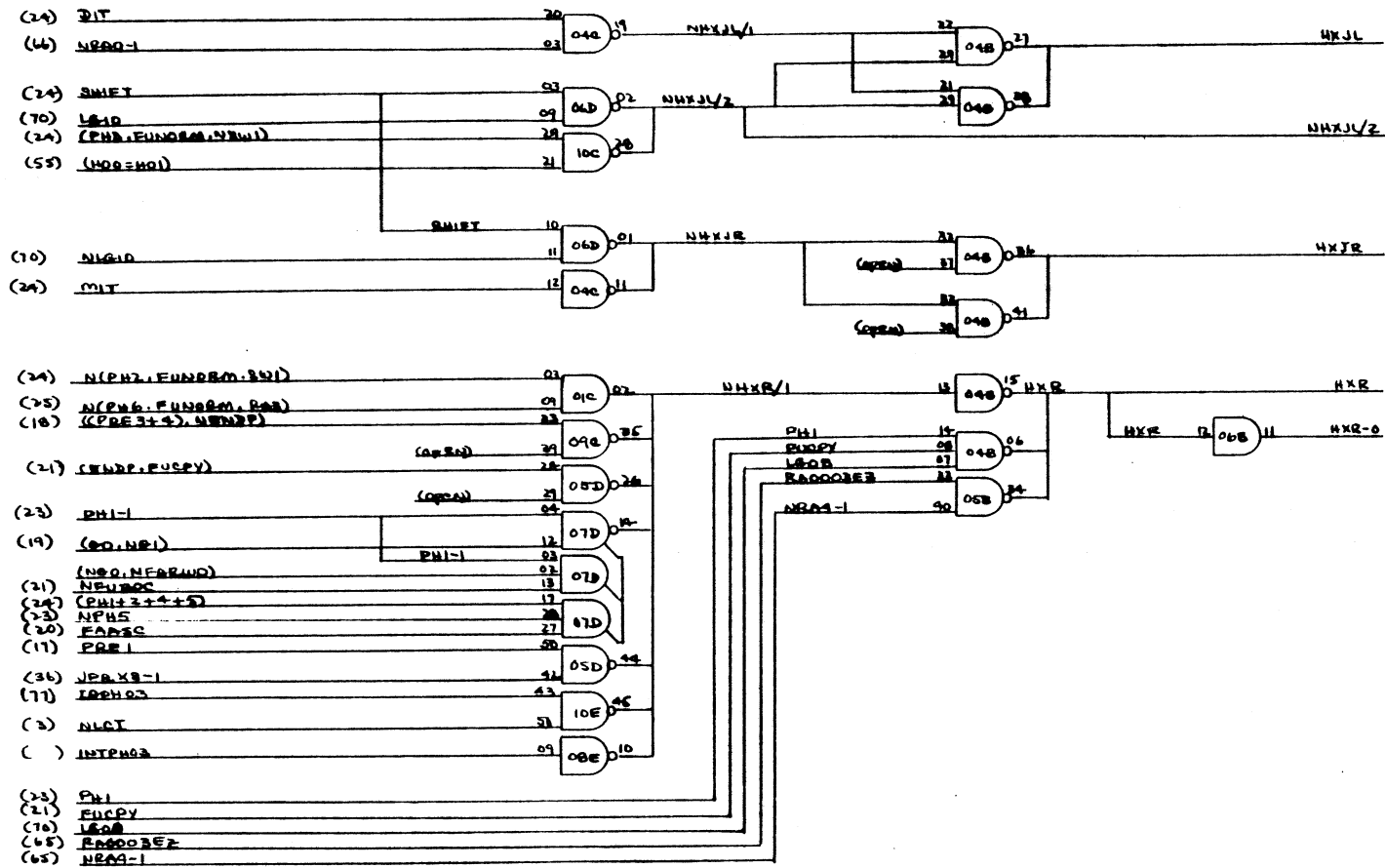
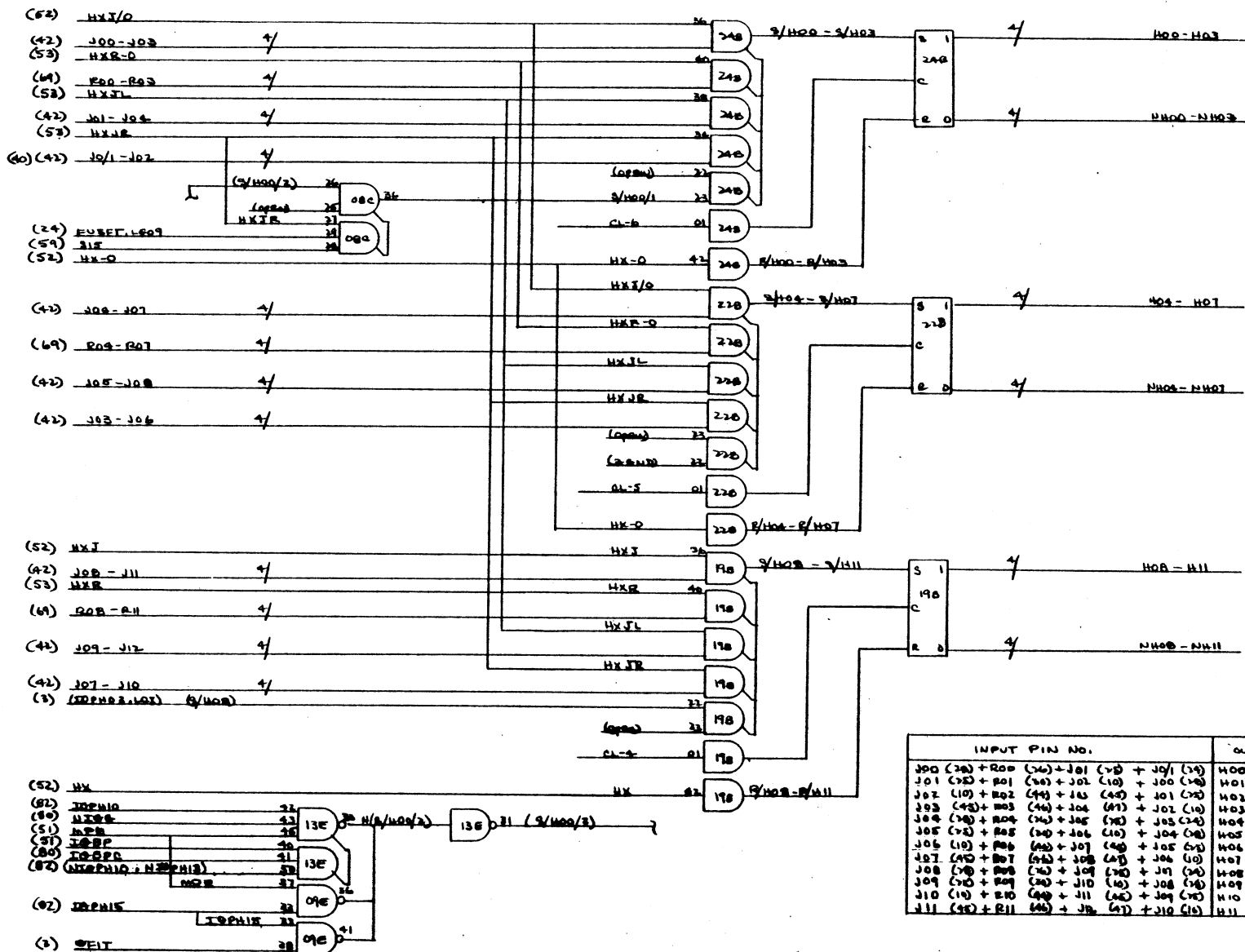
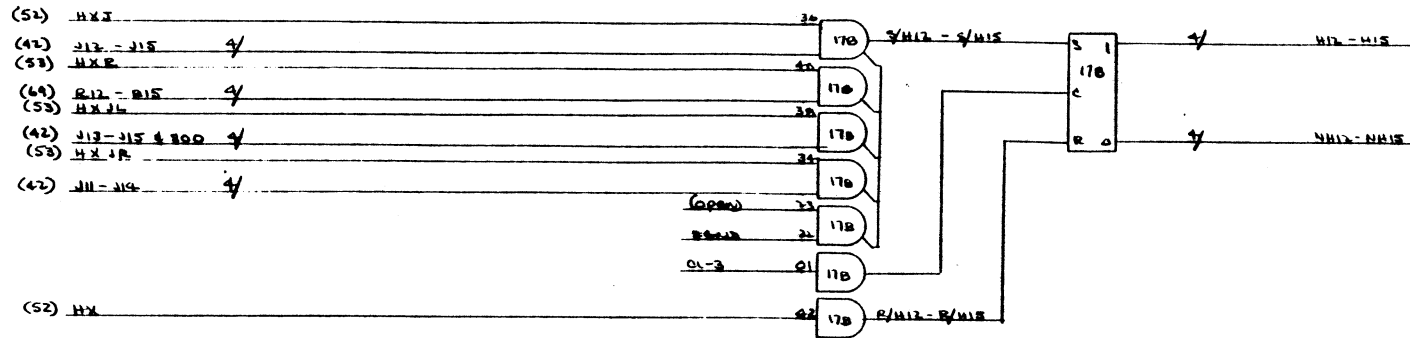


Figure 3-12. Logic Diagrams (sheet 53 of 91)



INPUT PIN NO.				OUTPUT PIN NO.	
J00 (36) + R00	(36)	+ J01 (37)	(37)	H00 (25)	NH00 (37)
J01 (38) + R01	(38)	+ J02 (10)	(10)	+ J00 (36)	H01 (38)
J02 (10) + R02	(44)	+ J03 (45)	(45)	+ J01 (37)	H02 (39)
J03 (45) + R03	(46)	+ J04 (47)	(47)	+ J02 (10)	H03 (41)
J04 (39) + R04	(46)	+ J05 (38)	(38)	+ J03 (45)	H04 (39)
J05 (38) + R05	(39)	+ J06 (41)	(41)	+ J04 (47)	H05 (42)
J06 (41) + R06	(40)	+ J07 (48)	(48)	+ J05 (38)	H06 (43)
J07 (48) + R07	(42)	+ J08 (49)	(49)	+ J06 (41)	H07 (44)
J08 (49) + R08	(43)	+ J09 (39)	(39)	+ J07 (48)	H08 (45)
J09 (39) + R09	(43)	+ J10 (40)	(40)	+ J08 (49)	H09 (46)
J10 (40) + R10	(44)	+ J11 (46)	(46)	+ J09 (39)	H10 (47)
J11 (46) + R11	(46)	+ J12 (47)	(47)	+ J10 (40)	H11 (48)

Figure 3-12. Logic Diagrams (sheet 54 of 91)



INPUT PIN NO.	OUTPUT PIN NO.
J12 (28) + R12 (24) + J13 (25) + J11 (24)	H12 (29) NH12 (21)
J13 (28) + R13 (24) + J14 (10) + J12 (28)	H13 (22) NH13 (22)
J14 (10) + R14 (44) + J15 (45) + J13 (28)	H14 (27) NH14 (23)
J15 (45) + R15 (44) + 300 (42) + J14 (10)	H15 (41) NH15 (42)

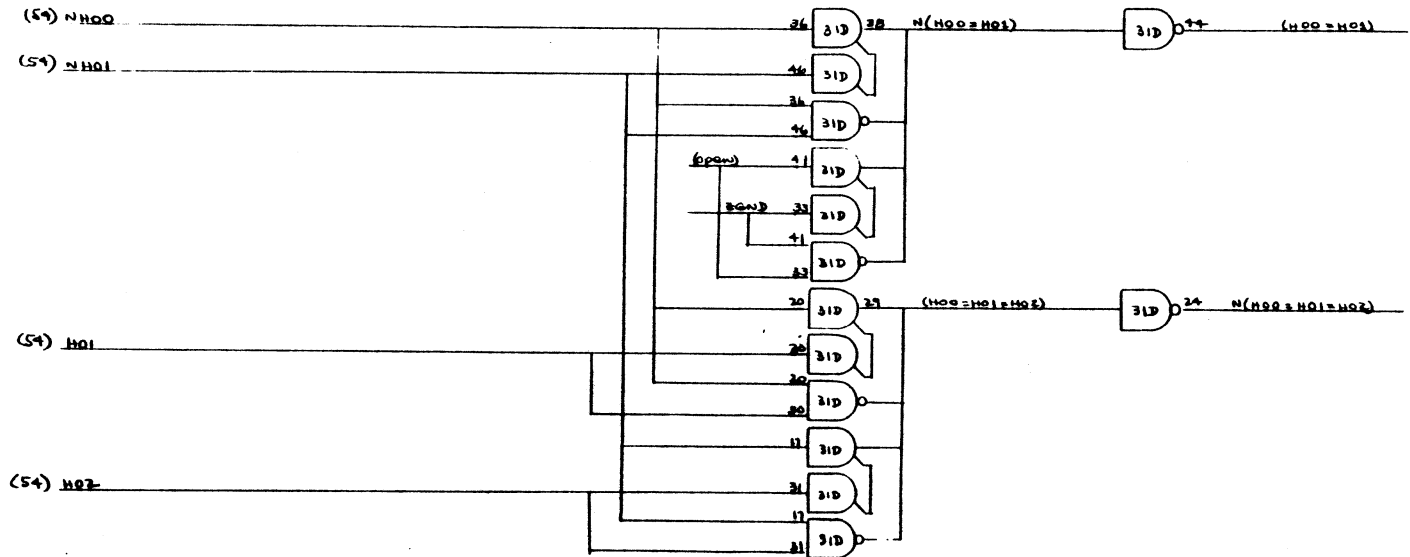
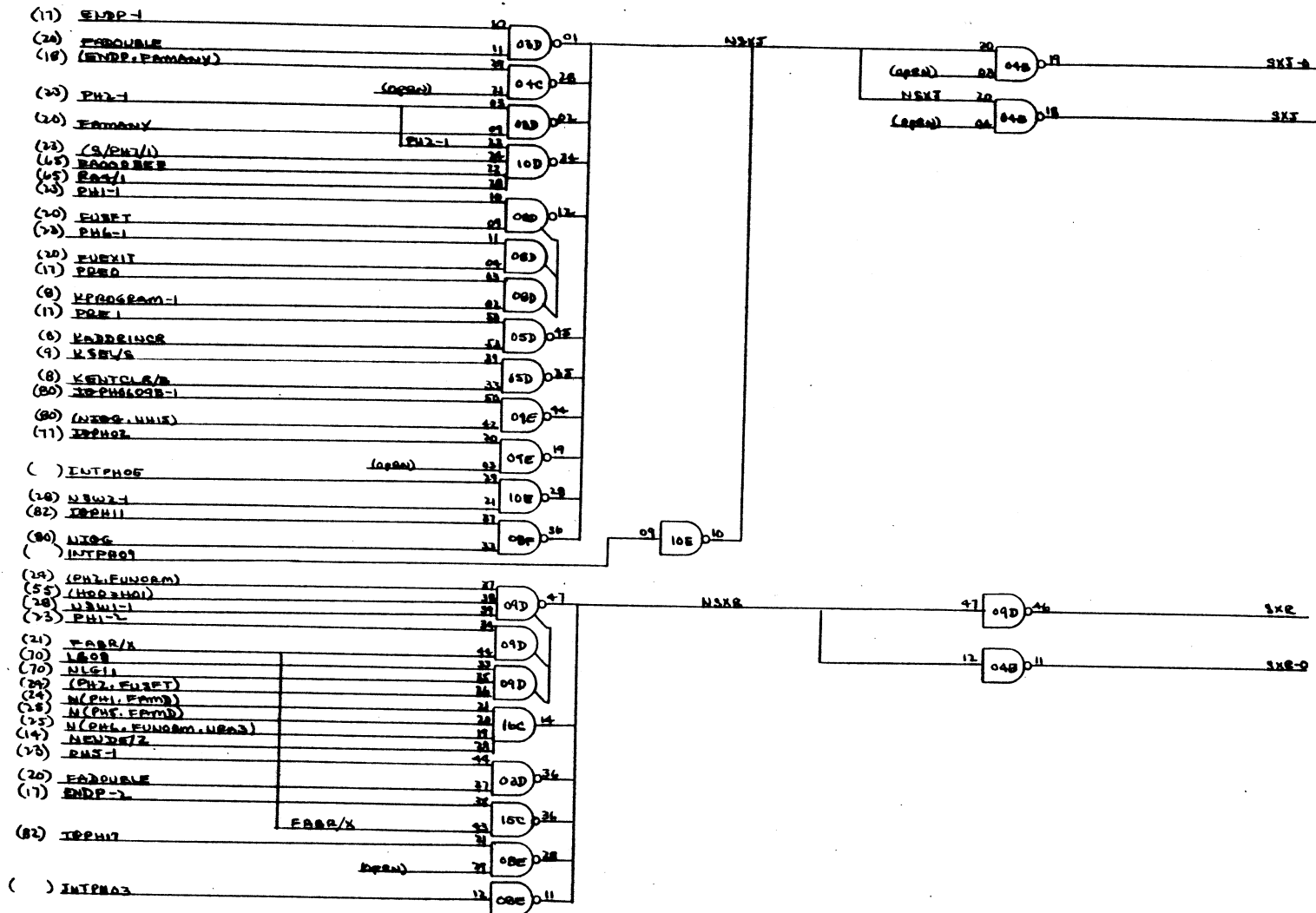


Figure 3-12. Logic Diagrams (sheet 55 of 91)



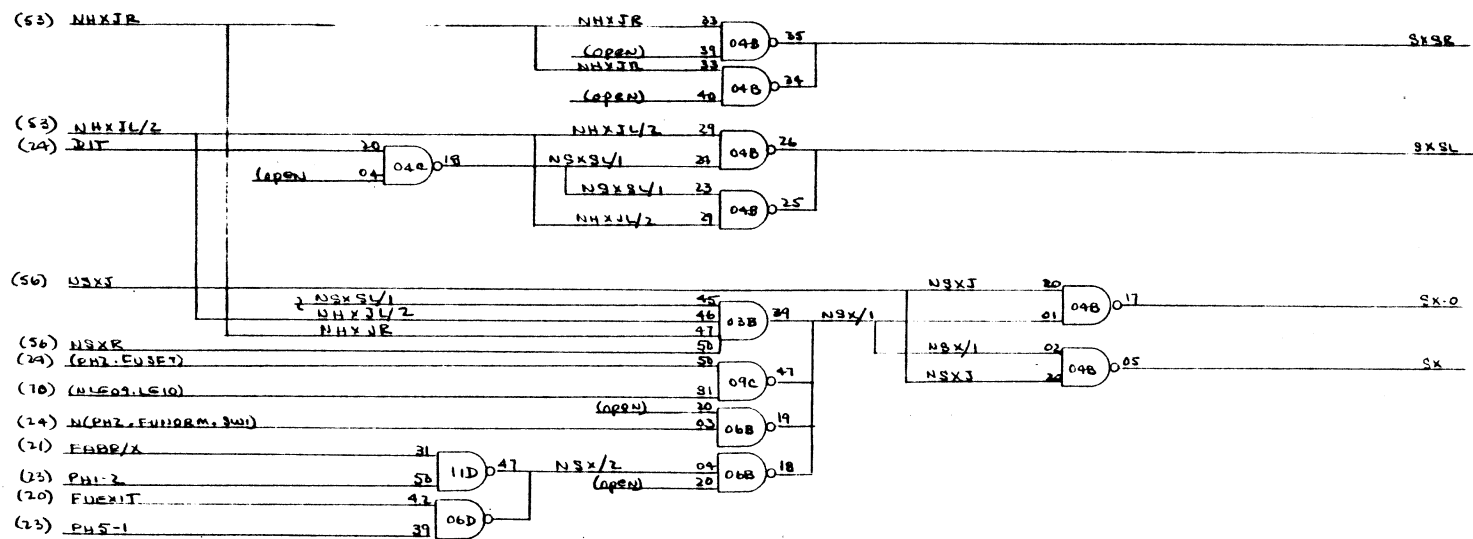
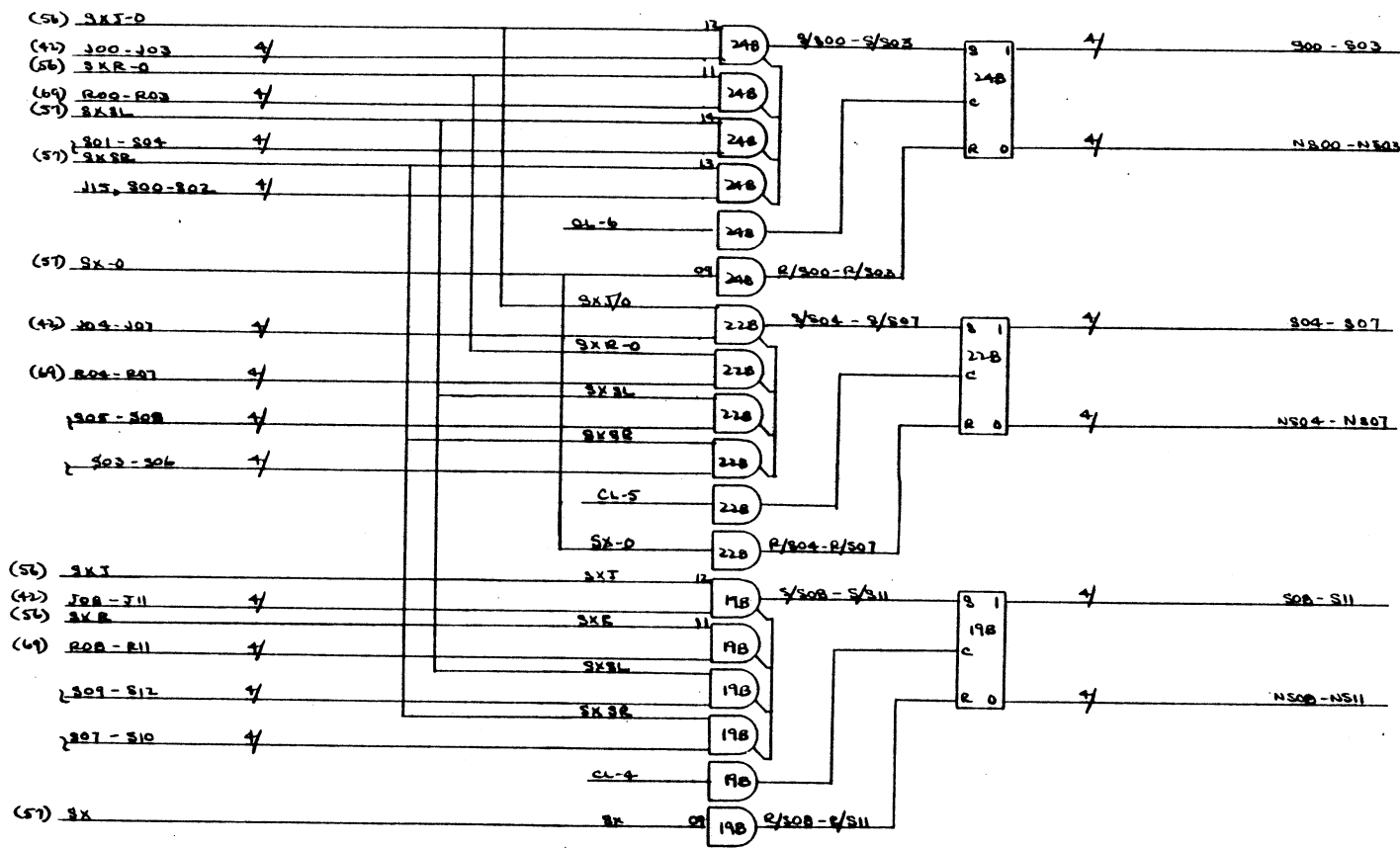
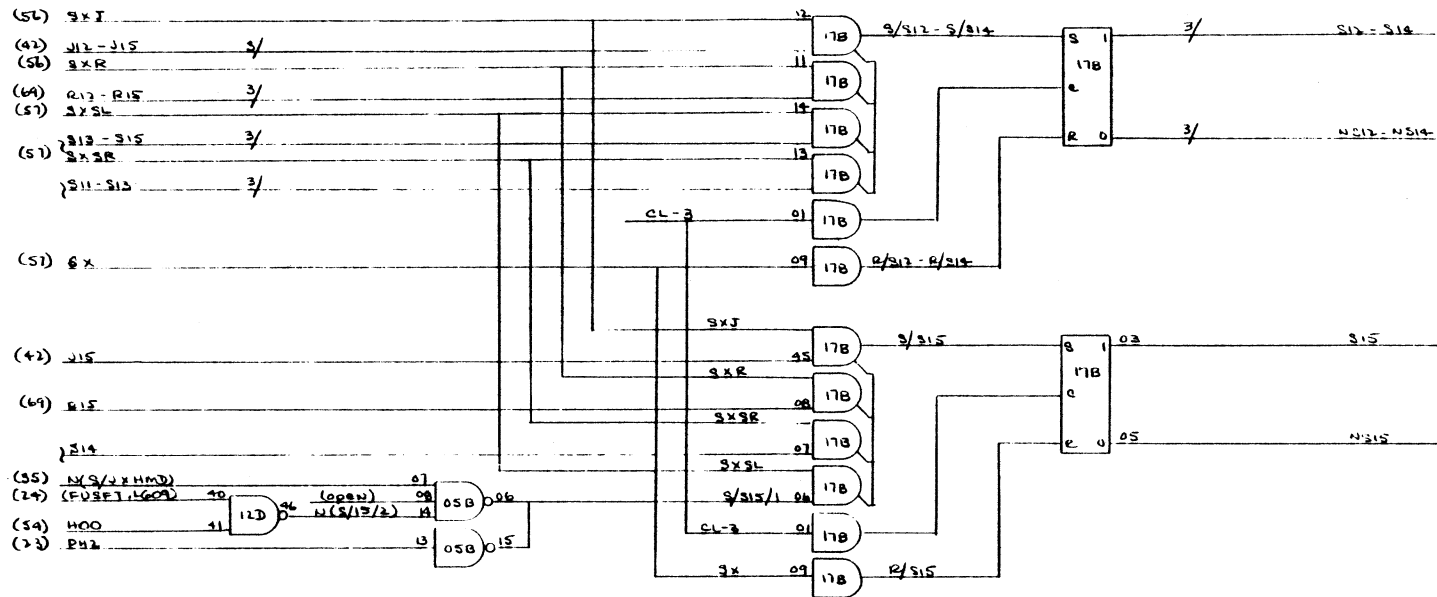


Figure 3-12. Logic Diagrams (sheet 57 of 91)



	INPUT PIN NO.	OUTPUT PIN NO.
24B	300 (30) + 300 (30) + 301 (11) + 318 (08)	300 (30) NS00 (30)
	301 (31) + 301 (31) + 302 (12) + 380 (34)	301 (31) NS01 (31)
	302 (10) + 302 (34) + 303 (35) + 301 (11)	302 (32) NS02 (10)
22B	303 (38) + 303 (38) + 304 (36) + 302 (31)	303 (33) NS03 (38)
	304 (38) + 304 (38) + 305 (17) + 303 (33)	304 (34) NS04 (37)
	305 (35) + 305 (35) + 306 (37) + 304 (33)	305 (35) NS05 (35)
19B	306 (10) + 306 (34) + 307 (38) + 305 (17)	306 (36) NS06 (10)
	307 (38) + 307 (38) + 308 (34) + 306 (37)	307 (38) NS07 (38)
	308 (38) + 308 (38) + 309 (17) + 307 (38)	308 (38) NS08 (38)
	309 (35) + 309 (35) + 310 (31) + 308 (34)	309 (37) NS09 (35)
	310 (10) + 310 (34) + 311 (38) + 309 (17)	310 (37) NS10 (10)
	311 (38) + 311 (38) + 312 (36) + 310 (31)	311 (38) NS11 (38)

Figure 3-12. Logic Diagrams (sheet 58 of 91)



INPUT PIN NO.	OUTPUT PIN NO.
J12 (20) + R12 (20) + S13 (17) + S11 (16)	S12 (22) NS12 (23)
J13 (21) + R13 (20) + S14 (04) + S12 (22)	S13 (17) NS13 (19)
J14 (10) + R14 (04) + S15 (03) + S13 (17)	S14 (07) NS14 (18)

* SEE LOGIC DIAGRAM FOR PIN INFORMATION

Figure 3-12. Logic Diagrams (sheet 59 of 91)

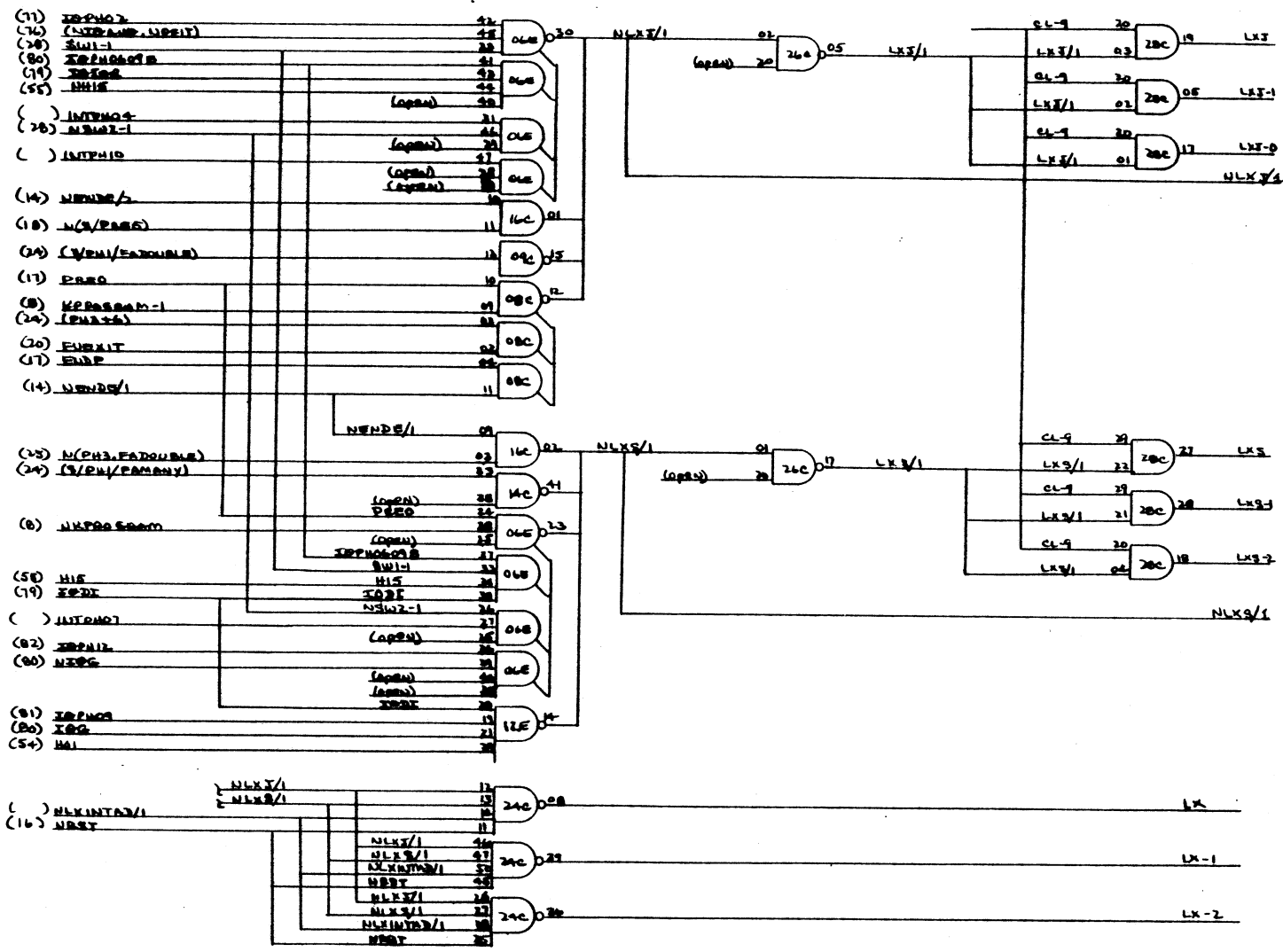
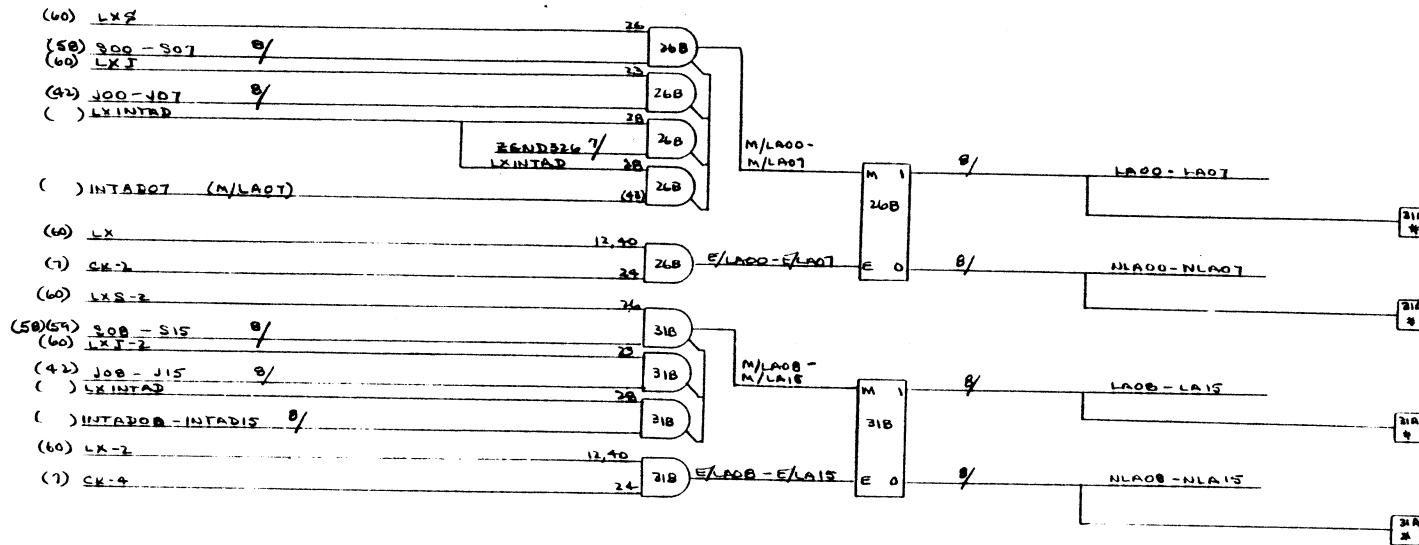


Figure 3-12. Logic Diagrams (sheet 60 of 91)



INPUT PIN NO.	OUTPUT PIN NO.	# RIBBON CABLE TO MEMORY - PIN NO. (31A)
900 (03) + J00 (01) + 26+D (05)	LA00 (04)	LA00 (24)
901 (09) + J01 (10) + 26+D (05)	LA01 (06)	LA01 (25)
902 (15) + J02 (11) + 26+D (05)	LA02 (10)	LA02 (26)
903 (19) + J03 (21) + 26+D (05)	LA03 (14)	LA03 (27)
904 (27) + J04 (20) + 26+D (05)	LA04 (18)	LA04 (28)
905 (33) + J05 (30) + 26+D (05)	LA05 (22)	LA05 (29)
906 (37) + J06 (31) + 26+D (05)	LA06 (26)	LA06 (30)
907 (43) + J07 (50) + INTAD07 (43)	LA07 (30)	LA07 (31)
908 (03) + J08 (01) + INTAD08 (08)	LA08 (04)	LA08 (32)
909 (09) + J09 (10) + INTAD09 (07)	LA09 (06)	LA09 (33)
910 (13) + J10 (11) + INTAD10 (05)	LA10 (10)	LA10 (34)
911 (19) + J11 (21) + INTAD11 (17)	LA11 (14)	LA11 (35)
912 (27) + J12 (20) + INTAD12 (19)	LA12 (18)	LA12 (36)
913 (33) + J13 (30) + INTAD13 (31)	LA13 (22)	LA13 (37)
914 (37) + J14 (31) + INTAD14 (41)	LA14 (26)	LA14 (38)
915 (43) + J15 (20) + INTAD15 (43)	LA15 (30)	LA15 (39)
	NLA00 (02)	NLA00 (24)
	NLA01 (08)	NLA01 (25)
	NLA02 (14)	NLA02 (26)
	NLA03 (20)	NLA03 (27)
	NLA04 (26)	NLA04 (28)
	NLA05 (32)	NLA05 (29)
	NLA06 (38)	NLA06 (30)
	NLA07 (44)	NLA07 (31)
	NLA08 (50)	NLA08 (32)
	NLA09 (06)	NLA09 (33)
	NLA10 (12)	NLA10 (34)
	NLA11 (18)	NLA11 (35)
	NLA12 (24)	NLA12 (36)
	NLA13 (30)	NLA13 (37)
	NLA14 (36)	NLA14 (38)
	NLA15 (42)	NLA15 (39)

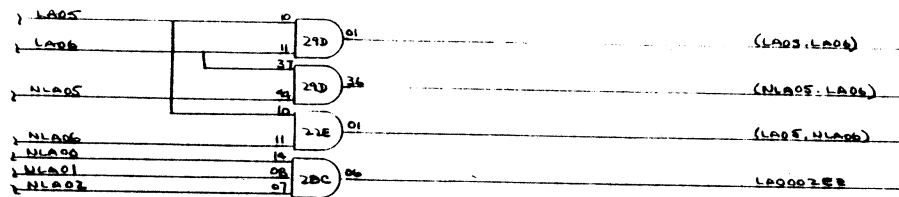


Figure 3-12. Logic Diagrams (sheet 61 of 91)

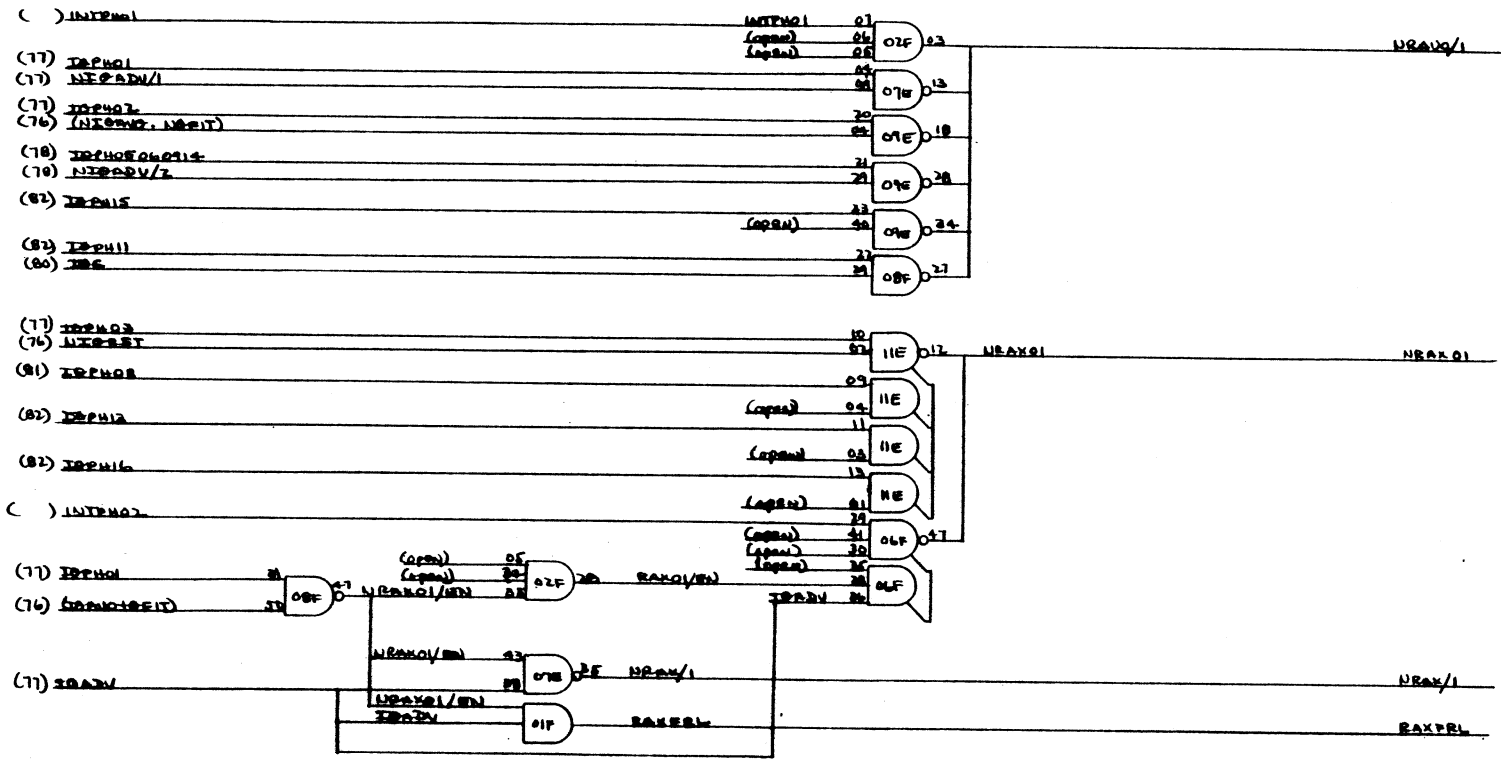


Figure 3-12. Logic Diagrams (sheet 62 of 91)

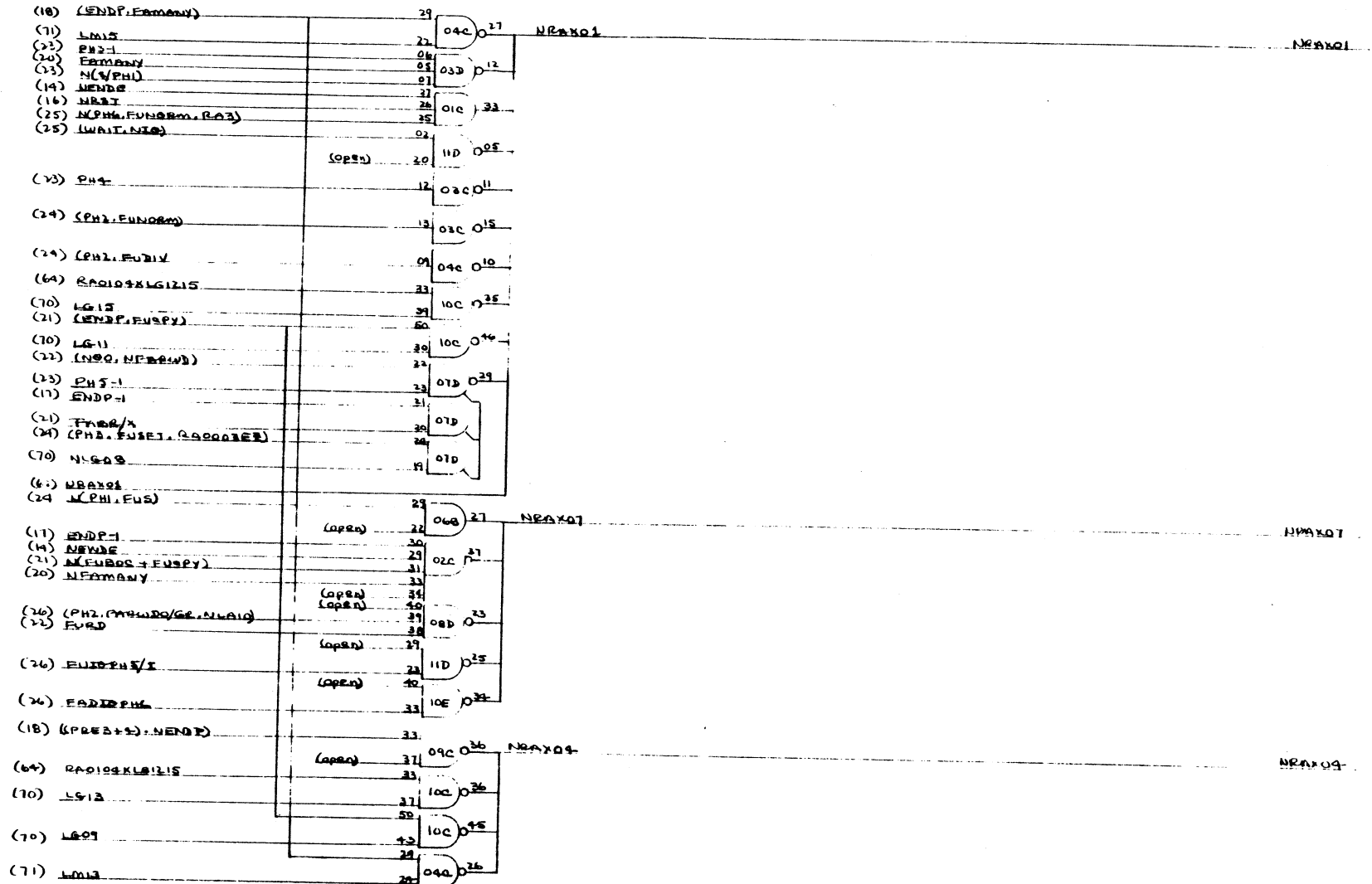


Figure 3-12. Logic Diagrams (sheet 63 of 91)

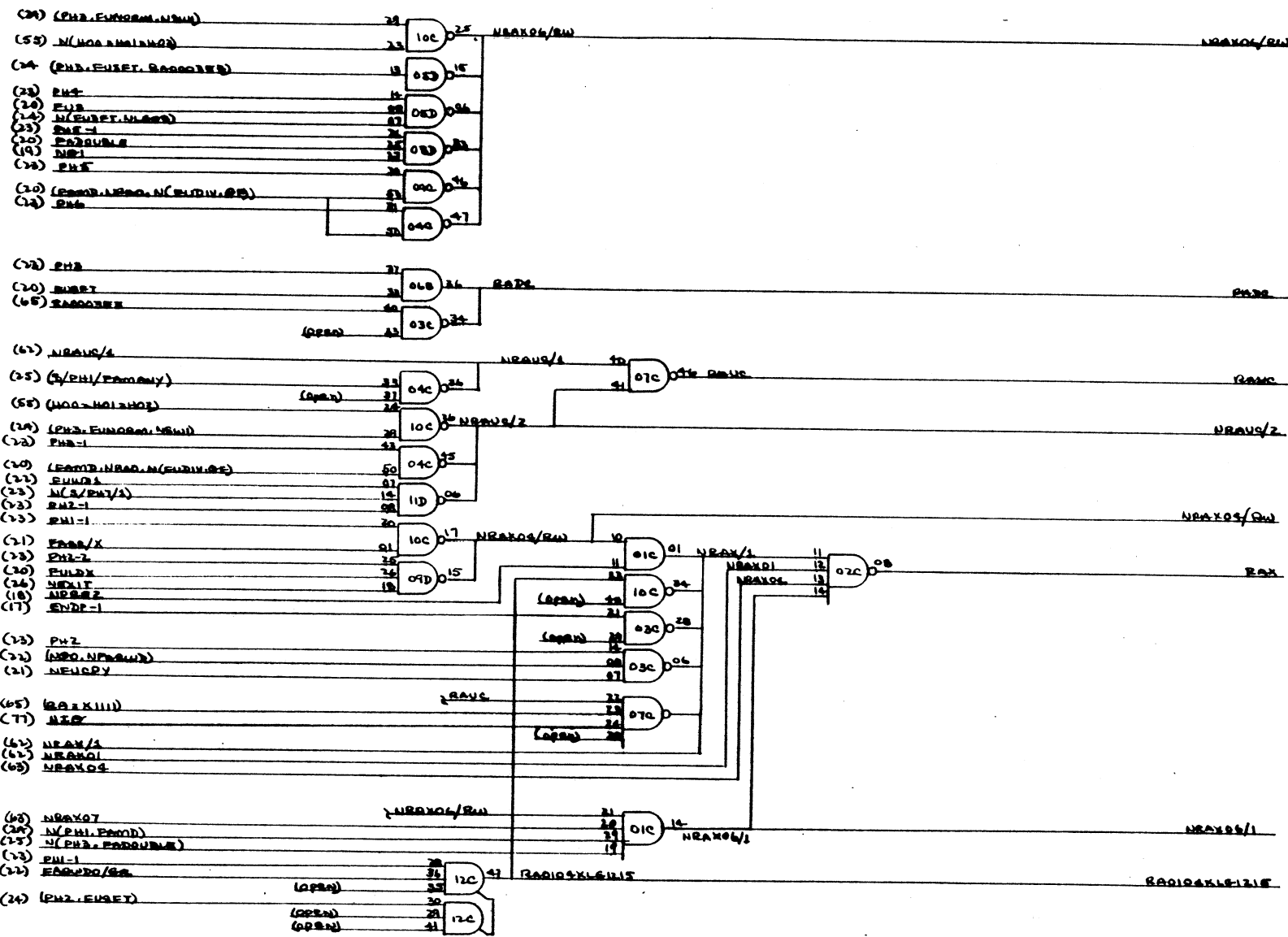


Figure 3-12. Logic Diagrams (sheet 64 of 91)

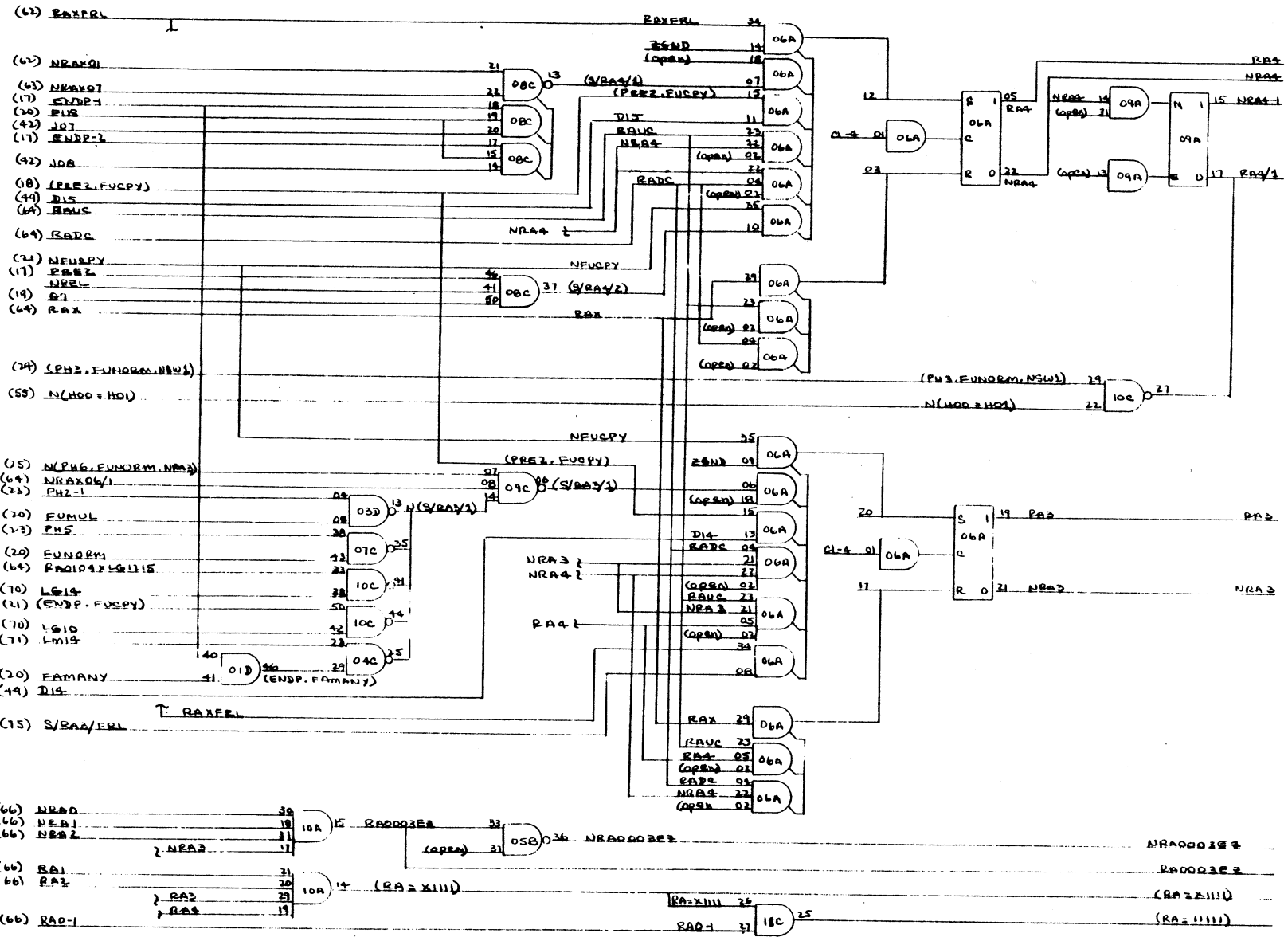


Figure 3-12. Logic Diagrams (sheet 65 of 91)

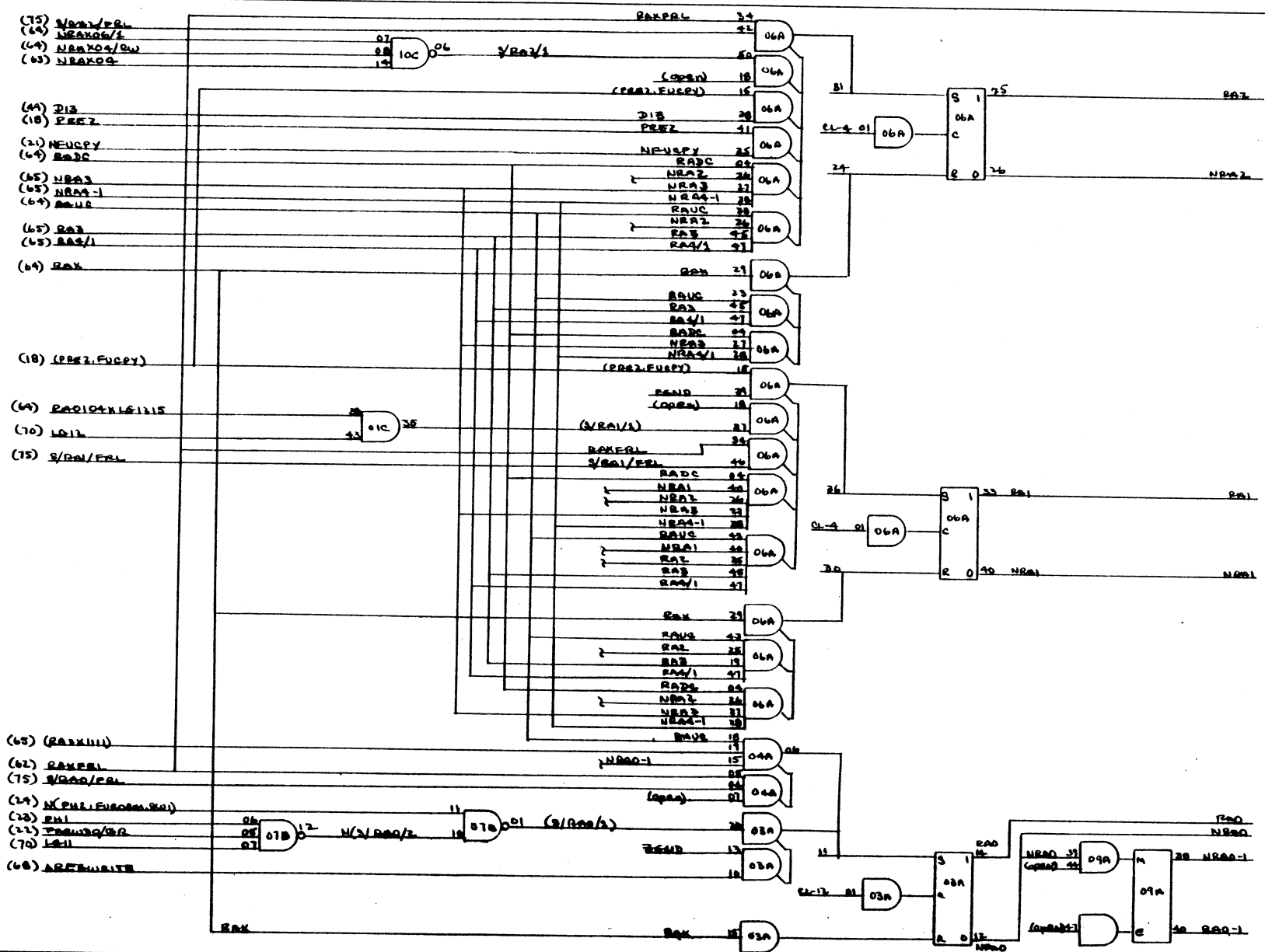


Figure 3-12. Logic Diagrams (sheet 66 of 71)

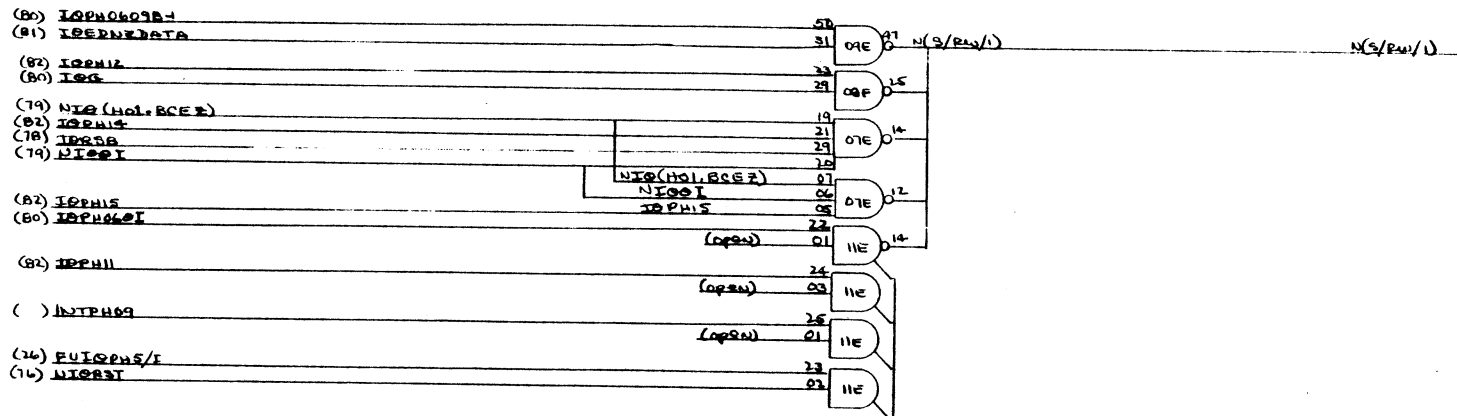


Figure 3-12. Logic Diagrams (sheet 67 of 91)

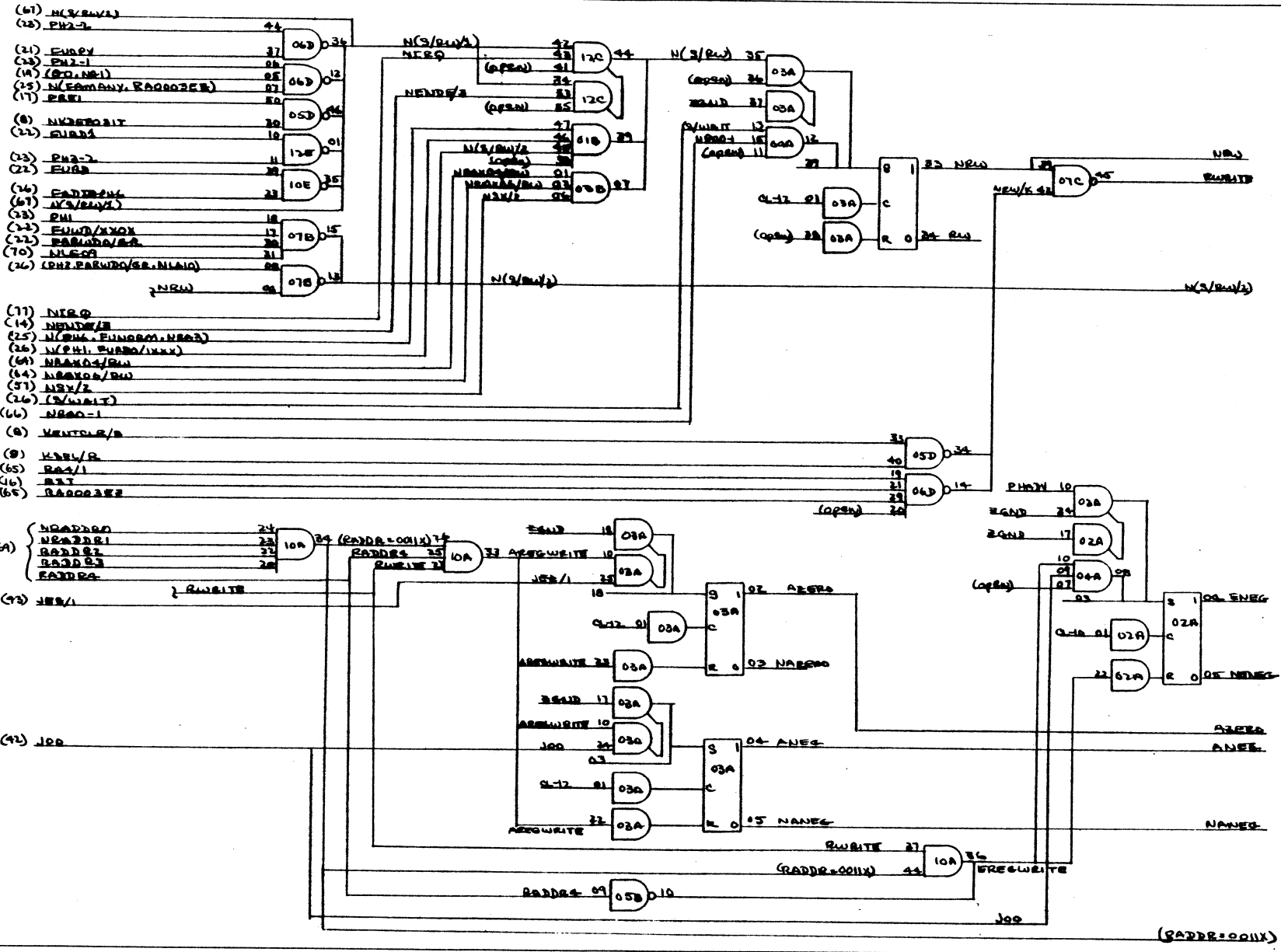
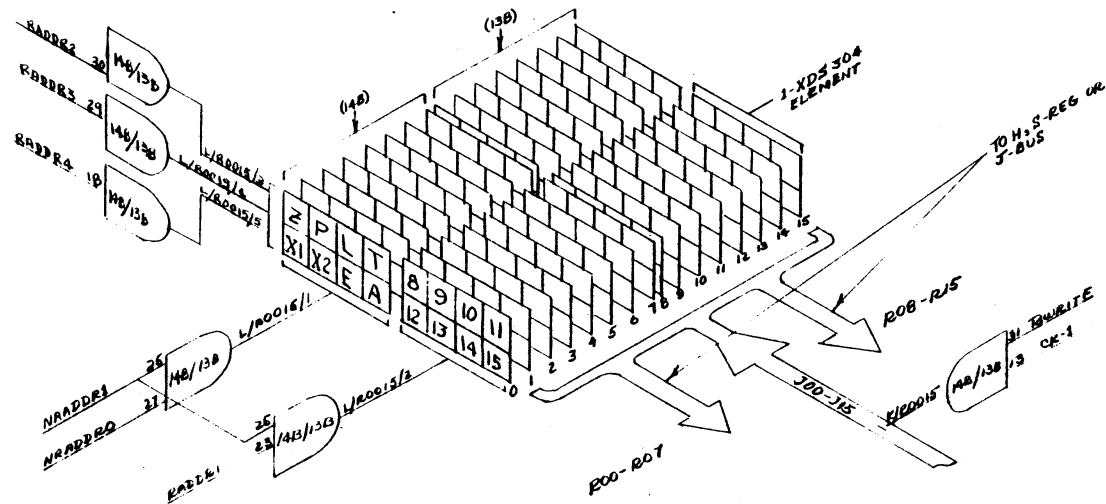


Figure 3-12. Logic Diagrams (sheet 68 of 91)



GENERAL AND I/O REGISTERS			
J-BUS INPUT PIN		REG. OUTPUT PIN	
J00 (35)	J08 (35)	R00 (43)	R08 (43)
J01 (34)	J09 (34)	R01 (41)	R09 (41)
J02 (19)	J10 (19)	R02 (09)	R10 (09)
J03 (11)	J11 (11)	R03 (02)	R11 (02)
J04 (23)	J12 (23)	R04 (45)	R12 (45)
J05 (28)	J13 (28)	R05 (46)	R13 (46)
J06 (17)	J14 (17)	R06 (04)	R14 (04)
J07 (11)	J15 (11)	R07 (07)	R15 (07)
14B	13B	14B	13B

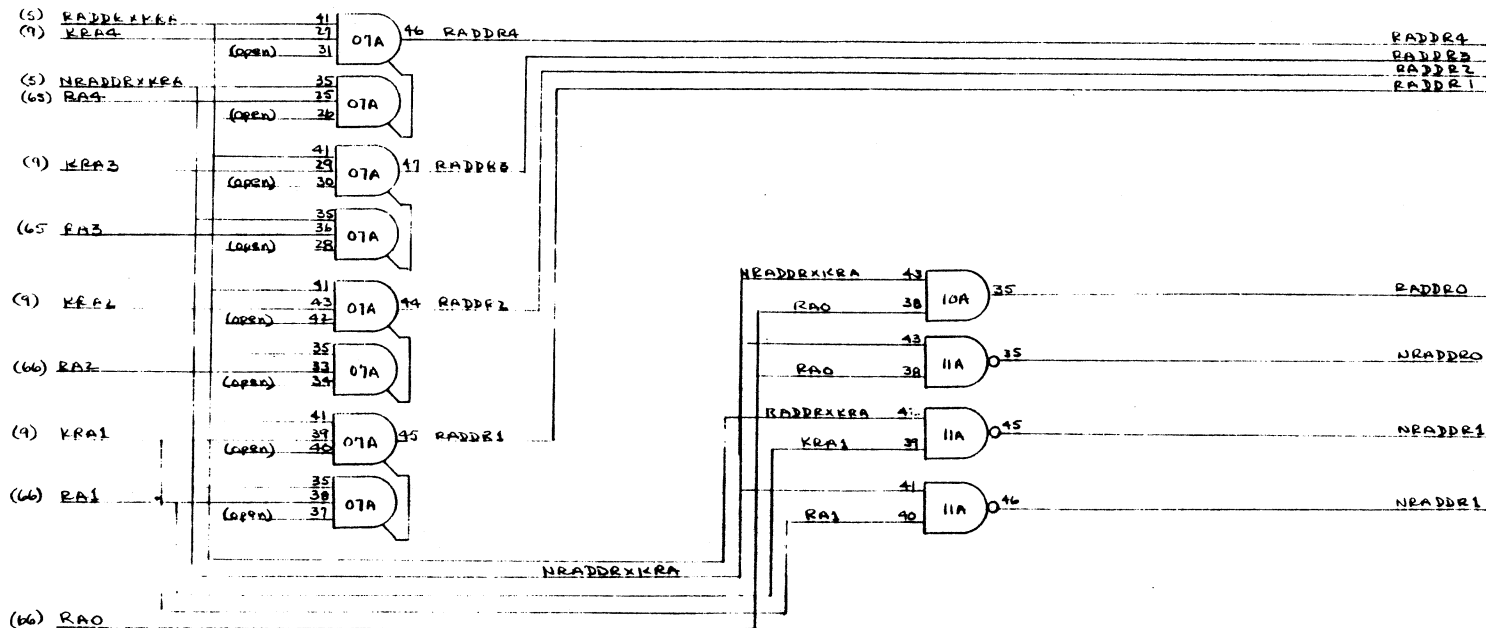
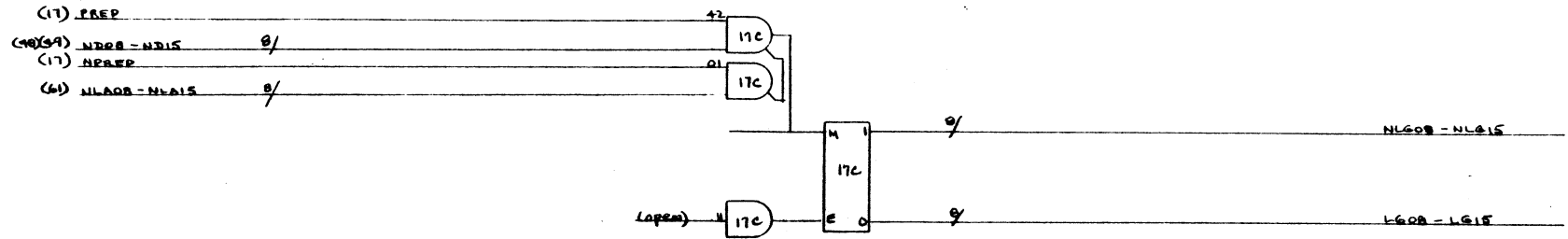
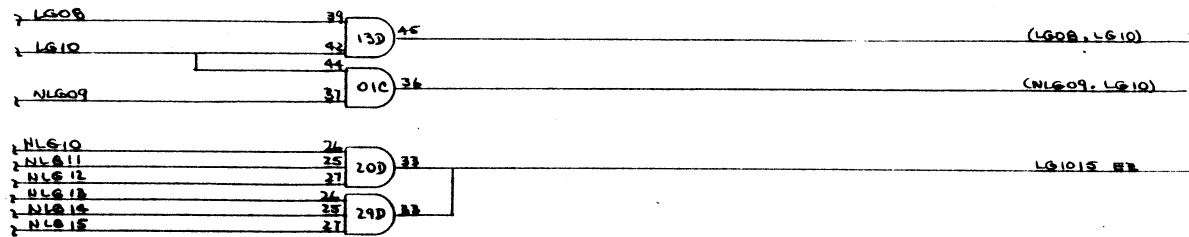


Figure 3-12. Logic Diagrams (sheet 69 of 91)



INPUT PIN NO.	OUTPUT PIN NO.
ND08 (13) + NLA08 (25)	NLG08 (07) LG08(03)
ND09 (24) + NLA09 (22)	NLG09 (08) LG09 (04)
ND10 (15) + NLA10 (11)	NLG10 (25) LG10 (21)
ND11 (33) + NLA11 (29)	NLG11 (34) LG11 (30)
ND12 (21) + NLA12 (65)	NLG12 (30) LG12 (31)
ND13 (46) + NLA13 (47)	NLG13 (40) LG13 (38)
ND14 (39) + NLA14 (41)	NLG14 (45) LG14 (50)
ND15 (19) + NLA15 (21)	NLG15 (22) LG15 (43)



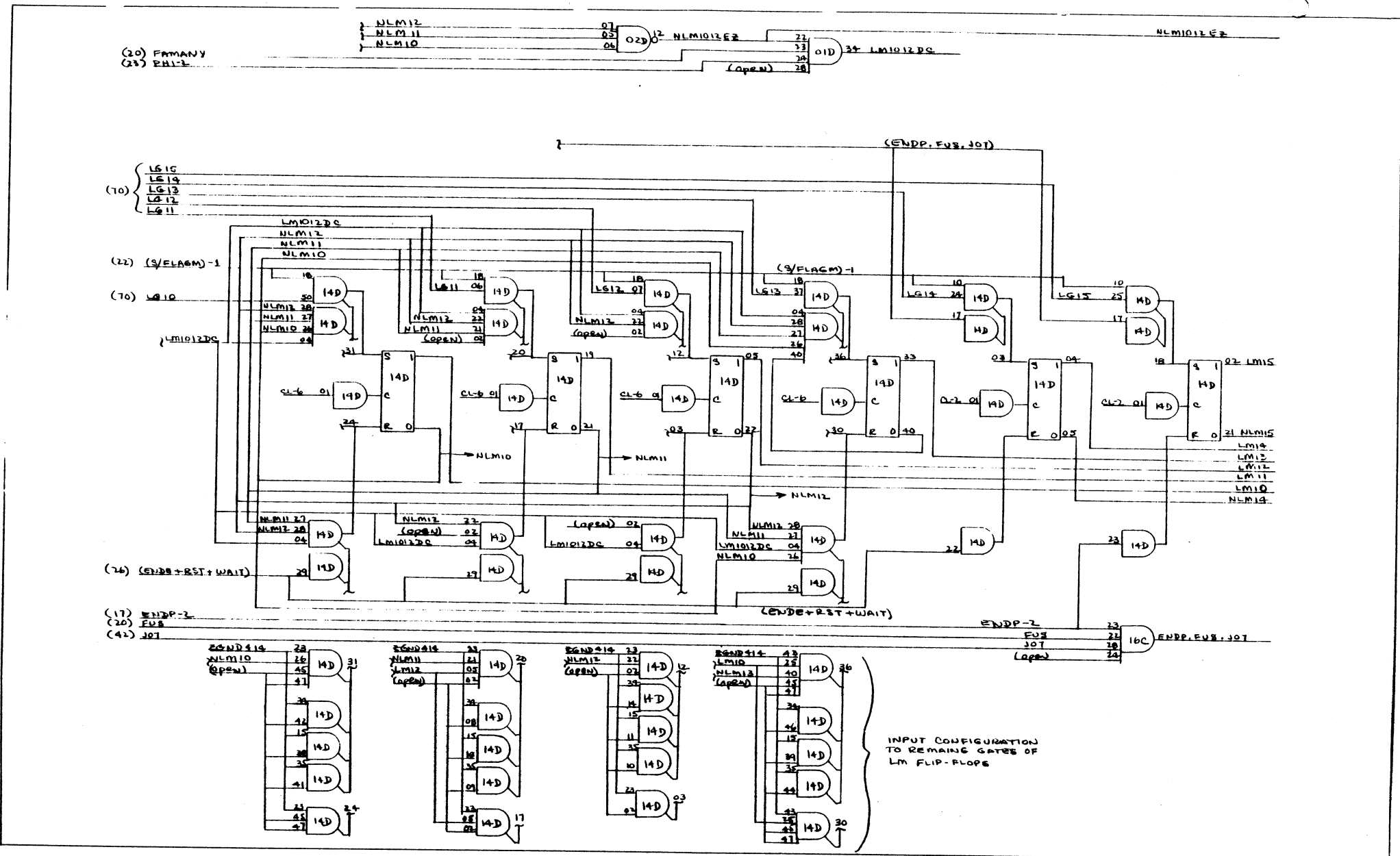


Figure 3-12. Logic Diagrams (sheet 71 of 91)

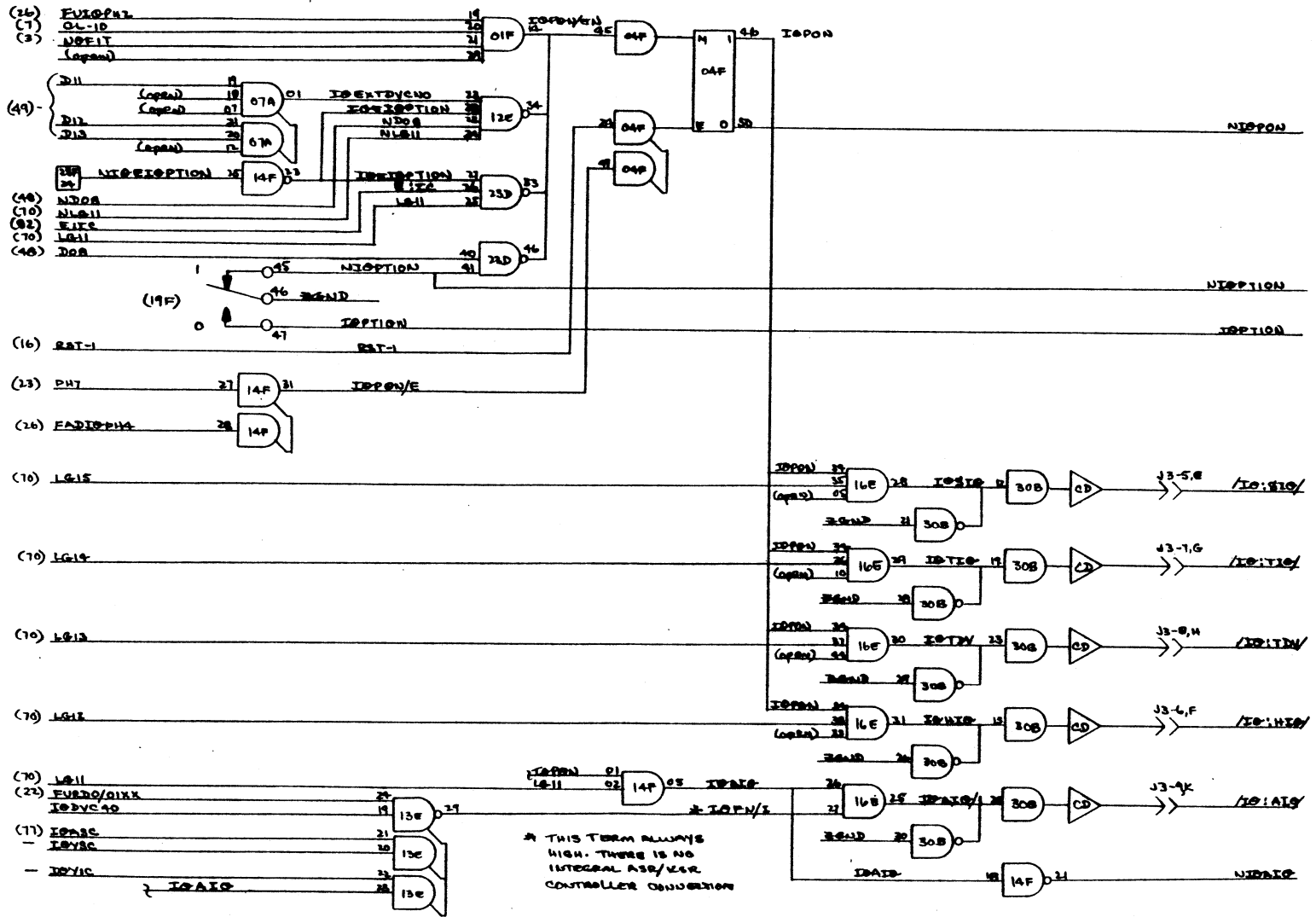


Figure 3-12. Logic Diagrams (sheet 72 of 91)

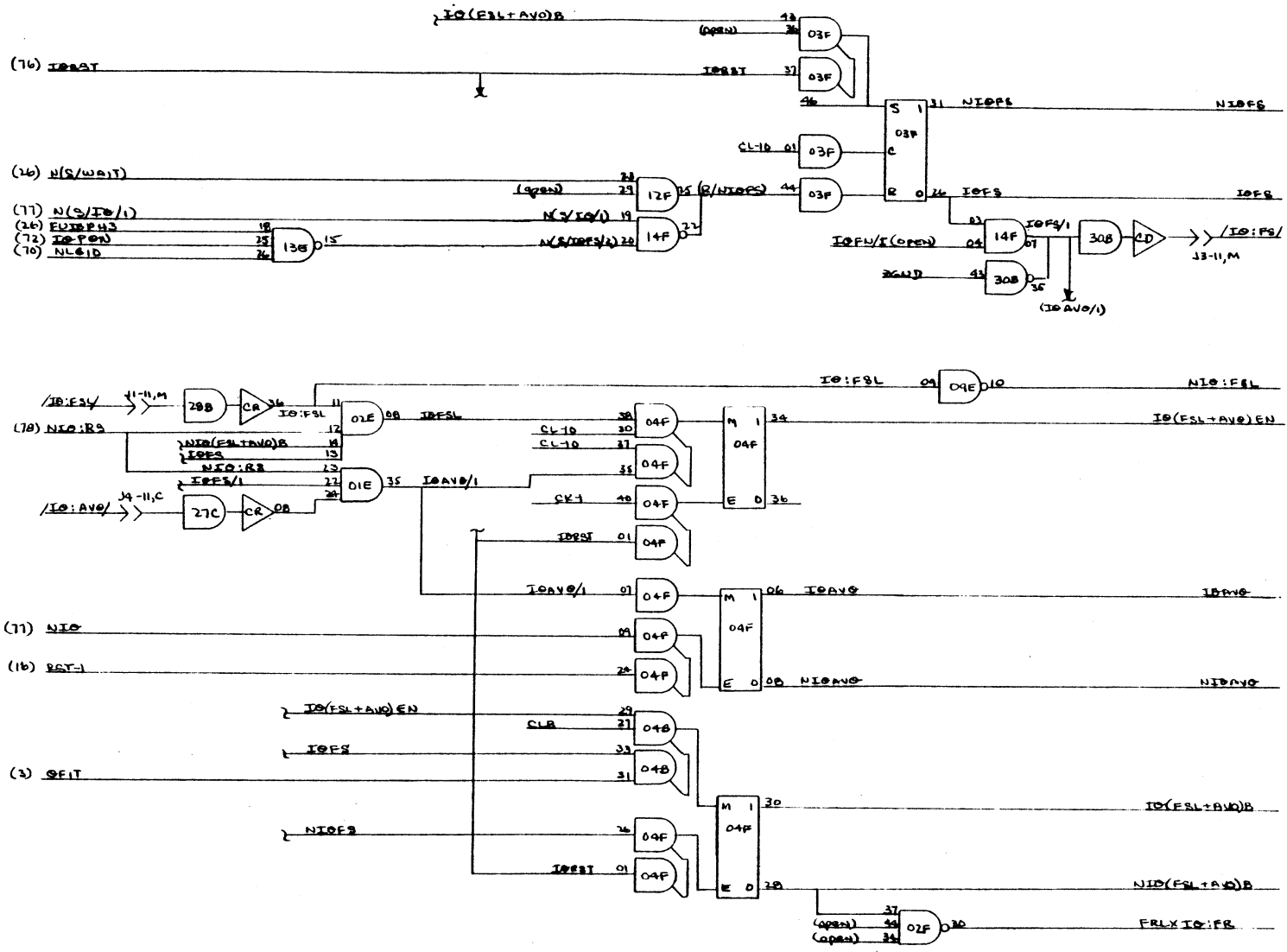
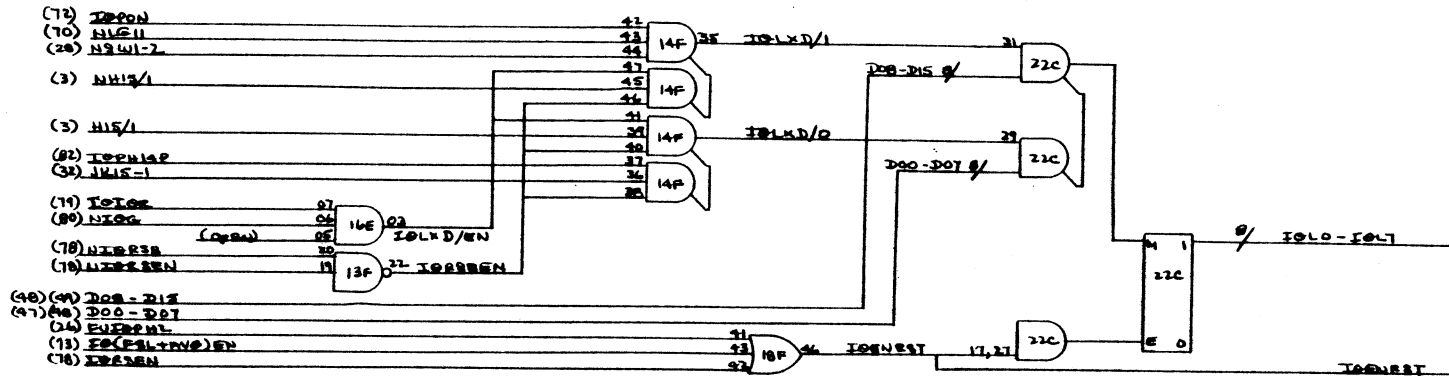
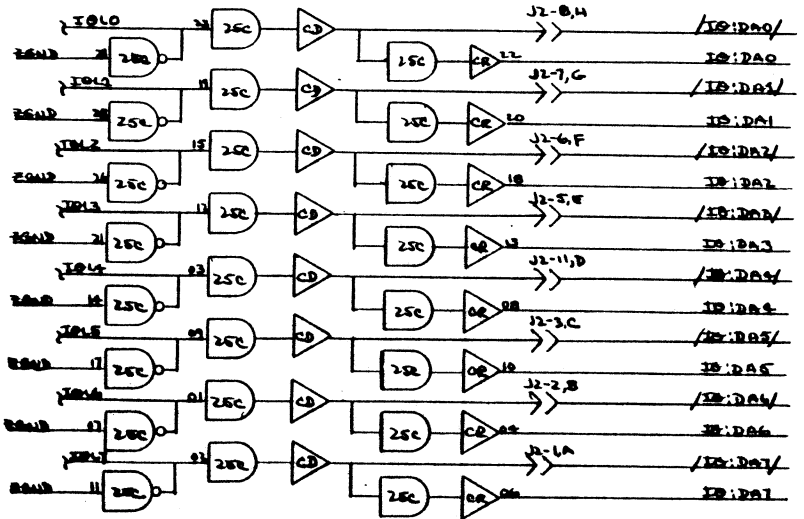


Figure 3-12. Logic Diagrams (sheet 73 of 91)



IOL-REGISTER INPUT PIN NO.	OUTPUT PIN NO.
D00 (03) + D08 (02)	IOL0 (15)
D01 (06) + D09 (17)	IOL1 (16)
D02 (10) + D10 (11)	IOL2 (17)
D03 (20) + D11 (21)	IOL3 (18)
D04 (20) + D12 (24)	IOL4 (20)
D05 (29) + D13 (36)	IOL5 (21)
D06 (40) + D14 (43)	IOL6 (22)
D07 (44) + D15 (47)	IOL7 (23)



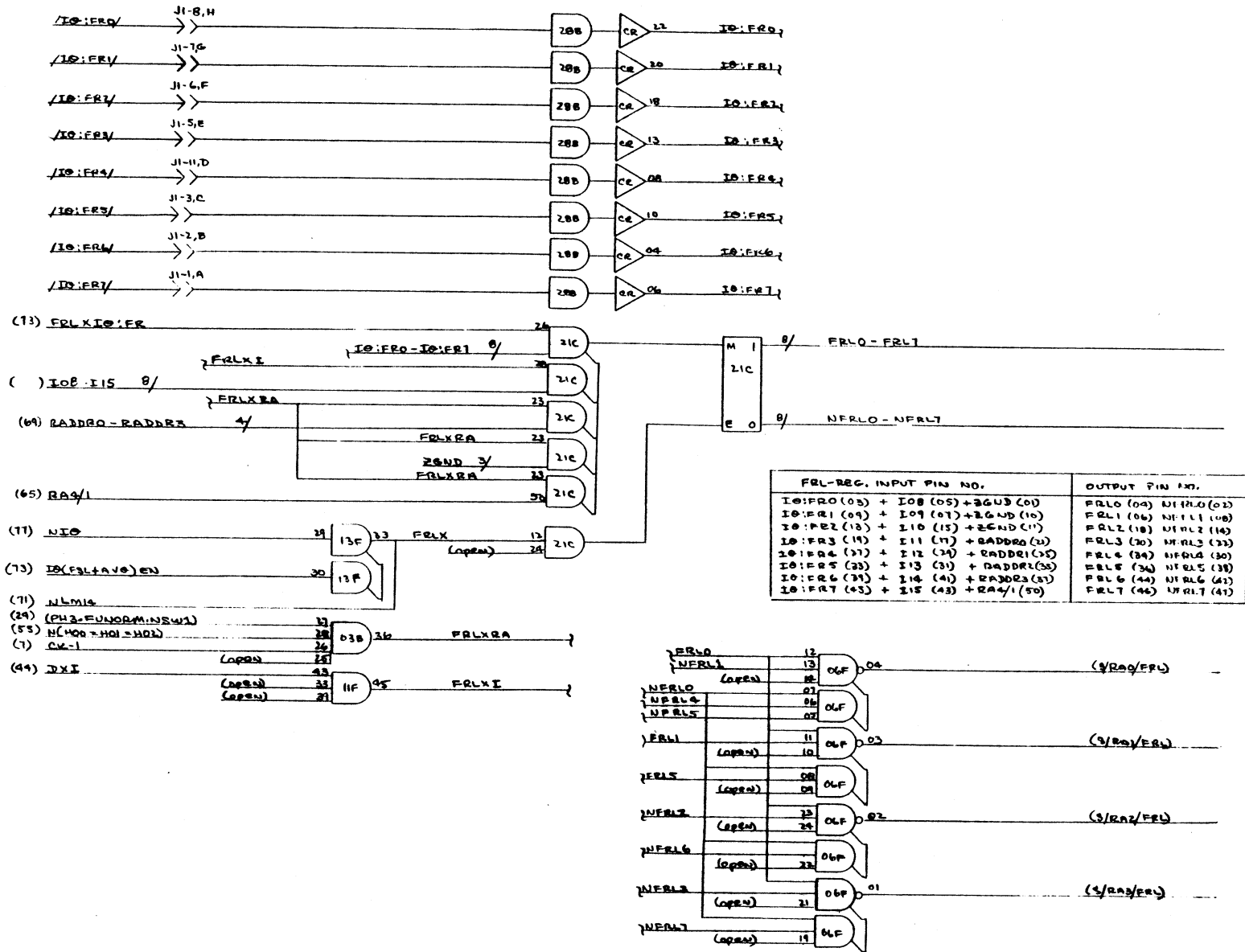


Figure 3-12. Logic Diagrams (sheet 75 of 91)

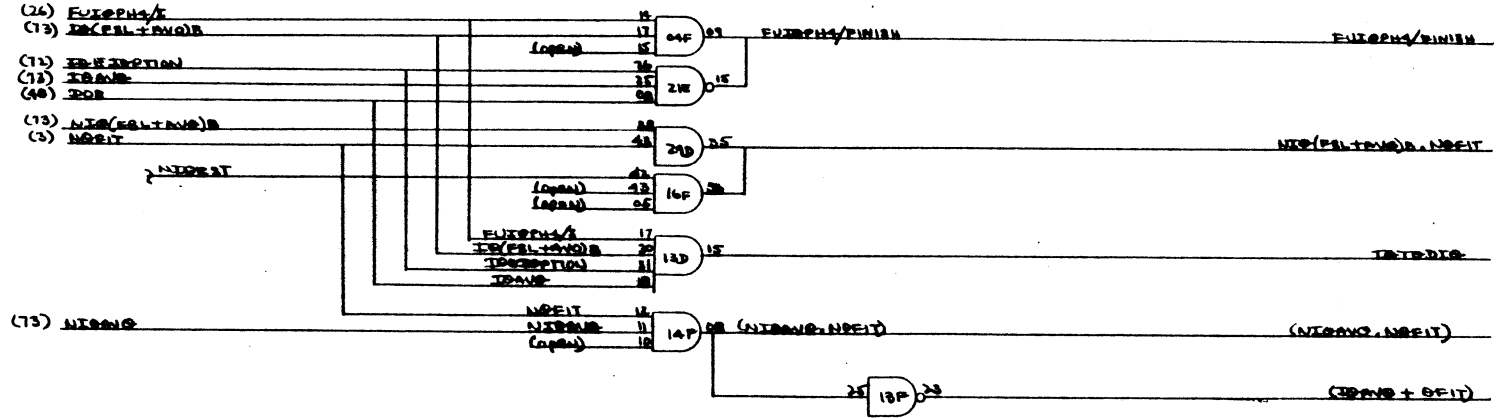
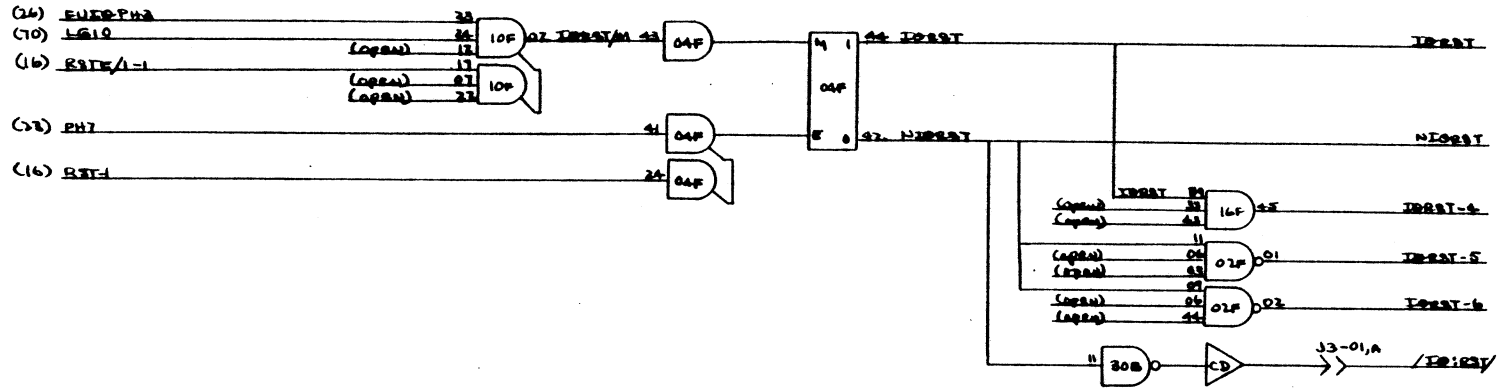
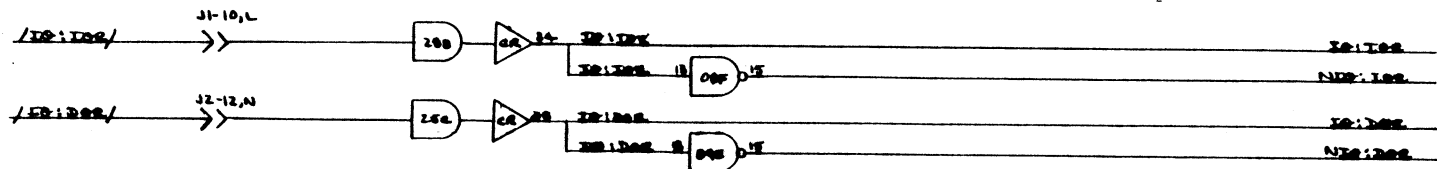


Figure 3-12. Logic Diagrams (sheet 76 of 91)

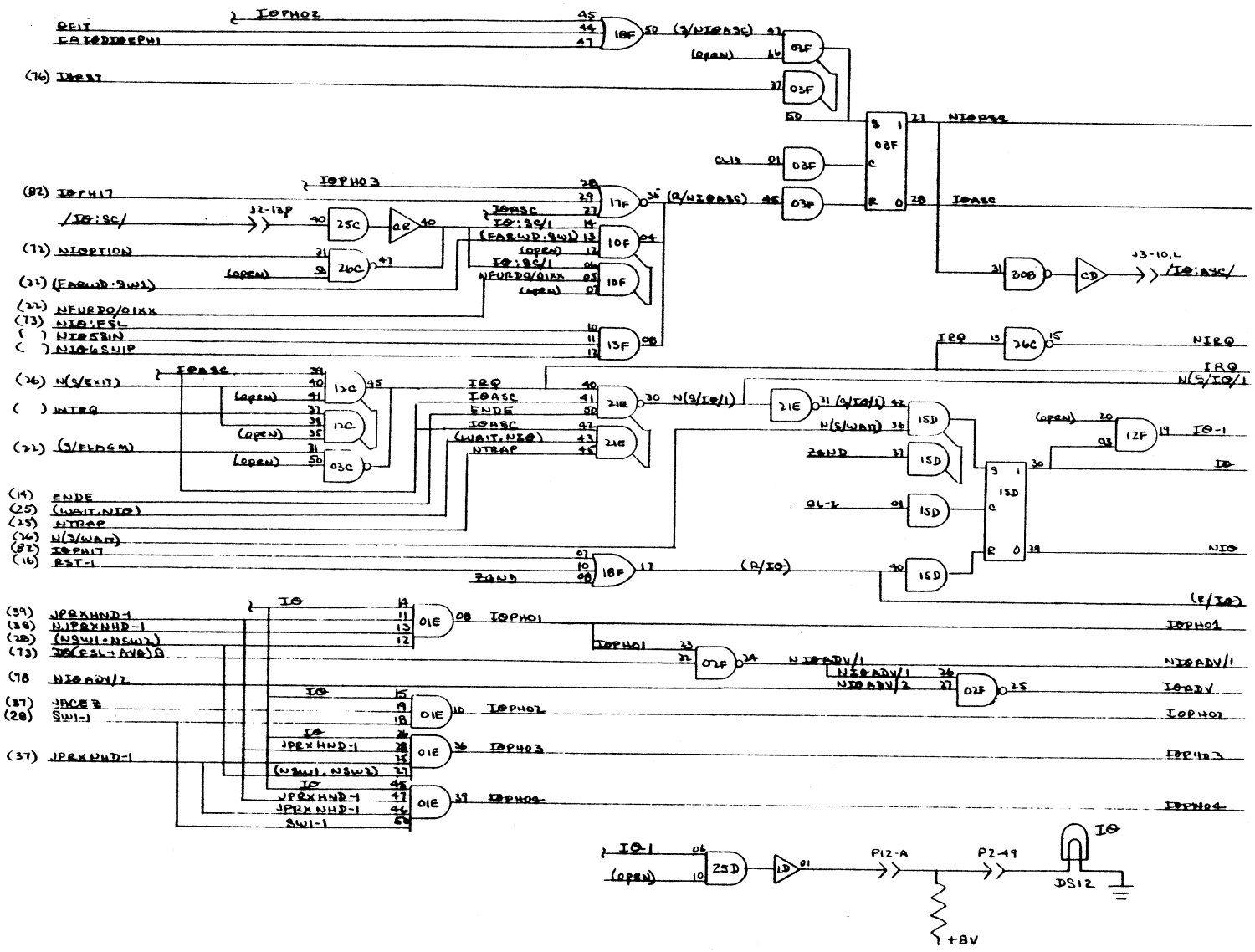


Figure 3-12. Logic Diagrams (sheet 77 of 91)

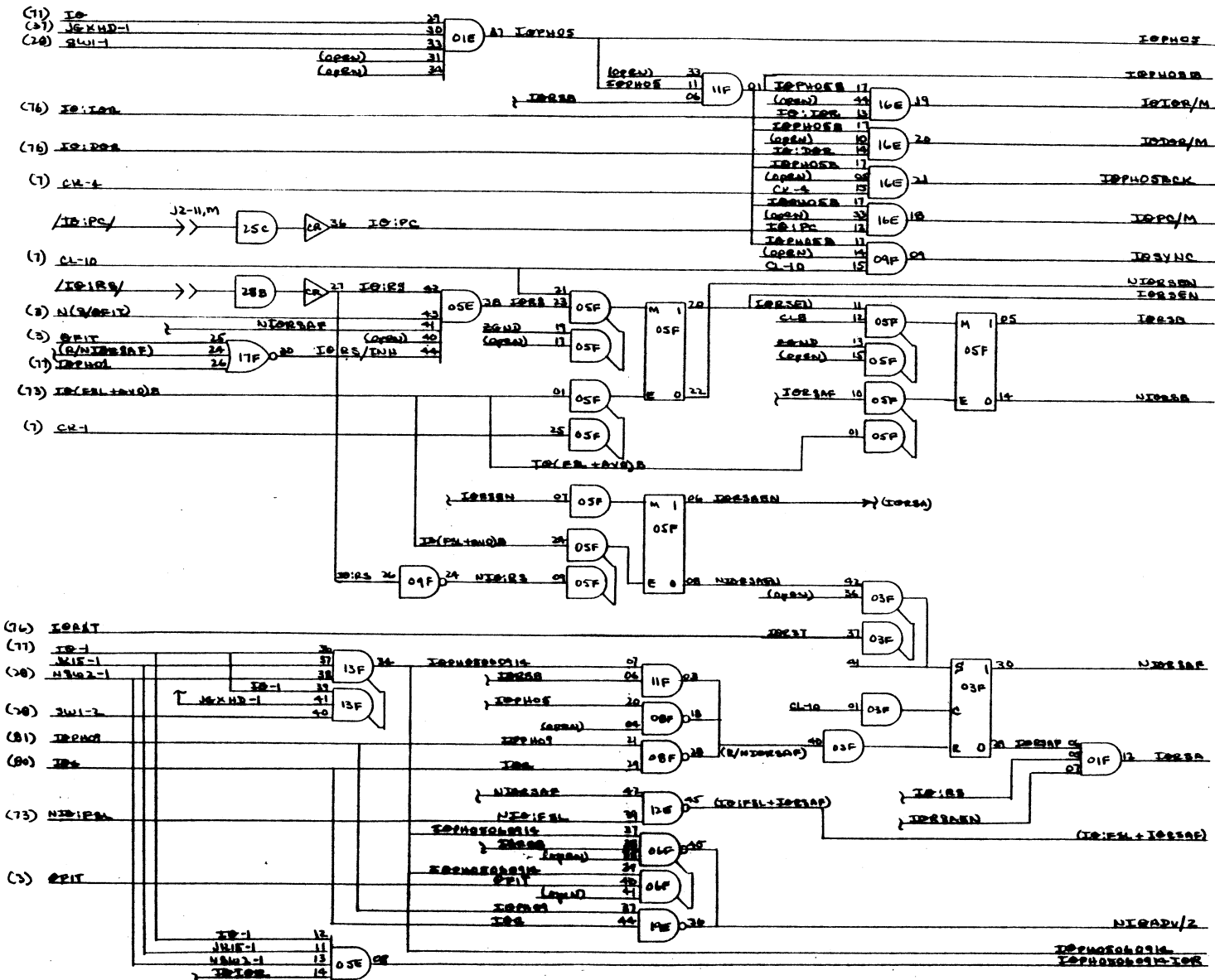


Figure 3-12. Logic Diagrams (sheet 78 c 1)

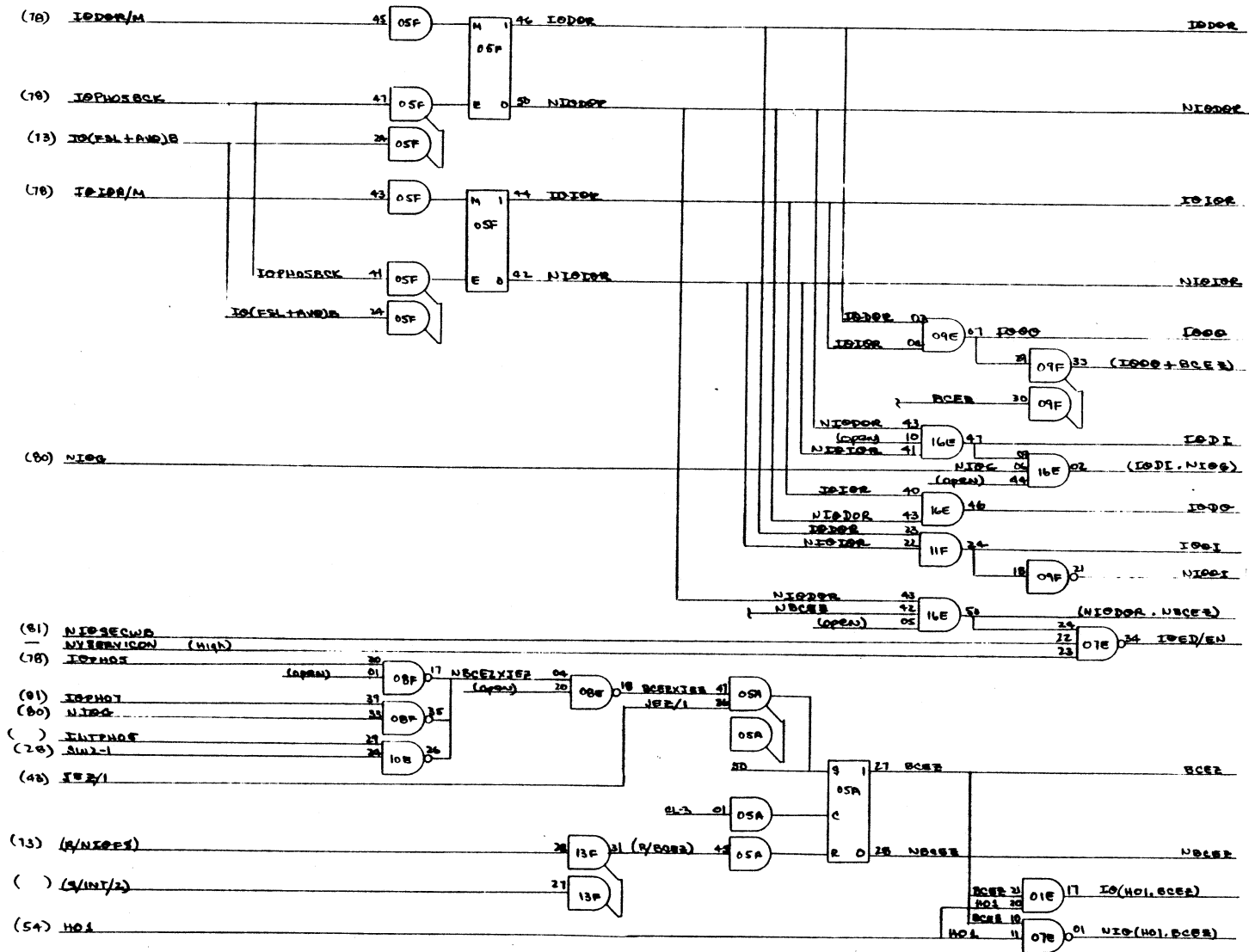


Figure 3-12. Logic Diagrams (sheet 79 of 91)

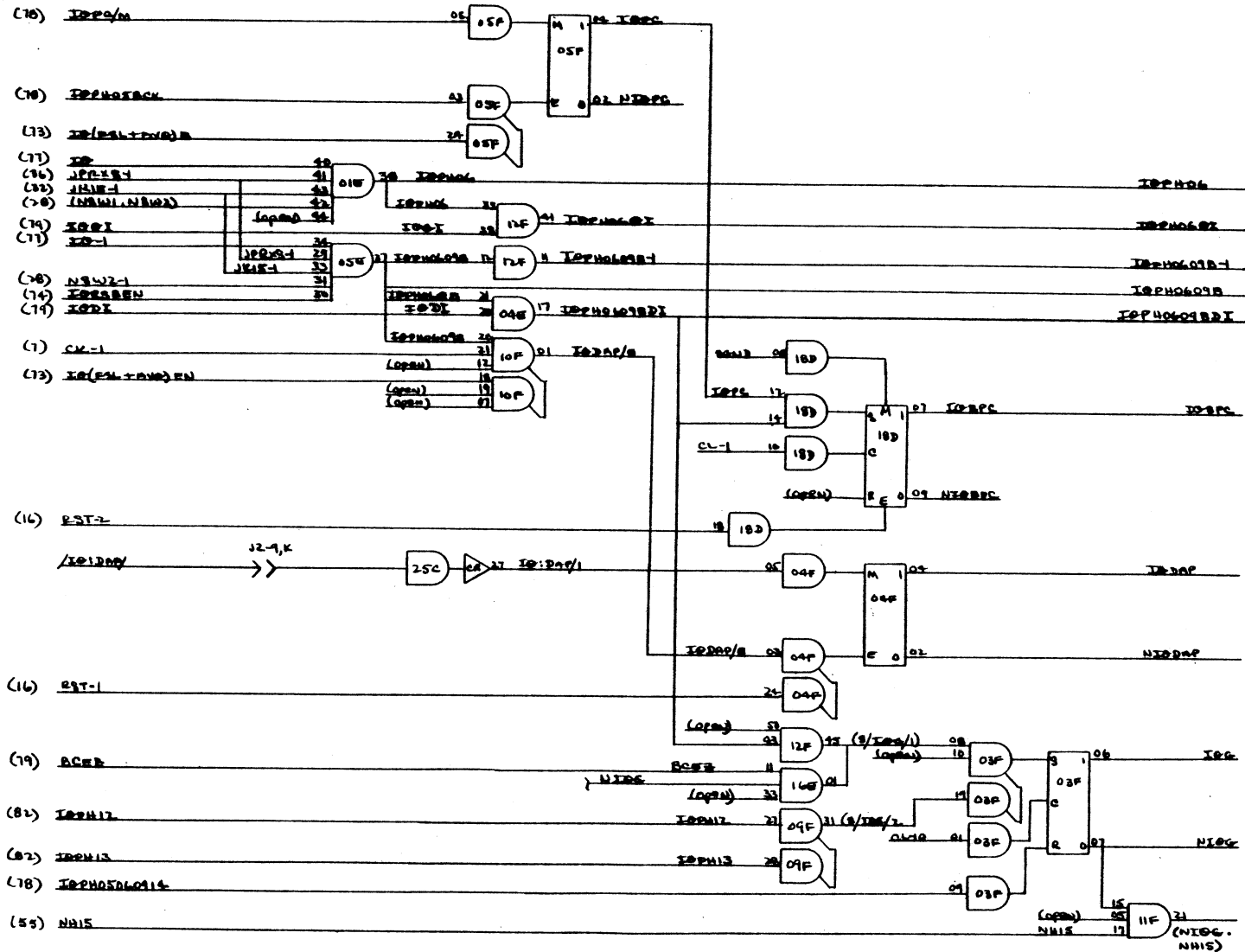


Figure 3-12. Logic Diagrams (sheet 80 of 91)

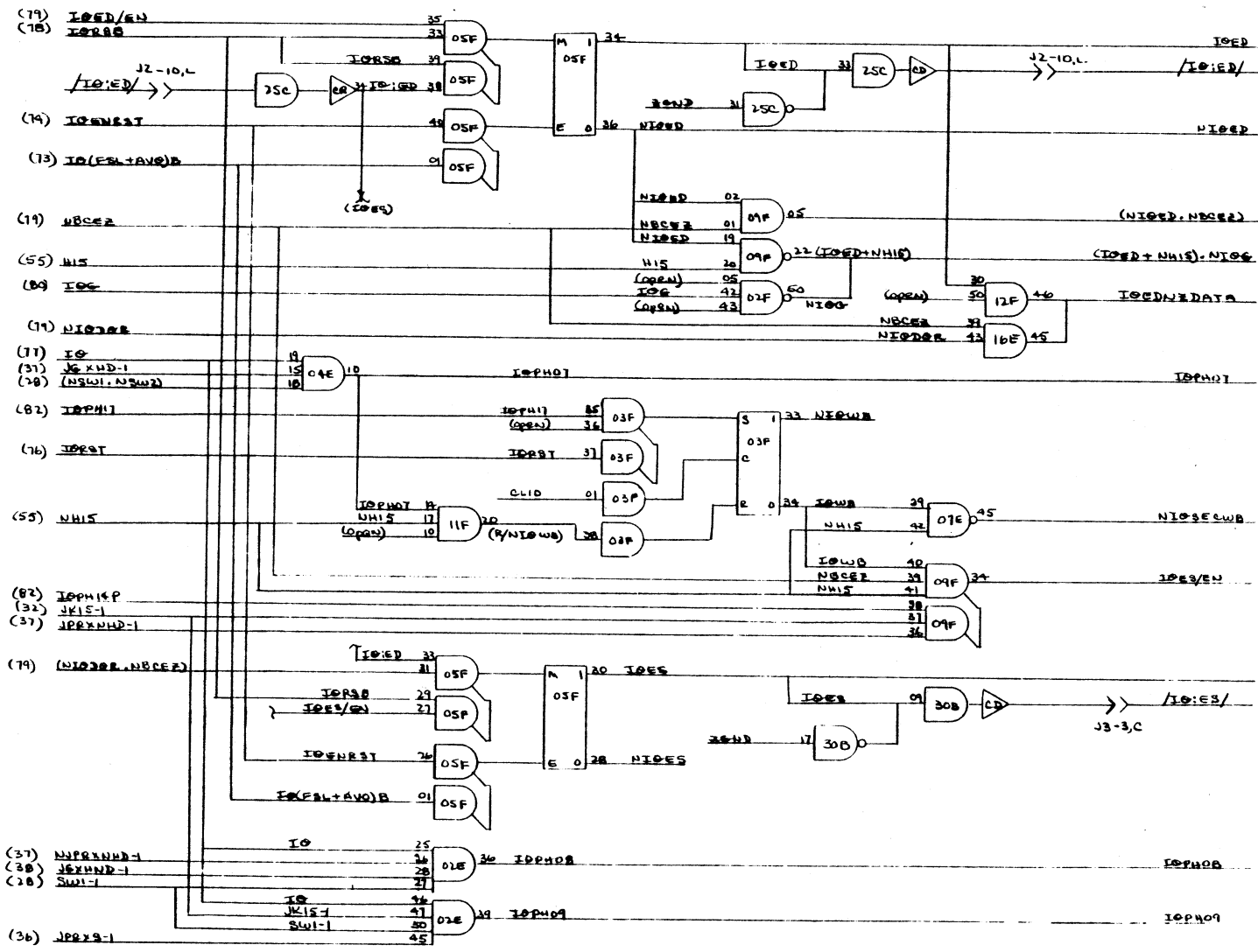


Figure 3-12. Logic Diagrams (sheet 81 of 91)

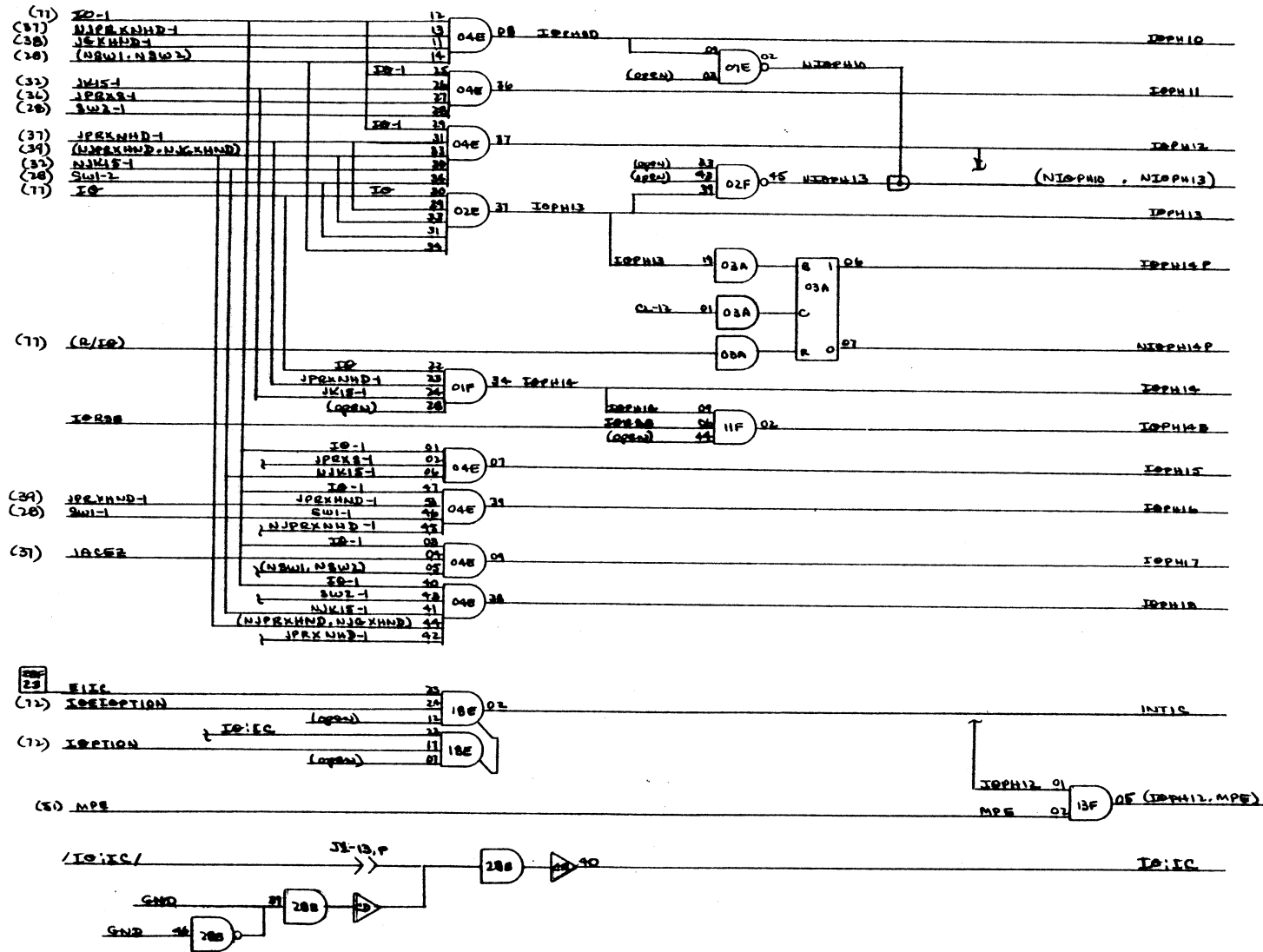


Figure 3-12. Logic Diagrams (sheet 82 of 100)

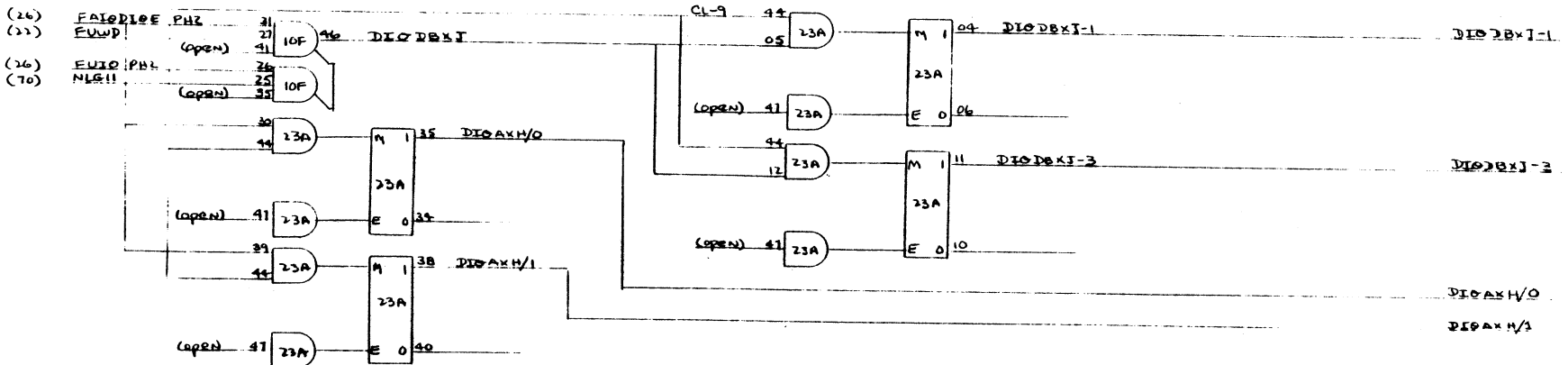
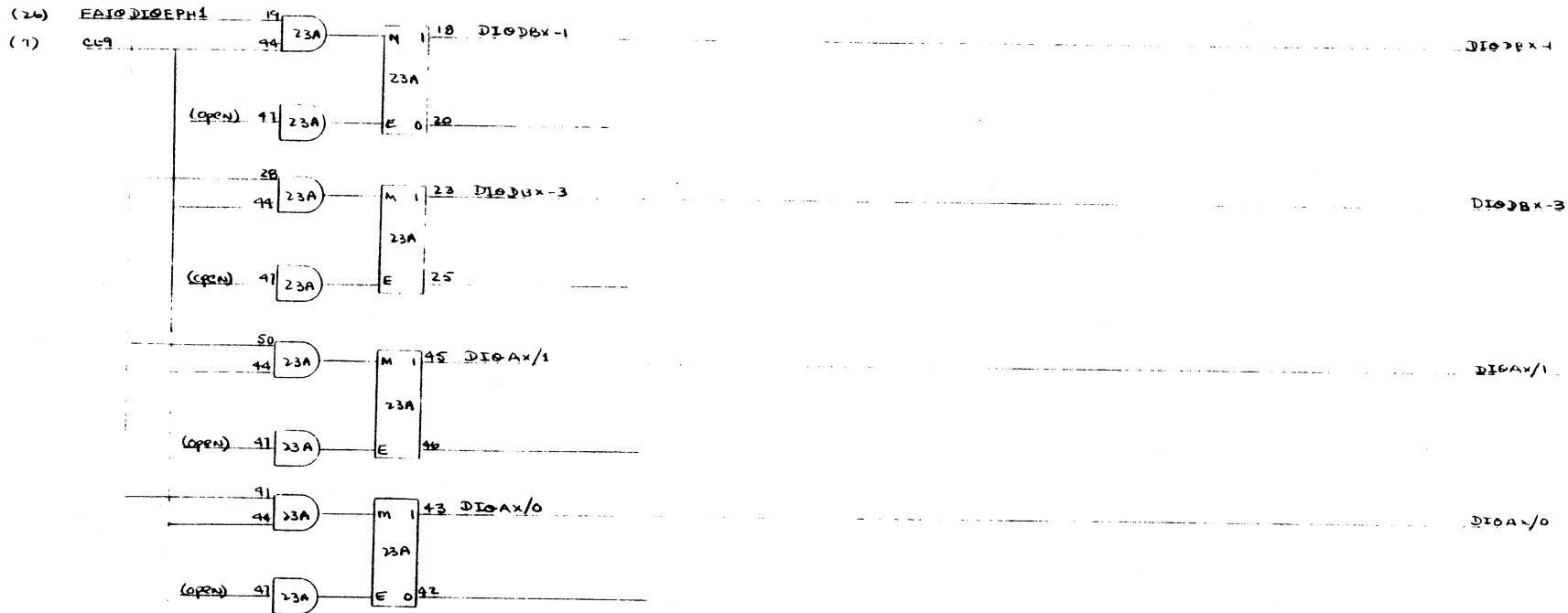
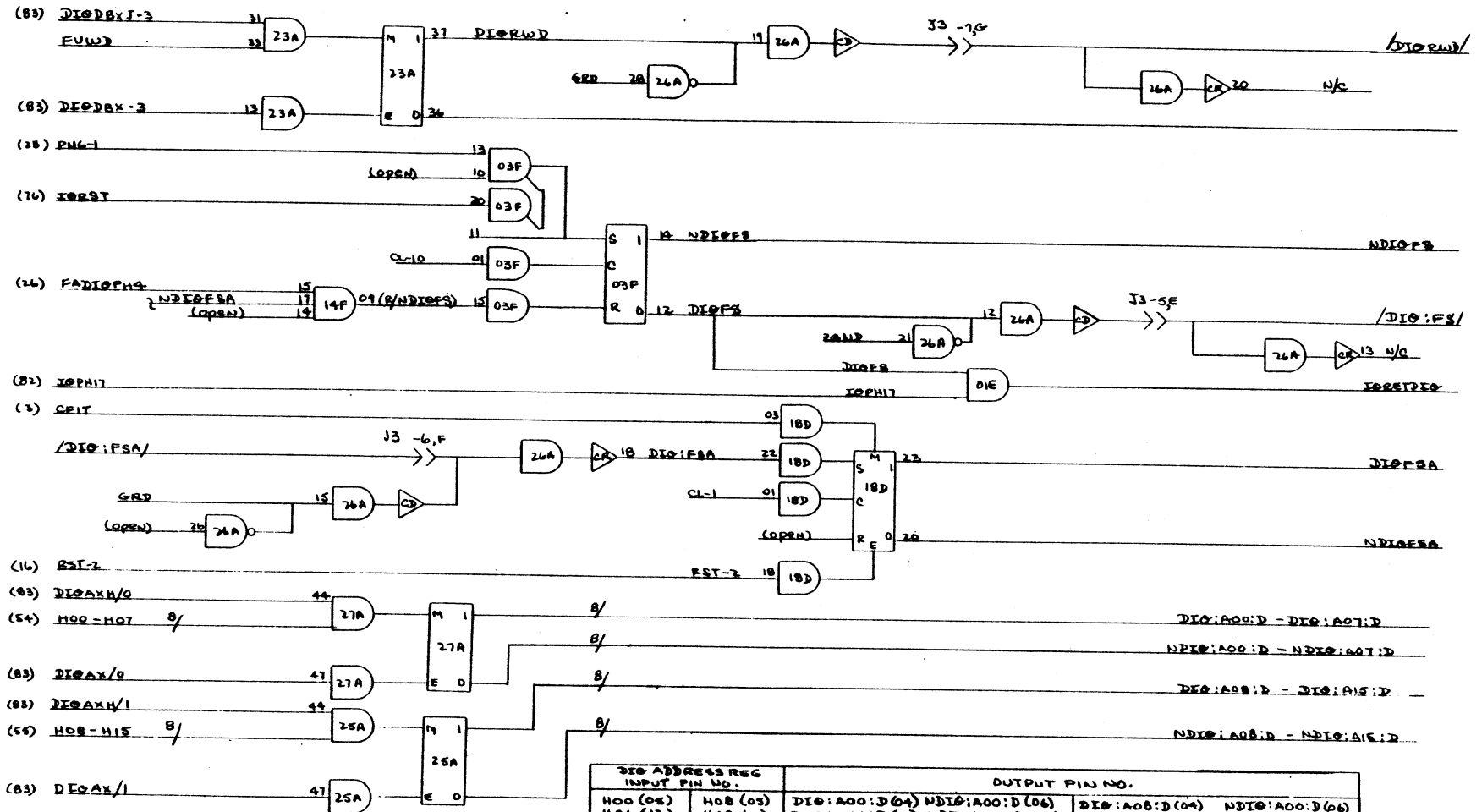


Figure 3-12. Logic Diagrams (sheet 83 of 91)



SIG ADDRESS REG INPUT PIN NO.		OUTPUT PIN NO.			
H00 (05)	H08 (05)	DI@:A00:D (04)	NDI@:A00:D (06)	DI@:A08:D (04)	NDI@:A08:D (06)
H01 (12)	H09 (11)	DI@:A01:D (11)	NDI@:A01:D (10)	DI@:A09:D (11)	NDI@:A09:D (10)
H02 (19)	H10 (19)	DI@:A02:D (18)	NDI@:A02:D (20)	DI@:A10:D (18)	NDI@:A10:D (20)
H03 (28)	H11 (28)	DI@:A03:D (25)	NDI@:A03:D (25)	DI@:A11:D (25)	NDI@:A11:D (25)
H04 (30)	H12 (30)	DI@:A04:D (35)	NDI@:A04:D (34)	DI@:A12:D (35)	NDI@:A12:D (34)
H05 (39)	H13 (39)	DI@:A05:D (40)	NDI@:A05:D (40)	DI@:A13:D (40)	NDI@:A13:D (40)
H06 (41)	H14 (41)	DI@:A06:D (43)	NDI@:A06:D (42)	DI@:A14:D (43)	NDI@:A14:D (42)
H07 (50)	H15 (50)	DI@:A07:D (45)	NDI@:A07:D (44)	DI@:A15:D (45)	NDI@:A15:D (44)
(27A)	(25A)	(27A)	(25A)	(27A)	(25A)

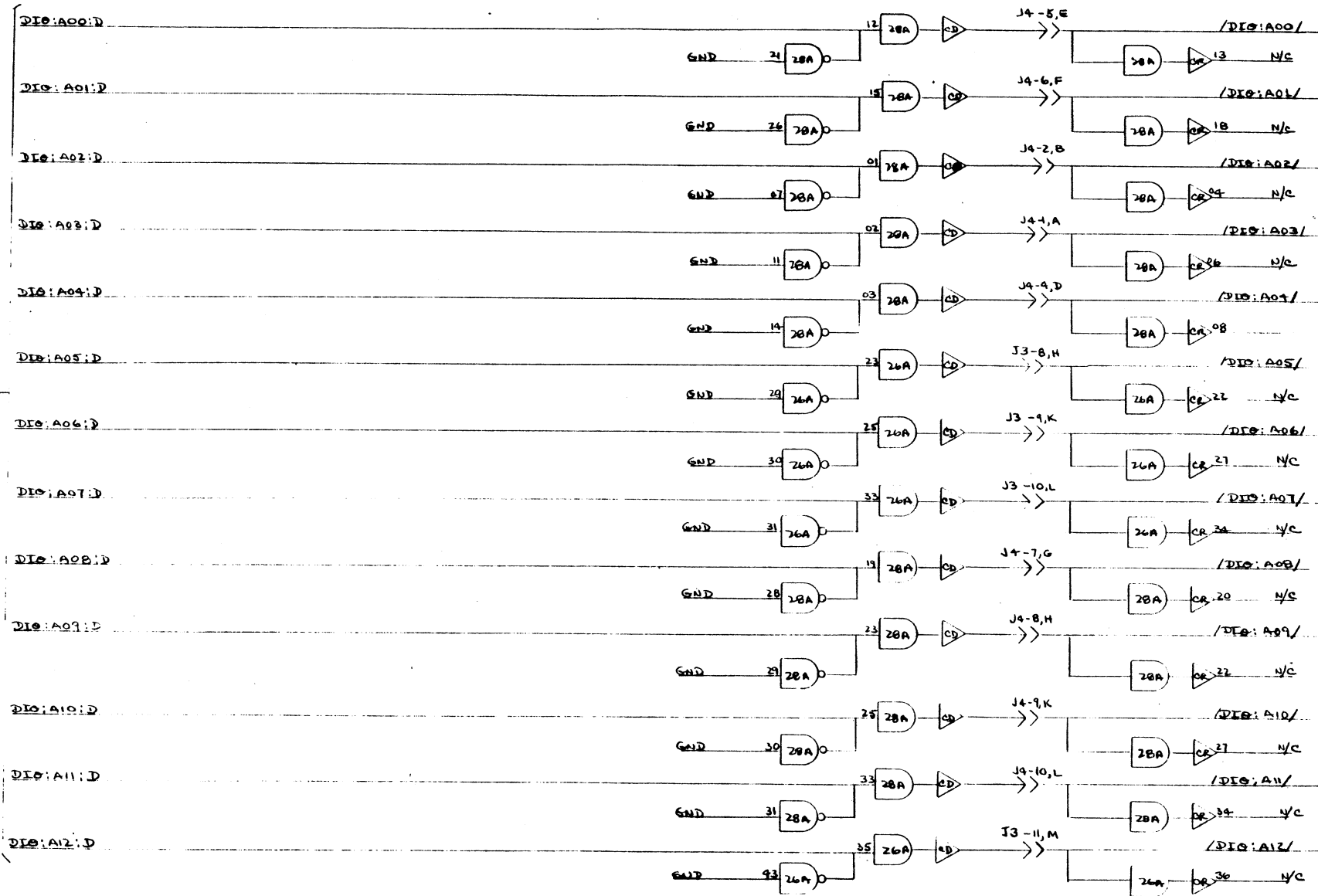
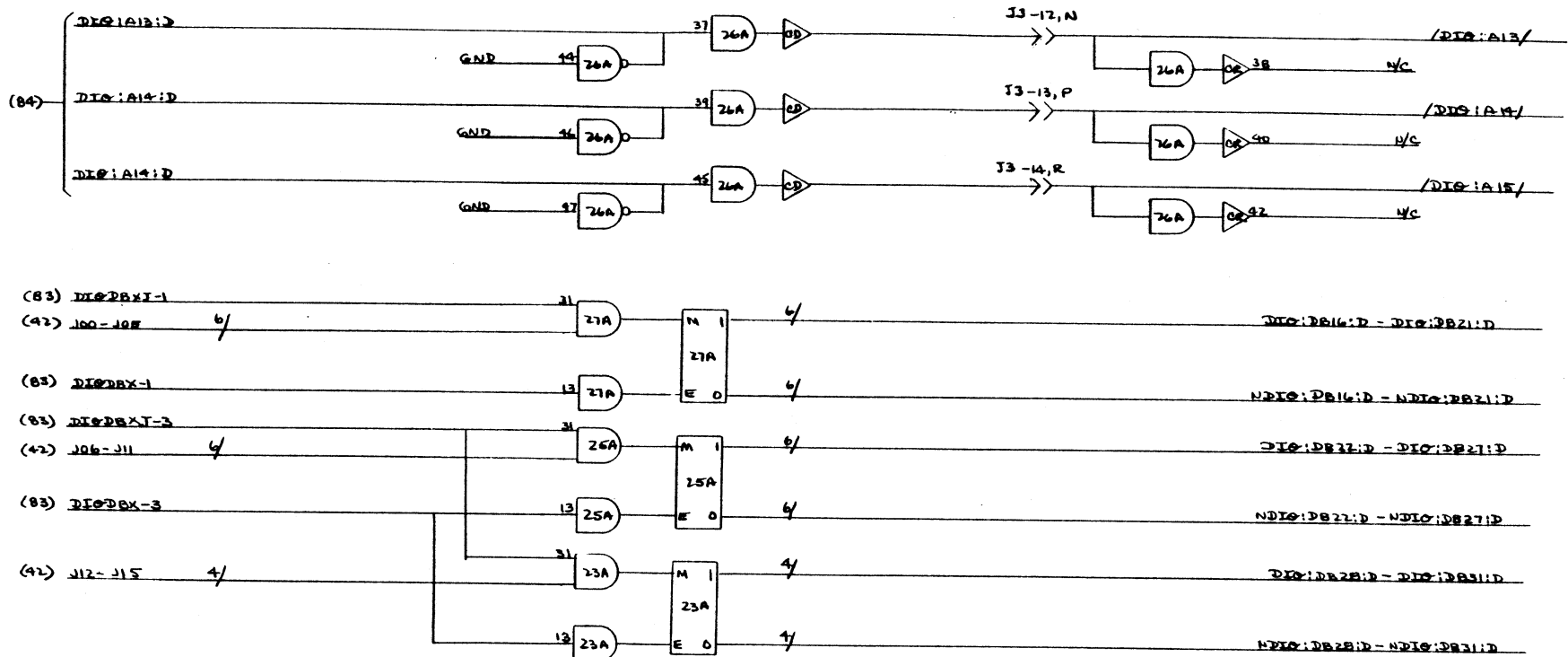


Figure 3-12. Logic Diagrams (sheet 85 of 91)



DIO DATA REG. INPUT PIN NO.	OUTPUT PIN NO.
J00 (01)	DIO:DB16:D (01) N DIO:DB16:D (
J01 (06)	DIO:DB17:D (06) N DIO:DB17:D (
J02 (14)	DIO:DB18:D (14) N DIO:DB18:D (
J03 (21)	DIO:DB19:D (21) N DIO:DB19:D (
J04 (29)	DIO:DB20:D (29) N DIO:DB20:D (
J05 (33)	DIO:DB21:D (33) N DIO:DB21:D (
J06 (01)	DIO:DB22:D (01) N DIO:DB22:D (
J07 (08)	DIO:DB23:D (08) N DIO:DB23:D (
J08 (14)	DIO:DB24:D (14) N DIO:DB24:D (
J09 (20)	DIO:DB25:D (20) N DIO:DB25:D (
J10 (29)	DIO:DB26:D (29) N DIO:DB26:D (
J11 (33)	DIO:DB27:D (33) N DIO:DB27:D (
J12 (01)	DIO:DB28:D (01) N DIO:DB28:D (
J13 (08)	DIO:DB29:D (08) N DIO:DB29:D (
J14 (14)	DIO:DB30:D (14) N DIO:DB30:D (
J15 (20)	DIO:DB31:D (20) N DIO:DB31:D (

Figure 3-12. Logic Diagrams (sheet 86 of 1)

(86)

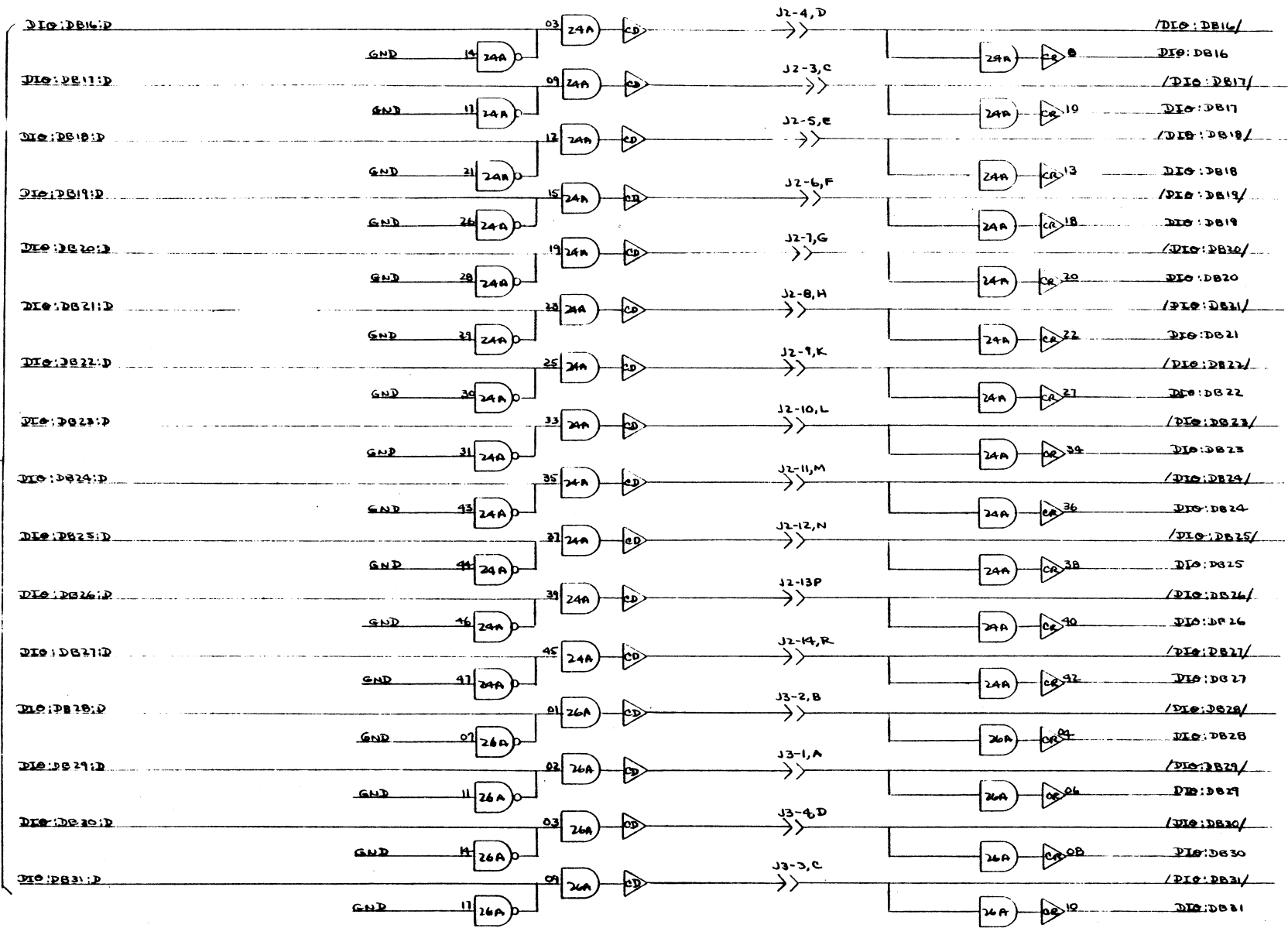


Figure 3-12. Logic Diagrams (sheet 87 of 91)

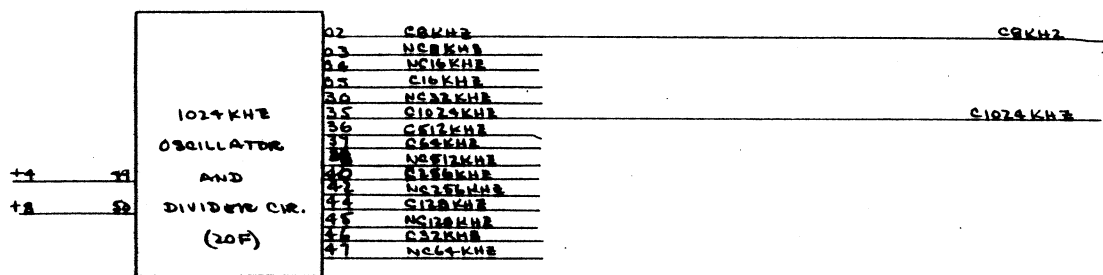
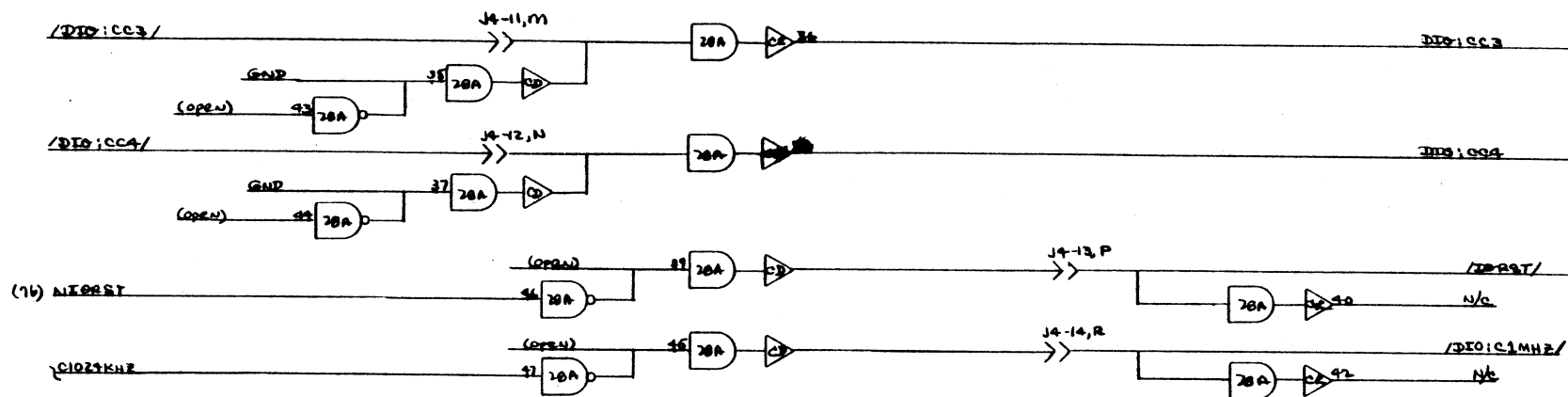


Figure 3-12. Logic Diagrams (sheet 88 of 91)

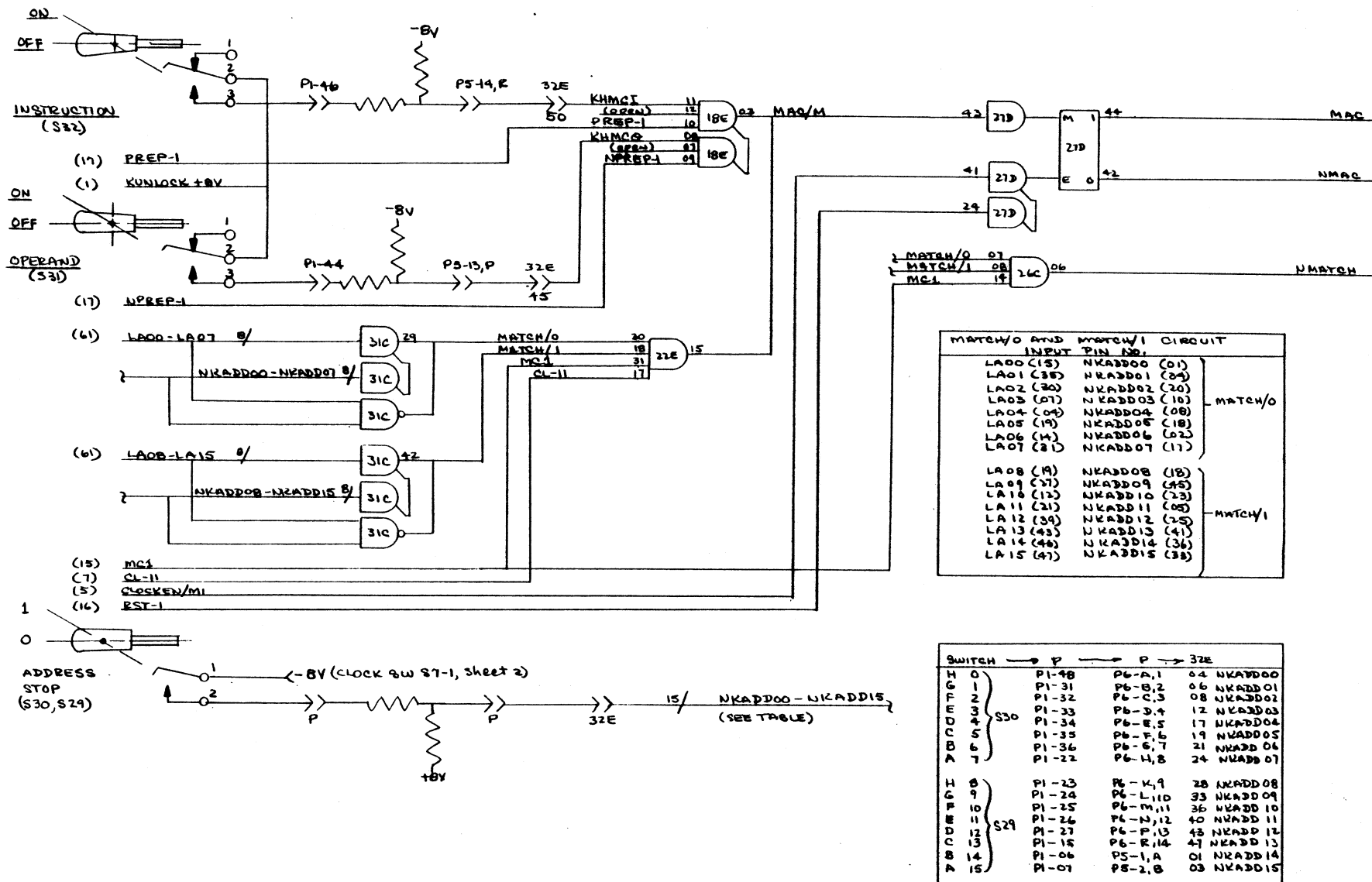


Figure 3-12. Logic Diagrams (sheet 89 of 91)

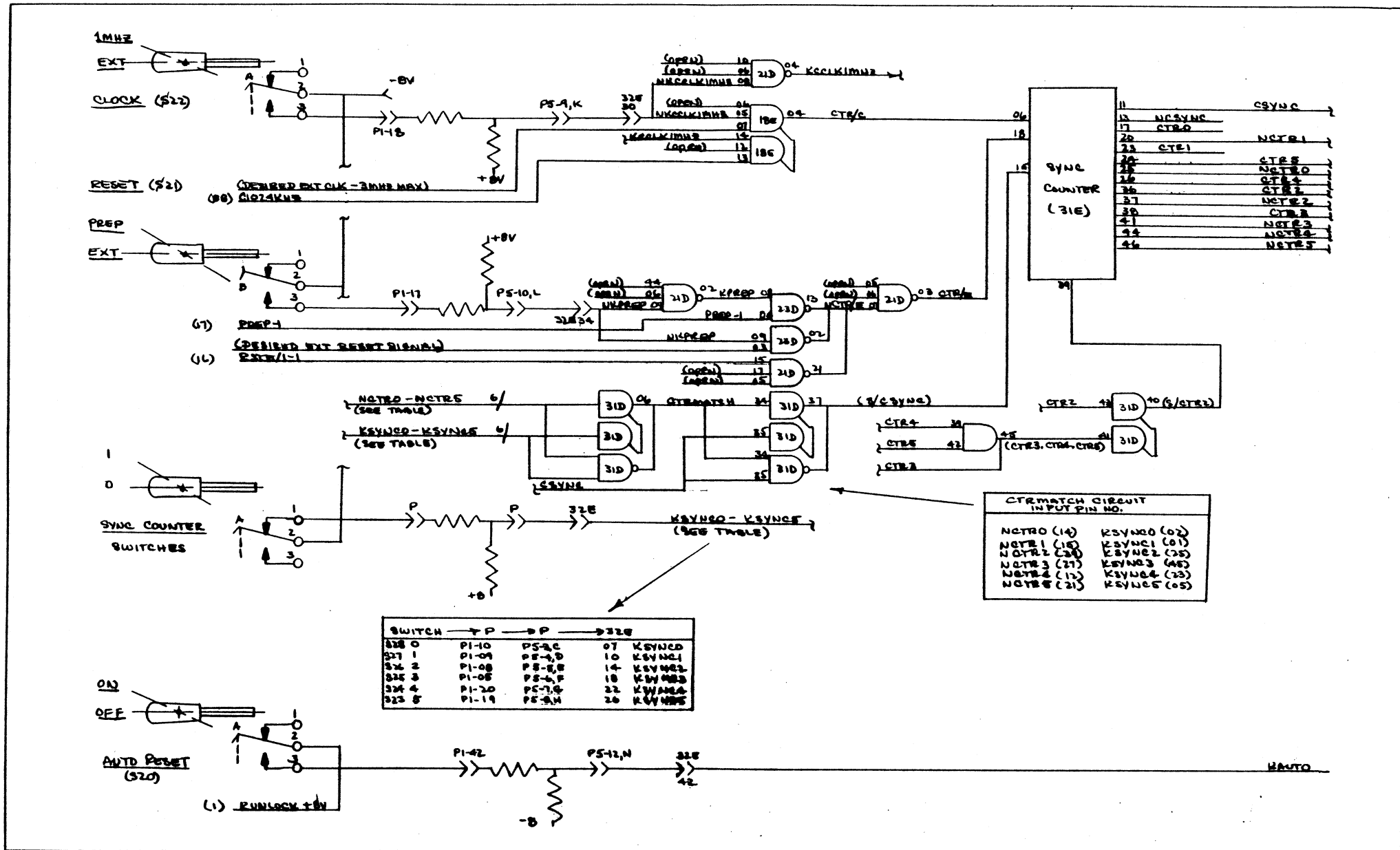


Figure 3-12. Logic Diagrams (sheet 90 of 91)

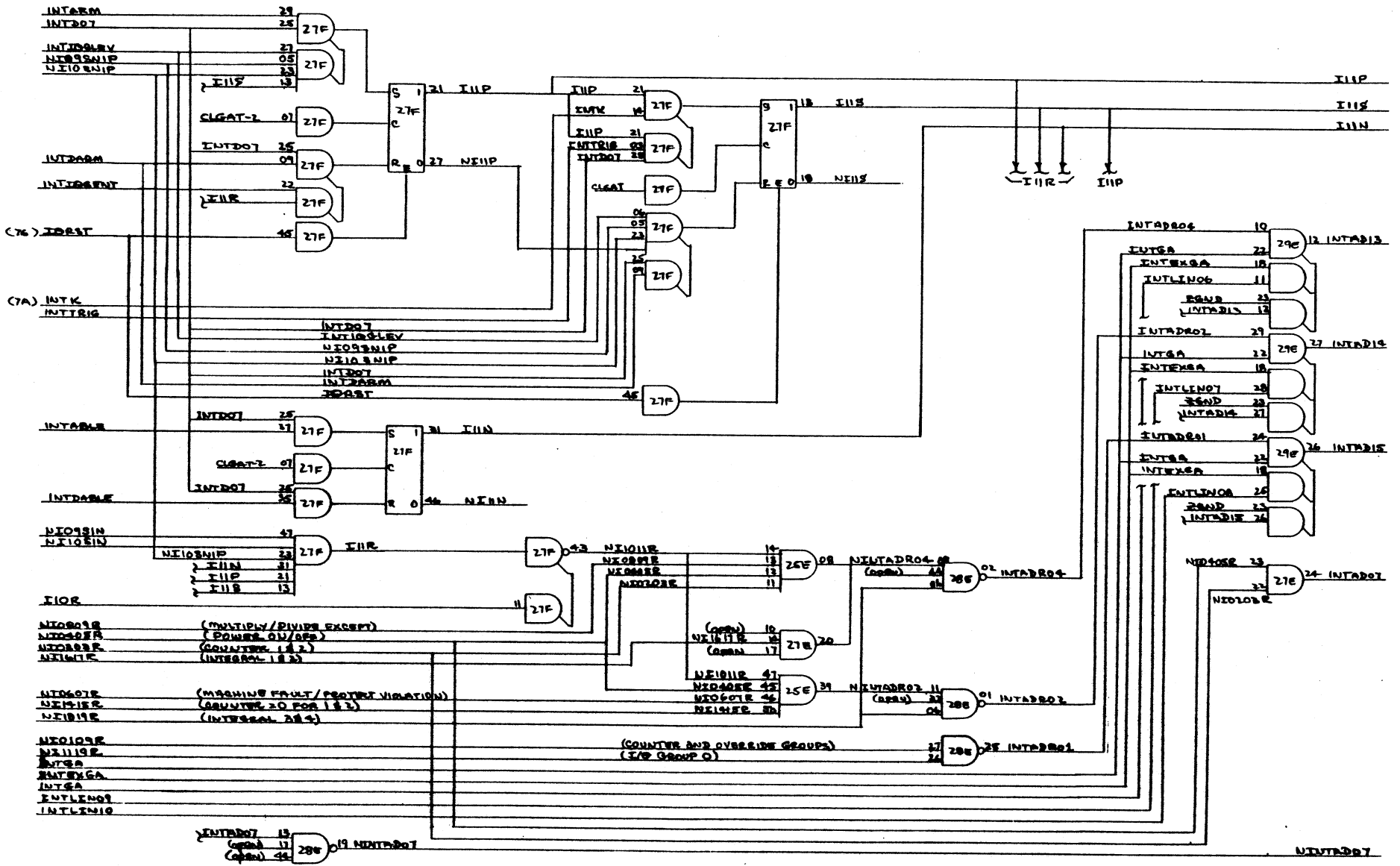


Figure 3-12. Logic Diagrams (sheet 91 of 91)