V-73

Varian 73 System Handbook

V-73

Varian 73 System Handbook

VARIAN 73

SYSTEM

HANDBOOK

Specifications subject to change without notice. Address comments regarding this handbook to Varian Data Machines, Publications Department, 2722 Michelson Drive, Irvine, California, 92664.

varian data machines / a varian subsidiary
2722 michelson drive / p.o. box e / irvine / california / 92664

98 A 9906 011

June 1975

INTRODUCING
THE VARIAN 73 SYSTEM

The Varian 73 System is a system-oriented, general-purpose minicomputer. It is designed for maximum performance in instrumentation, data acquisition, and communications systems, making it ideal for a variety of scientific, commerical, and industrial applications, including business data processing, industrial control, data communications, scientific research, and education.

The simplest computer system, the computer and a Teletype for input/output, is suitable for the research laboratory for simple computations. This basic system can be enhanced by the addition of peripherals that present results in graphic form, i.e., digital plotters and oscilloscope display systems. High-speed paper tape equipment (reader and punch) added to the basic system increases input/output efficiency and makes it possible to record computation results in a form suitable for later analysis.

Business and accounting applications require a large data base that can be efficiently stored, easily accessed, and promptly displayed or printed out. Adding disc memory and/or magnetic tape units (for bulk data storage) and a line printer (for output) to the basic computer-Teletype system meets these requirements. This system can be made even more flexible and efficient with the use of punched-card and/or high-speed paper-tape equipment.

Industrial process control is another important computer use. This application presents situations that require frequent sampling and interpretation of data as materials are being processed and as goods are being manufactured. Here again the basic computer-Teletype system comprises the heart of the system. Analog, digital, and relay I/O devices can be connected on the I/O bus for the input of signals from the processing/manufacturing area and for the output of signals to control devices or displays. Secondary data storage devices make it possible to record data for allied operations, i.e., accounting, production control, etc. If additional operator input is required, several Teletypes can be interfaced on the I/O bus.

The computer and Teletype can also be used to integrate and unify a communications network by adding a Varian data communications multiplexor. Used as a data concentrator, this expanded system links and switches variable-speed modems and communications lines, thus, adapting them to a variety of communications devices and transfer rates. With a second addition of disc or magnetic tape units, the expanded system can be used as a preprocessor that can organize data for efficient, time-shared processing in larger computer systems.

The Varian 73 features:

- Efficient Operation -- Operates on 16-bit words with a full-cycle execution time of 660 nanoseconds.
- Large Instruction Set -- Recognizes 160 arithmetic, decision, and control instructions, many of which can be

microcoded to extend the effective instruction repertoire into the hundreds.

- Modular Core Memory -- Can be expanded in 8,192-word (8K) increments, from the minimum of 8K to a maximum of 256K with the memory expansion chassis and memory map option. Cycle time is 660 nanoseconds.

- Modular Semiconductor Memory -- Can be expanded in 8,192-word (8K) increments, from a minimum of 8K to a maximum of 256K with the memory expansion chassis and memory map option. Cycle time is 330 nanoseconds.

- Modular 16K Core Memory -- Can be expanded in 16,384-word (16K) increments, from a minimum of 16K to a maximum of 64K. This low cost memory is a single-port memory with a cycle time of 1200 nanoseconds.

- Multiple Addressing Modes -- Including direct, multilevel indirect, immediate, indexed/indirect, relative, and extended with preindexing and postindexing.

- Effective Input/Output -- Includes a full complement of peripheral controllers and devices and three types of I/O operations: program-controlled, direct memory access, and priority memory access.

- Extensive Software -- The available package includes the DAS assemblers; binary load/dump (BLD II program; AID II for debugging; MAINTAIN III for troubleshooting; FORTRAN IV; BASIC; the business-oriented RPG IV compiler; the Varian Master Operating System (MOS); a comprehensive mathematical and utility subroutine library; the VORTEX and VORTEX II real-time, multitasking operating system; a Basic Real-Time Executive (BEST); and PERT.

Scope of the Handbook

This Handbook explains the Varian 73 system and its capabilities. It contains both hardware and software information. The organization of the Handbook allows the experienced programmer or systems engineer to refer directly to the material that will aid him in understanding specific, task-related elements of the system.

Varian Data Machines invites suggestions on the contents of the Handbook so future revisions can be more useful to you - the reader. Direct any comments to the Publications Department, Varian Data Machines, 2722 Michelson Drive, Irvine, California, 92664.

Support Documentation

To ensure full utilization of the Varian 73 system, complete documentation (including this Handbook) is provided with each system.

Documents such as logic diagrams, schematics, and parts lists are supplied in a System Maintenance Manual. This manual is assembled when the equipment is shipped and reflects the configuration of the specific system.

Other documentation provided with Varian 73 systems includes: technical manuals for major computer components and peripheral devices; the Varian 73/620 Test Programs Manual, which describes operation of the MAINTAIN III system for diagnosing hardware malfunctions; programming manuals explaining software systems not discussed in detail in this Handbook, e.g., FORTRAN IV, BASIC, MOS, VORTEX, etc.; and manuals and other documents relating to equipment designed to meet special, nonstandard applications requirements.

TABLE OF CONTENTS

INTRODUCING THE VARIAN 73 SYSTEM

SECTION 1 - VARIAN 73 SYSTEM COMPUTER

SECTION 2 - PROCESSOR

SECTION 3 - CORE MEMORY

SECTION 4 - SEMICONDUCTOR MEMORY

SECTION 5 - OPTIONS

SECTION 6 - INPUT/OUTPUT SYSTEM

SECTION 7 - PERIPHERALS AND I/O INTERFACES

CONTENTS

SECTION 8 - SYSTEM CONFIGURATIONS

SECTION 9 - INSTALLATION

SECTION 10 - OPERATION

SECTION 11 - MAINTENANCE

SECTION 12 - DATA AND INSTRUCTION FORMATS

SECTION 13 - ADDRESSING MODES

SECTION 14 - INSTRUCTION SET

SECTION 15 - VARIAN 73 DAS ASSEMBLERS

SECTION 16 - BINARY LOAD/DUMP PROGRAM

CONTENTS

SECTION 17 - AID II DEBUGGING PROGRAM

SECTION 18 - SOURCE PROGRAM EDITOR

SECTION 19 - VORTEX/VORTEX II

SECTION 20 - MASTER OPERATING SYSTEM

SECTION 21 - BEST

SECTION 22 - FORTRAN IV

SECTION 23 - RPG IV (Report Program Generator)

CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS *(continued)*

LIST OF TABLES

LIST OF TABLES *(continued)*

LIST OF TABLES *(continued)*

SECTION 1 - VARIAN 73 SYSTEM COMPUTER

General Description

The Varian 73 system (figure 1-1) has been designed with flexibility as the keystone. The Varian 73 system offers the capability to configure systems with a wide range of application requirements, modular expansion and open-ended system growth, microprogramming for control, adaptability to changing technology, reliability, easy maintenance, and compatibility with existing 620 series programs and peripherals. It is designed for maximum performance in instrumentation, data acquisition, and communications systems, making it ideal for a variety of scientific, commercial, and industrial applications.

The computer processes 16-bit words in a full cycle memory time of 330 nanoseconds (semiconductor memory) and 660 nanoseconds (8K core memory). Both are dual port memories.

Optional 16K (16,384-word module) core memories are also available with cycle times of 990 and 1200 nanoseconds. These are single-port memories.

The expandable, random-access, 3W / 3D, magnetic core memory can be expanded in 8,192 word (8K) increments; from a minimum of 8K to a maximum of 256K with memory expansion chassis and memory map.

The random-access semiconductor memory can be expanded from 8K to 256K words (maximum) in 8K word increments. This semi-conductor memory includes a programmed Data Save feature that sustains memory content during prolonged periods of low primary ac power operation or primary ac power failure.

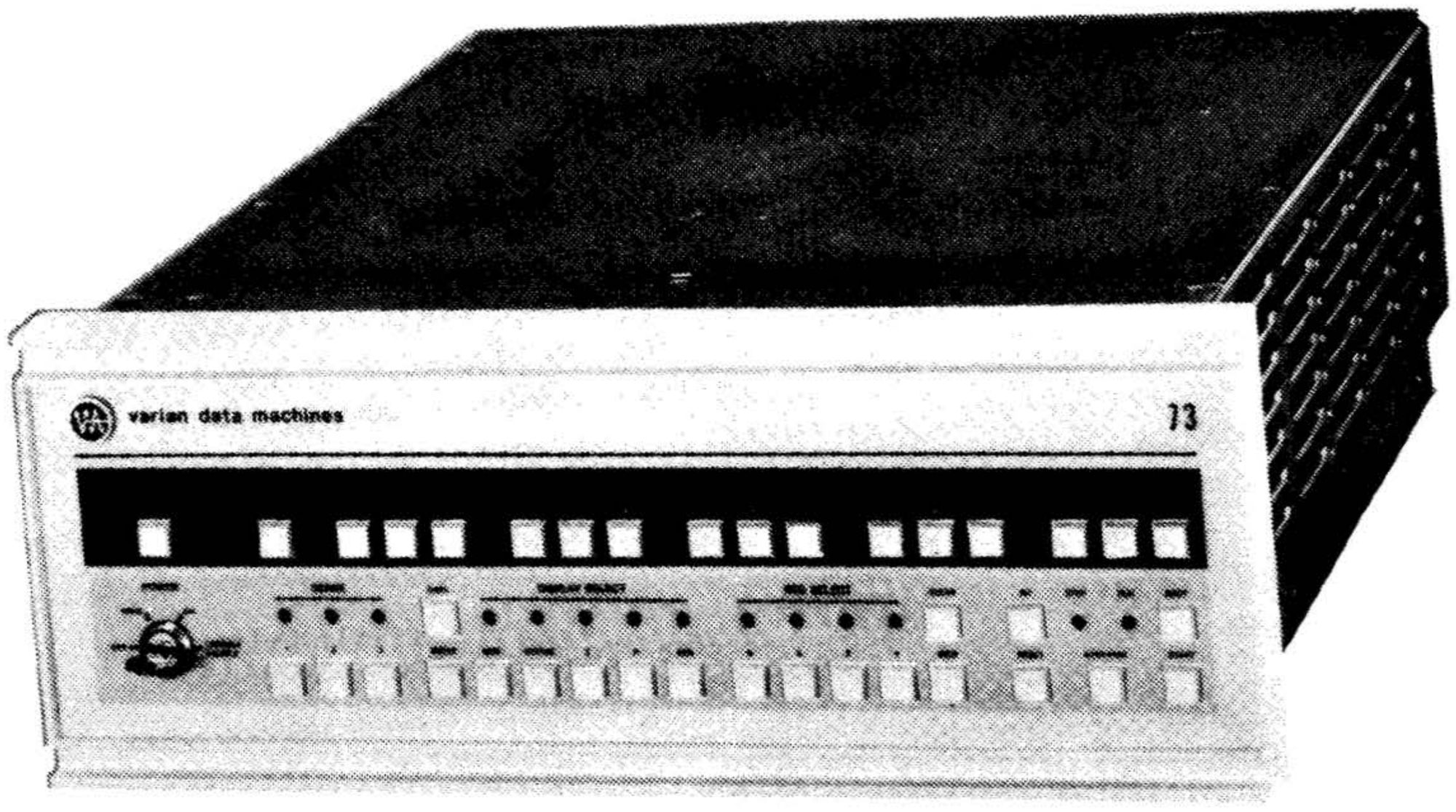
The instruction set of the Varian 73 system comprises 160 instructions, many of which can be microcoded to extend the effective repertoire to several hundred instructions. The V75 instruction set option is also available with Varian 73 systems.

The central processing unit features a general purpose set of registers, 16-bit wide data paths, arithmetic and logical function generators, and data path selection logic under control of microprogramming firmware stored in a read only memory or writable control store. The processor, while completely general purpose, is offered in a variety of configurations for the widest possible range of applications. Also included, is a convenient full programmer's console or a modified operator's console.

The Varian 73 maintains software compatibility with the 620 series computers through microprogramming. Increased performance is obtained via a faster processing system. This compatibility includes direct, multilevel indirect, immediate, preindexing and postindexing, relative, and extended addressing modes.

The power supply is housed in a separate chassis and supplies all necessary power for the mainframe and memory expansion, up to 64K words depending on memory type. I/O expansion outside the mainframe is powered by a separate supply. A remote control turn-on from the console is included.

The standard mainframe chassis is 7-inches (7 card slots) high. An optional 14-inch (17 card slots) chassis is also available. The flexible design and packag-



VHT-181

Figure 1-1. The Varian 73 Mainframe (7-Inch Chassis)

ing techniques allow the multilayer printed circuit (PC) boards to be mounted in any desired slots. Boards which may be included in the mainframe chassis are the processor, option board, core memories, semiconductor memories, writable control store, and memory map. The option board provides I/O bus control logic (with DMA), Teletype controller (TC), power failure/restart (PF/R), real-time clock (RTC), and memory protection (MP) as standard. Memory parity, priority memory access (PMA), and RTC special configurations, are available as options on the option board. The mainframe chassis provides the capability for I/O and memory expansion.

The 14-inch mainframe chassis with a processor and option board can contain any size semiconductor or core memory (in 8K or 16K increments) up to 64K depending on memory type. Eight remaining slots may be used for writable control store, memory map, etc.

Systems options include: priority interrupt module (PIM), buffer interlace controller (BIC), and block transfer controller (BTC). The PIM establishes eight levels of interrupt requests on the I/O bus in order of priority. The BIC implements the direct memory access (DMA) capabilities of the basic computer, permitting cycle-stealing I/O data transfers between memory and peripheral controllers at rates of up to 361,800 words per second. The BTC implements automatic data transfers between peripheral controllers and memory via the priority memory access (PMA).

The I/O expansion chassis can accommodate up to 22 etched-circuit controller cards, up to eight wire-wrap controller cards, or a combination of both. It connects to the I/O port at the bottom rear of the mainframe chassis via flat I/O cable and paddle-board connectors.

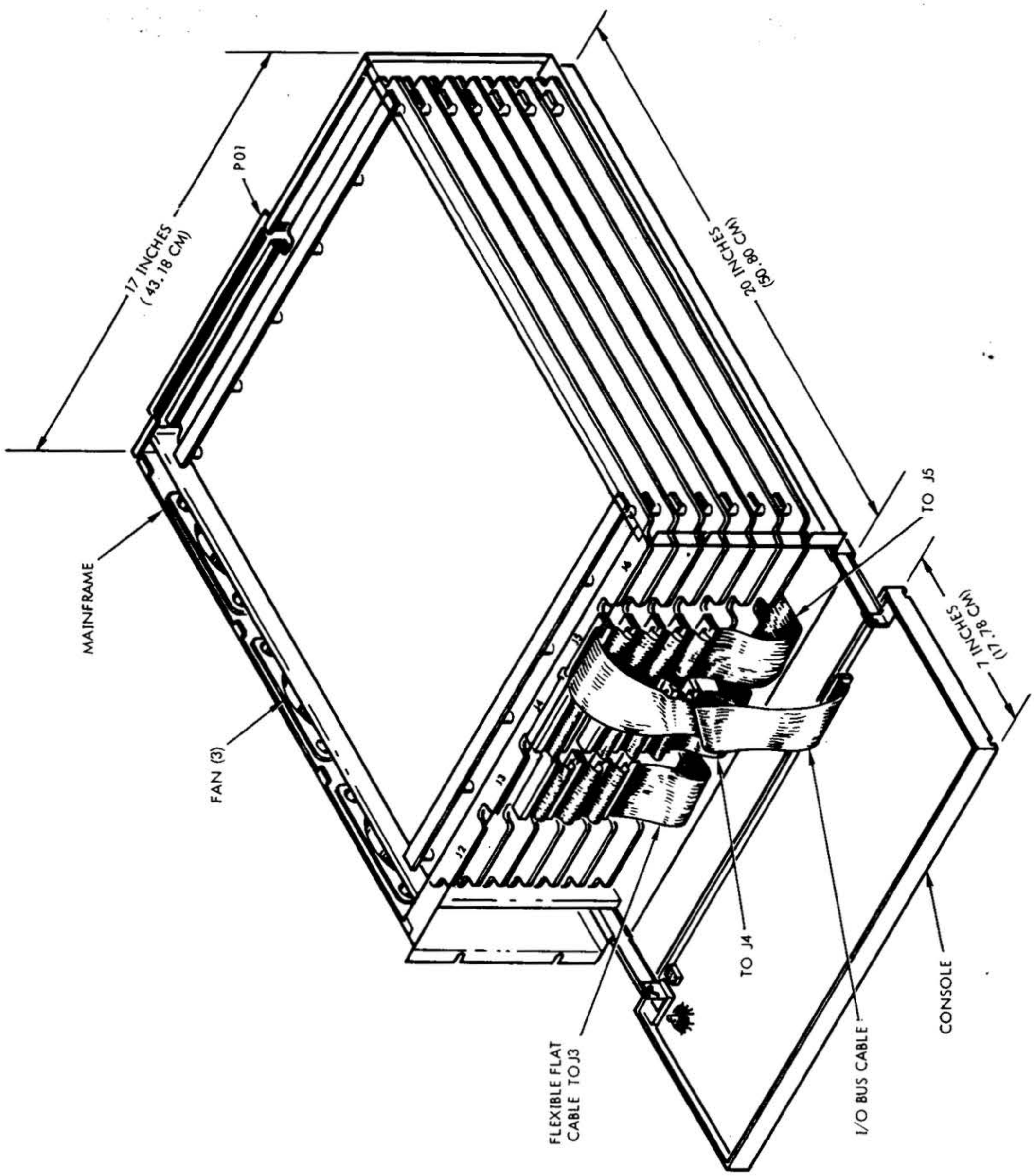
The memory expansion chassis is a standard 7-inch (7 card slots) or optional 14-inch (17 card slots) mainframe chassis. The memory printed circuit PC bus in the rear of each chassis is connected via flat cable to extend the memory buses between chassis.

Varian offers a complete complement of peripheral devices and their controllers to simplify total system planning and installation. Available peripherals include: high-speed paper-tape equipment, magnetic tape units, disc and drum memories, punched-card equipment, and a variety of analog, digital, and communications equipment.

Standard software for the Varian 73 includes: the DAS symbolic assemblers, FORTRAN IV, BASIC, Master Operating System (MOS), the business-oriented RPG IV, AID II for debugging, MAINTAIN II for troubleshooting, plus a complete library of mathematical and utility subroutines also available is the Varian Omnitask Real-time Executive (VORTEX or VORTEX II), which is a modular operating system for controlling, scheduling, and monitoring tasks in a real-time multiprogramming environment.

The Varian field-service organization provides a comprehensive customer service program to assist the user in system planning, installation, and maintenance for his own unique application. Service contracts cover necessary scheduled preventive maintenance and emergency repairs. VOICE, Varian's user organization, acts as a clearing house for user inquiries and suggestions and provides information about new products and system applications in both hardware and software.

The Varian Data Machines (VDM) department of customer education offers regu-



VTI1-1488

Figure 1-2. Mainframe Chassis (Typical Layout)

larly scheduled training courses covering the complete spectrum of Varian's growing computer family. Both programming and maintenance courses are offered as well as a complete course on Varian's dynamic new software VORTEX system. All classes are a combination of lecture/application with special emphasis given to hands-on training. For further details, the customer education department can be contacted directly or through any local VDM office.

Physical Characteristics

This section explains and illustrates the standard optional 7-inch (17.78 cm) or optional 14-inch (35.56 cm) mainframe chassis, I/O expansion chassis and memory expansion chassis, along with power distribution and control. The chassis fits into a RETMA standard 19-inch (48.26 cm) wide and 24-inch (60.96 cm) deep cabinet. Detailed explanations of physical dimensions, circuit locations, bus structures, connections, and general assembly information is covered in the following subsections.

Mainframe Chassis

The standard mainframe chassis (figure 1-2) or optional mainframe chassis (not shown) accommodates the control panel, processor, core memory, option board, fans, and the capability for I/O and memory expansion.

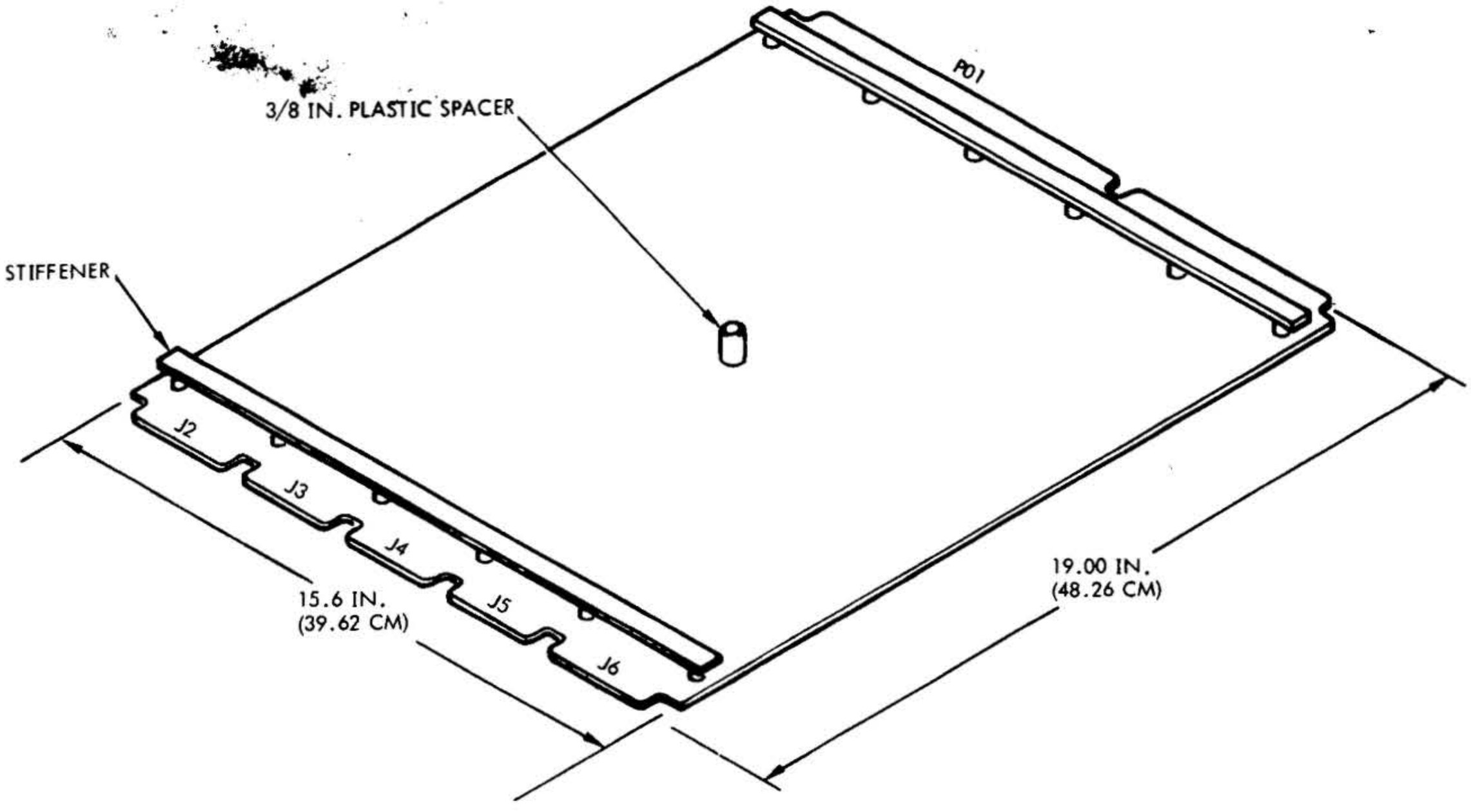
The standard 7-inch (17.78 cm) high, 19-inch (48.26 cm) wide, and 1.38-inch (3.51 cm) thick molded-plastic control panel contains all of the controls and indicators necessary to operate the computer (section 10) and has a printed circuit (PC) card mounted to the back side which holds light emitting diodes and switches. The control panel, hinged at the bottom to the mainframe chassis, folds down for easy access to the circuit boards.

The 14-inch (35.56 cm) high, 19-inch (48.26 cm) wide and 1.38-inch (3.51 cm) thick front panell contains a control panel in the bottom half and a blank panel in the top half. The control panel contains all of the controls and indicators necessary to operate the computer. The front panel is molded plastic with a printed circuit (PC) card mounted behind the control panel to hold the light emitting diodes and switches. The front panel, hinged at the bottom of the control panel to the mainframe chassis, folds down, parallel to the floor, for easy access to the circuit boards.

The control panel makes connection to the various circuit boards in the mainframe via a 50-wire flat I/O cable, from a 50-pin connector on the rear of the control panel. A power cable also connects to the rear of the control panel to activate indicators and switches.

The 15.6-inch (39.62 cm) wide, 19-inch (48.26 cm) deep, 0.062-inch (.157 cm) thick circuit boards (figure 1-3) slide into the mainframe from the front and plug into the power and memory buses in the rear. The PC boards are spaced on 0.6-inch (1.524 cm) centers with the bott slot made for a board with components that would require two slots. Also, there are device connectors for the I/O controllers, PMA (J7A, J7B), TTY (J8), and RTC (J9) mounted at the rear of the option board.

Because of the flexible design of the mainframe, any multilayer circuit board (processor/option are four layers; core/SC memory are three layers) can be mounted in any of the seven universal slots (17 slots in the 14-inch mainframe) numbered from bottom to top. Any combination of circuit



VTII-3226

Figure 1-3. Typical Circuit Board

boards are tied together via a 50-wire flat cable at the front. The front of each circuit board has up to five 50-pin board edge connectors and are numbered from left to right (facing front) from J2 through J6 in vertical rows. The I/O flat cable buses connect the appropriate circuit boards and can then loop in along the bottom several inches to connect with the I/O port.

The I/O port consists of a subassembly of card guides, connectors, and PC board interconnections. If an expansion chassis is used, an I/O extender board plugs into the I/O port. Otherwise an I/O bus termination shoe is plugged into the I/O port.

The power supply control and dc power signals connect to the rear, right side of either the 7-in or 14-inch chassis (facing from the rear). The ac power for the fans enters at the lower right rear with the three, 4.5-inch square, muffin fans (six in the 14-inch chassis) mounted vertically on the right side of the mainframe (facing rear). Memory and dc voltage distribution is via a three-layer PC bus connector board that provides access to all boards in either the 7- or 14-inch mainframes.

The 14-inch chassis has the same physical configuration as the 7-inch chassis except that it is designed to contain 17 universal

slots. The console mounts on the bottom front half with a blank panel covering the top half.

I/O Chassis

The I/O chassis is 10.5-inches (26.60 cm) high, 19-inches (48.3 cm) wide. The chassis can accommodate up to 22 etched-circuit controller cards, up to eight wire-wrap controller cards, or a combination of both. One card slot contains an I/O cable connector (figure 1-4), one card slot contains the party line I/O expander and cable, one card slot contains a terminator, one card slot contains the other end of the I/O expander cable, and one card slot contains either another I/O expansion cable for daisy chaining to the next I/O expansion chassis or a termination shoe for the I/O bus signals.

The connector panel, located at the bottom rear of the chassis, contains power connector J31 and mainframe power supply connector J30. The circuit card slots are numbered at the rear of the chassis from right to left. Each half of the backplane contains 12 card slots numbered 2 through 13.

Each peripheral controller card is 7.75-by-12-inches (19.7 by 30.3 cm) and contains a 122-pin connector for mating with a backplane connector in the I/O expansion chassis. The other end of the card has two 44-pin connectors that mate with peripheral device cables.

The mainframe connects to the I/O expansion chassis with a flat cable from a paddleboard connection at the I/O port. The flat cable extends between the paddleboard, in the I/O port and the paddleboard in a connector location at the left of the I/O chassis (facing from the rear). If another expansion chassis is added, paddleboard connectors and flat cable tie them together. The termination shoe mounts in an end card slot in the I/O expansion chassis.

Memory Expansion Chassis

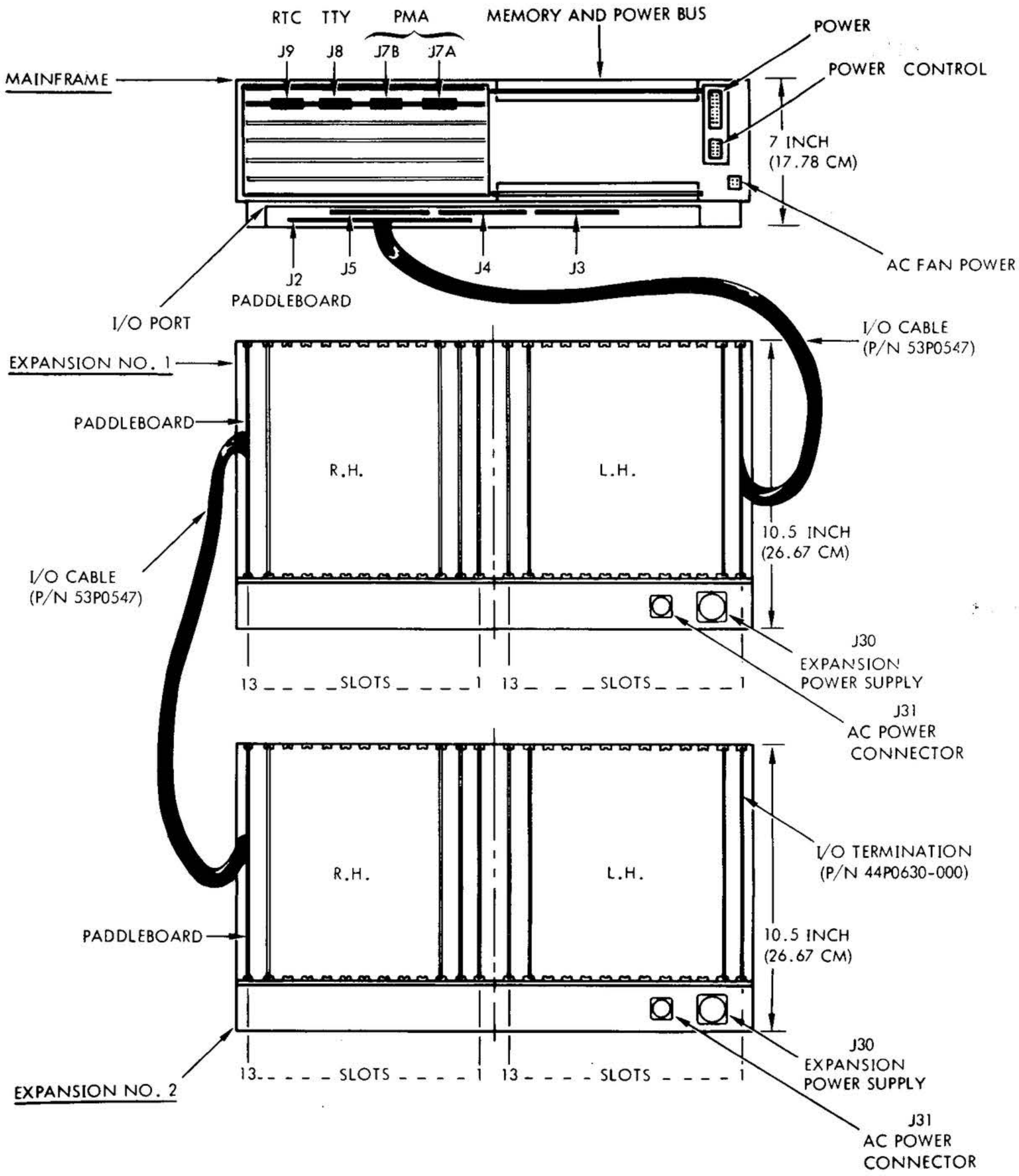
The memory expansion chassis is a 7-inch (17.78 cm) high frame that can hold four PC boards.

The rear of the memory expansion chassis connects with the rear of the mainframe chassis via the flat cable from the bottom of the memory bus in the memory chassis to the top of the memory bus in the mainframe (figure 1-5).

Power enters at a 16-pin connector on the right rear (facing rear) with power control beneath it on an 8-pin connector. Ac fan power is brought in on a small connector at the bottom rear corner.

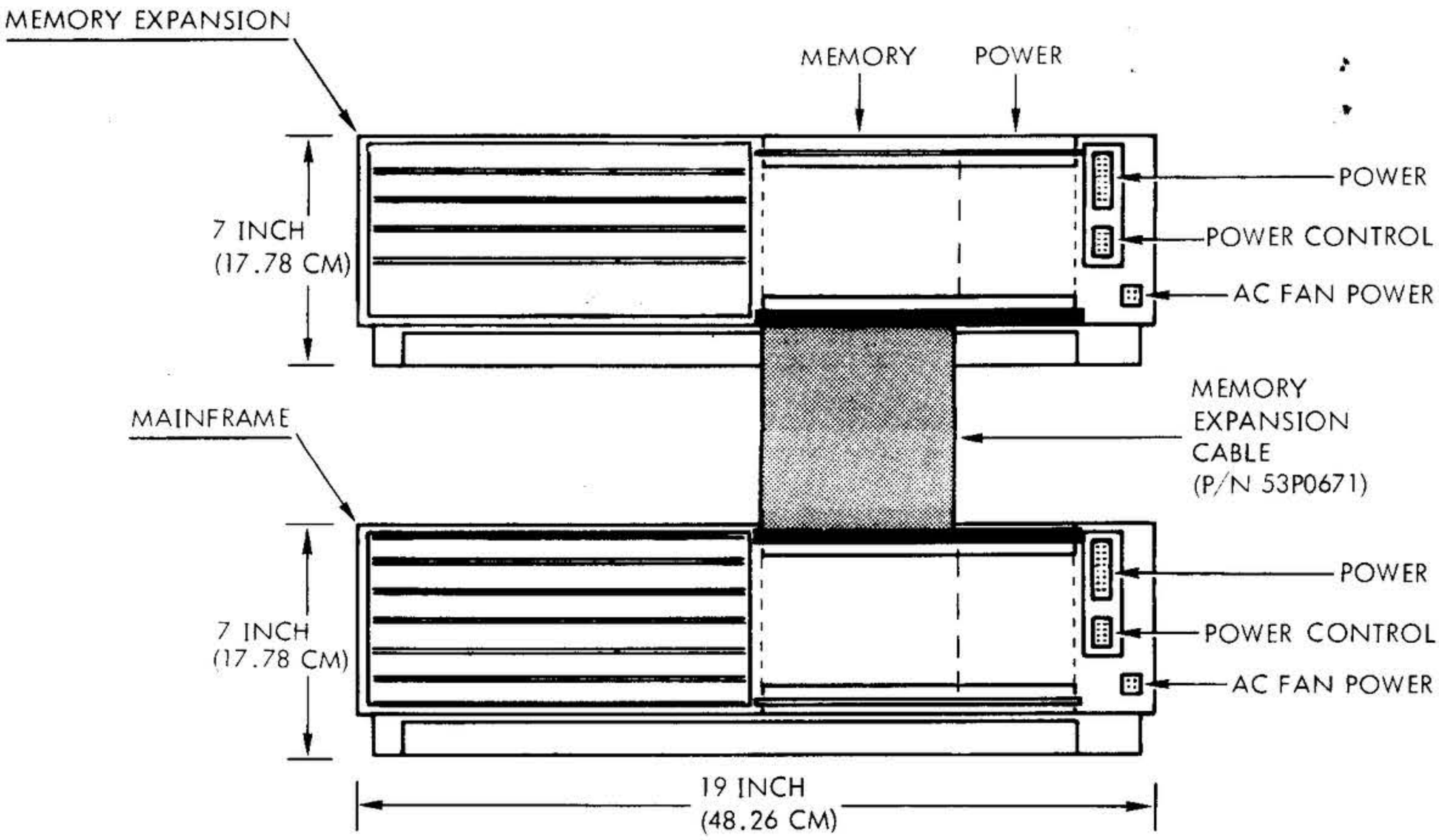
Power Distribution and Control

All power to mainframe chassis (exclusive of WCS, memory map, and floating point processor) is generated by the power supply contained in a 5.22-inch (13.26 cm) high, 19-inch (48.26 cm) wide, 19-inch (48.26 cm) deep, standard retma rack chassis (figure 1-6). The outputs are linearly regulated, protected for current overload, and monitored to maintain outputs within the prescribed limits. In addition, the +5V is protected for overvoltage.



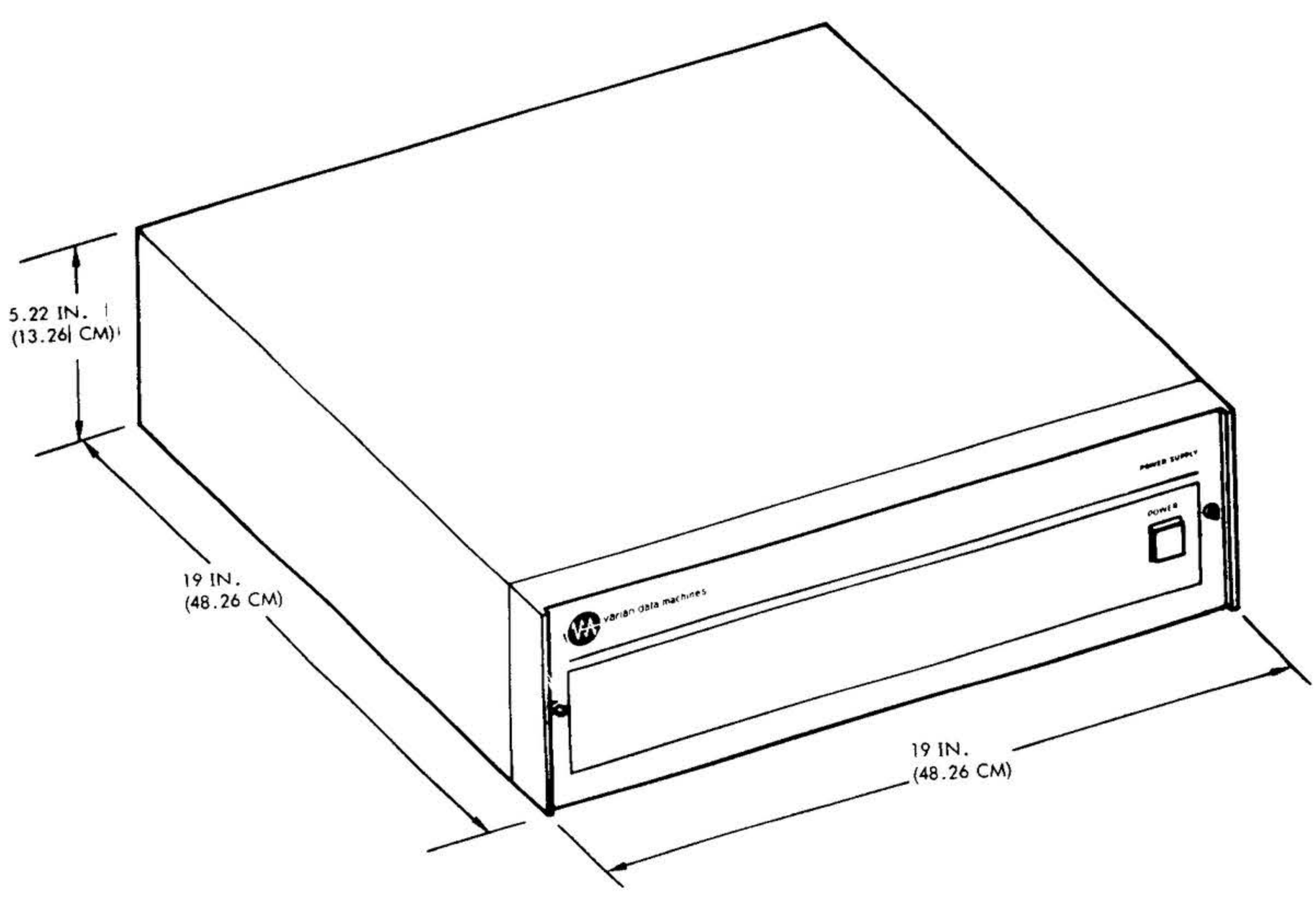
VTII-1489 A

Figure 1-4. I/O Expansion-Rear View



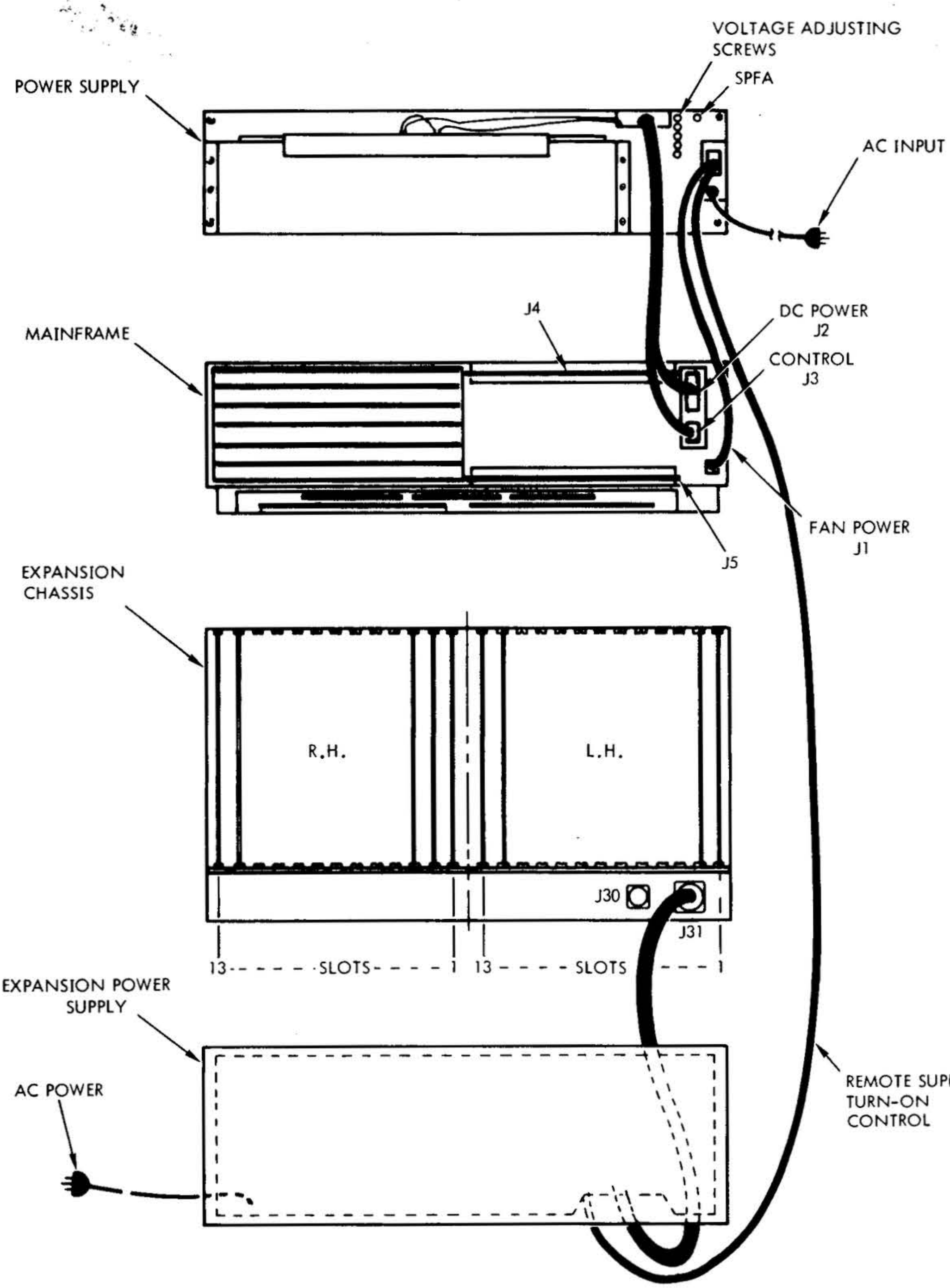
VTI1-1490 A

Figure 1-5. Memory Expansion (Rear View)



VTI1-3227

Figure 1-6. Power Supply



VTI1-1491

Figure 1-7. Power Distribution/Control (Rear View)

The power supply is turned on, remotely, from the power switch on the mainframe console. With the switch on, the lines, designated 24V ac to RELAY and 24V ac, are connected together to apply 24V across the relay and energize the power supply (figure 1-7).

The power supply provides the following features:

a. A power failure alarm and time delayed system reset.

b. A 900 nanosecond energy storage following the power failure alarm.

c. Power shutdown if excessive temperatures are attained within the supply.

d. Power turn on/turn off sequence.

e. 24V ac sine wave signal for RTC control.

f. Power for the ac fans.

Varian 73 Specifications

Type	General purpose microprogrammed digital computer.
Memory, Semiconductor	Dual port semiconductor memory with 16-bit word length. Available in 8,192 word modules. Optional 18 bit word for byte parity
Memory, Core	Dual port magnetic core memory with 16-bit word length. Available in 8,192 modules. Optional 18 bit word for byte parity
Memory Size	Expandable to 32,768 words in any combination of semiconductor and core modules. Expandable to 65,536 words with Writable Control Store option. Expandable to 262,144 words with memory map option.
Word Length	Sixteen bits.
Registers	Sixteen total; three general-purpose registers and thirteen microprogramming registers.
Arithmetic	Binary, two's complement.
Cycle Times	
Semiconductor Memory	330 nanoseconds
Core Memory	660 nanoseconds 990 nanoseconds 1200 nanoseconds

Instruction Execution Times	
Semiconductor Memory	Register-register: 330 nanoseconds Memory-register: 660 nanoseconds Jump: 701 nanoseconds
Core Memory	Register-register: 660 nanoseconds Memory-register: 1200 nanoseconds Jump: 1361 nanoseconds
I/O Transfer Rates	
Semiconductor Memory	DMA: 969,600 words per second DMA (620 compatible): 372,900 words per second PMA*: 1,102,000 words per second (writing) 1,010,000 words per second (reading)
Core Memory (660 ns)	DMA: 897,800 words per second DMA (620 compatible): 361,800 words per second PMA*: 1,010,000 words per second (writing) 932,000 words per second (reading)
Instructions:	160 standard; may be extended with Writable Control Store. Floating Point Processor option adds 14 additional instructions
Instruction Types	Single-word, addressing Single-word, nonaddressing Double-word, addressing Double-word, nonaddressing
Addressing Modes	Direct to 2,048 words Relative to P, X or B register to 512 words Preindexing with X or B register Multilevel indirect to 32,768 words Indirect indexed Immediate Post indexing with X or B register Extended mode to 32,768 words Direct to 65,536 words with Writable Control store option, to 262,144 words with Memory Map option.

* Assuming burst-mode operation, a 55-nanosecond controller delay, and a 10-foot cable.

Logic Levels	Positive logic: (Internal) True: +2.4V minimum, +5V maximum False: −0.5V minimum, +0.5V maximum Negative logic: (I/O Bus) True: − 0.5V minimum, +0.4 maximum False: − +2.8V minimum, +3.6V maximum
Standard Features	Teletype controller Hardware multiply/divide Power failure/restart Real time clock Hardware priority interrupt Direct Memory Access
Computer Options	Automatic bootstrap loader (paper tape, rotating memory) Priority memory access (PMA) Writable control store (WCS) Memory map Memory parity Floating point processor (FPP) Core memory interleaving 16K core memory (1.2 microsecond cycle time)
I/O Options	Buffered interlace controller (VIC) Buffered I/O controller (BIOC) Priority interrupt module (PIM) Block transfer controller (BTC)
Dimensions	Mainframe and memory expansion chassis are 14-(35.56 cm) and 7-inches (17.78 cm) high (respectively), 19-inches (48.26 cm) wide, and 20.5-inches (52.10 cm) deep. I/O expansion chassis is 10-1/2-inches (26.60 cm) high, 19-inches (48.26 cm) wide, 19-inches (48.26 cm) deep
Input Voltage	105 to 125V ac or 210 to 250V ac, at 50 or 60 Hz
Temperature	
Operating	0 to 50 degrees C
Storage	-20 to 70 degrees C

Humidity	
Operating	To 90 percent without condensation
Storage	To 95 percent without condensation

Software

Symbolic Assembler	Modular two-pass assemblers includes over 30 pseudo-operations; MR macro version operates under the Master Operating System (MOS) and VORTEX
BLD II	Binary load/dump program that allows the loading of object programs and the punching of the binary contents of memory for reloading
Subroutines	Complete library of basic mathematical and utility subroutines; provides for the addition of special-purpose routines
FORTRAN	Modular one-pass compiler; subset of ANSI FORTRAN for 8K of memory
BASIC	An easy-to-use programming language for business and scientific applications that permits an inexperienced operator to program the system with only a few hours training
MOS	A master operating system that provides for automatic batch-processing in as little as 8K of memory
RPG IV	An optional report program generator system for business applications; produces reports, financial statements, sales records, and other commercial documents
AID II	Program analysis package that assists programmers in operating the computer and debugging other programs; includes basic operational executive subroutines
EDIT	Program modification package that allows on-line correction of symbolic source programs
MAINTAIN III	Software package that provides efficient off-line verification of CPU and peripheral operation and assists in isolating and correcting suspected faults

VORTEX	A multiprogramming operating system that provides the versatility offered by large, expensive computer systems; VORTEX permits concurrent real-time programming and background processing
VORTEX II	Same as VORTEX but utilizes memory map hardware for more efficient relocation and allocation of memory resources
VTAM	Executive that operates under VORTEX to handle data communications
RPG II	An industry-compatible business compiler
TSS	Time Sharing Subsystem. Provides multi-terminal access to BASIC, EDIT, and batch background. Operates under VORTEX.

Varian 73 Model Numbers

Model	Description	Prerequisites
	Central Processors	
73-1100	73 Central Processor Unit with power failure/restart, multiply/divide, Teletype controller, automatic boot-strap for Teletype, real-time clock, memory protection, I/O bus with DMA, 8,192 words of dual-port core memory, 14 P slots, power supply, and control panel.	
73-1101	73 Central Processor Unit with power failure/restart, multiply/divide, Teletype controller, automatic boot-strap for Teletype, real-time clock, memory protection, I/O bus with DMA, 8,192 words of dual-port core memory, 14 P slots, power supply, priority memory access (PMA), and control panel.	
73-1200	73 Central Processor Unit with power failure/restart, multiply/divide, Teletype controller, automatic boot-strap for Teletype, real-time clock, memory protection, memory parity, I/O bus with DMA, 8,192 words of dual-port parity core memory, 14 P slots, power supply, and control panel.	
73-1201	73 Central Processor Unit with power failure/restart, multiply/divide, Teletype controller, automatic boot-strap for Teletype, real-time clock, memory protection, memory parity, I/O bus with DMA, 8,192 words of dual-port parity core memory, 14 P slots, power supply, priority memory access (PMA), and control panel.	

Model	Description	Prerequisites
	Core Memory	
73-2100	8,192-word (16 bits) Dual-port Core Memory 660 nanoseconds cycle time.	72-11XX
73-2101	8,192-word (18 bits) Dual-port Parity Core Memory, 660 nanoseconds cycle time.	72-12XX
73-2102	16,384-word (16 bits) Single-port Core Memory with 1200 nanoseconds cycle time.	73-CPU
73-2103	16,384-word (18 bits) Single-port parity Core Memory with 1200 nanoseconds cycle time.	73-CPU
73-2400	32,768-word (16 bits) Core Memory, includes memory expansion chassis "slave," power supply, cables, and four 8K x 16-bit core memories, 660 nanosecond cycle time.	73-110X 73-3300
73-2401	32,768-word (16 bits) Core Memory, includes memory expansion chassis "master," power supply, cables, and four 8K x 16-bit core memories, 660 nanosecond cycle time.	73-2400
73-2410	32,768-word (18 bits) Parity Core Memory, includes memory expansion chassis "slave," power supply, cables, and four 8K x 18-bit core memories, 660 nanosecond cycle time.	73-120X 73-3300
73-2411	32,768-word (18 bits) Parity Core Memory, includes memory expansion chassis "master," power supply, cables, and four 8K x 18-bit core memories, 660 nanosecond cycle time.	73-2410

Model	Description	Prerequisites
Semiconductor Memory		
73-2500	8,192-word (16 bits) Dual-port Semiconductor Memory with 330 nanosecond cycle time.	73-110X or 73-150X
73-2501	8,192-word (18 bits) Dual-port parity Semiconductor Memory with 330 nanosecond cycle time.	73-120X or 73-160X
	OPTIONS	
73-3001	Automatic Bootstrap Loader (ABL) for paper tape reader; instead of "standard" for Teletype.	70-6300 or 70-6301
73-3002	Automatic Bootstrap Loader (ABL) for rotating memory instead of "standard" for Teletype.	70-76X0
73-3003	Automatic Bootstrap Loader (ABL) for rotating memory instead of "standard" for Teletype.	70-7500
73-3004	Automatic Bootstrap Loader (ABL) for rotating memory instead of "standard" for Teletype.	70-7510
73-3010	Real-Time Clock (RTC) special configurations. Customers of non-VORTEX systems may specify source inputs of 10KHz, line frequency, external input, or alternate counter overflow for both the free-running counter and the variable-internal interrupt counter.	73-CPU
73-3030	Odd/Even Interleaving for dual-port, 660 nanosecond core memory, up to 256K.	73-9101
73-3031	Odd/Even Interleaving for single-port, 1200 nanosecond core memory in expansion chassis, up to 256K.	73-9101

Model	Description	Prerequisites
73-3060	230V ac, 50Hz System Power Input.	73-CPU
73-3075	V75 instruction set option has 27 instructions that permit programmer access to 8 general-purpose registers and operate on 8-, 16-, and 32-bit operands.	
73-3100	Block Transfer Controller (BTC) for automatic data transfers between peripheral and memory via the PMA channel. Maximum of four BTC modules per V73 processor with PMA.	73-CPU and 73-9004
73-3101	Priority Interrupt Module (PIM) for automatic storing and vectoring of eight levels of externally generated interrupts. Maximum is eight modules (64 levels) per V73 processor.	73-CPU and 73-9004
73-3102	Buffer Interlace Controller (BIC) provides block transfer supervisor for DMA transfers from up to 10 devices, a maximum of eight BIC's per V73 processor.	73-CPU and 73-9004
73-3200	Data Save Power Supply and Battery for semiconductor memory	73-15XX or 73-16XX
73-3300	Memory Map provides automatic allocation and control of up to 256K words of main memory, includes power cable	73-CPU and 32K memory
73-3400	Floating Point Processor performs single precision and double precision floating point arithmetic operations. Direct parallel connection to CPU and to memory. Provides high speed acquisition, processing and storage rates.	73-CPU
73-4000	256-word (64 bits) Writable Control Store (WCS), 190ns cycle time, 16-register stack.	73-CPU 73-102 73-4090

Model	Description	Prerequisites
73-4001	512-word (64 bits) Writable Control Store (WCS), 190ns cycle time, 16-register stack.	73-CPU 73-3102 73-4090
73-4002	512-word (64 bits) Writable Control Store (WCS), 190ns cycle time, 16-register stack, instruction register writable decoder control store, writable I/O control store. NOTE: A maximum of three WCS modules, or two WCS modules and one Floating Point Processor (FPP) module can be added to a V73 processor.	73-CPU 73-3102 73-4090
73-4090	WCS Power Supply with cable.	

EXPANSION CHASSIS

73-9004	First I/O Chassis with 19 I/O slots, power supply, cables and I/O party line expander for 10 bus loads. Note: Limit one per system, for further expansion use 72-9005.	73-CPU
73-9005	I/O Expansion Chassis with 22 I/O slots, power supply, and cables.	73-9004 or 70-591X
73-9006	I/O Party Line Expander for 10 unit load in I/O expansion chassis.	73-9005
73-9101	Memory Expansion Chassis "Slave" with cables and power supply for 32K dual-port core, 32K semiconductor memory, or 64K single-port core memory.	73-3300
73-9102	Memory Expansion Chassis "Master" with cables and power supply for 32K dual-port core, 32K semiconductor memory, or 64K single-port core memory	73-9101

Model	Description	Prerequisites
	CABINETS	
70-9200	Equipment Cabinet, 19-inch standard, rack mounting, 77-inches high, 30-inches deep, includes cooling unit, power distribution and breaker for 115V ac, 60 Hz, casters, and installation of standard equipment.	73-CPU
70-9201	Same as 70-9200 except 230V ac, 50 Hz.	73-CPU with 230V ac, 50 Hz power
70-9202	Equipment Cabinet, 19-inch standard, rack mounting. 30-inches deep, 60-inches high, includes cooling unit, power distribution and breaker for 115V ac, 60 Hz, casters and installation of standard assemblies	73-CPU
70-9203	Same as 70-9200 except 230V ac, 50 Hz.	73-CPU with 230V ac, 50 Hz power
	SPARE PARTS	
73-9910	Component Spares Kit for V73 processor, option board, and core memory.	
73-9920	Processor Module.	
73-9921	Option Module.	
73-9922	Option Module with PMA.	
73-9930	Programmers Console.	
70-9940	V73 Processor or Core/Semiconductor Memory-Expansion Power Supply	
70-9941	I/O Power Supply (17A).	
70-9942	I/O Power Supply (35A).	

Model	Description	Prerequisites
	ACCESSORIES	
70-9951	Extender Board for I/O Modules.	
70-9952	I/O Cable with connectors (5 feet).	
70-9953	Test Cable Set for V73 processor	
70-9960	Multi-Use I/O Socket Board.	
	Communications	
	DATA COMMUNICATION MULTIPLEXORS (DCM)	
70-5201	Data Communication Multiplexor (DCM) including message oriented control for up to 16 high-performance communication channels, supporting either synchronous, asynchronous, or direct-connection terminal control. Provides six asynchronous communication line speeds up to 9600 baud, DMA I/O control, multiplexor bus control for connected line adapters.	70-5910
70-5202	Data Communication Multiplexor (DCM) for up to 32 communication channels.	70-5910 or 70-5911
70-5203	Data Communication Multiplexor (DCM) for up to 64 communication channels.	70-5912
70-5211	Data Communication Multiplexor (DCM) including message oriented control for up to 16 high-performance communication channels, supporting either synchronous, asynchronous, or direct-connection terminal control. Provides six asynchronous communication line speeds up to 9600 baud, DMA I/O control, multiplexor bus control for connected line adapters. To be used only with systems containing memory map.	70-5910 or 70-5911

Model	Description	Prerequisites
70-5212	Data Communication Multiplexor (DCM) for up to 32 communication channels. To be used only with systems containing memory map.	70-5910 or 70-5911
70-5213	Data Communication Multiplexor (DCM) for up to 64 communication channels. To be used only with systems containing memory map.	70-5912
70-5702	Binary Synchronous Communication Multiplexor (BSCM) for message oriented control of up to eight (8) BSC communication channels. Includes DMA I/O control and multiplexor bus control for BSC line adapters (70-5306).	70-5910 or 70-5911
70-5712	Binary Synchronous Communication Multiplexor (BSCM) for message oriented control of up to eight (8) BSC communication channels. Includes DMA I/O control and multiplexor bus control for BSC line adapters (70-5306). To be used only with systems containing memory map.	70-5910 or 70-5911
	DCM LINE ADAPTERS	
70-5301	Asynchronous Line Adapter with RS232C or CCITT V24 compatibility for four channels of full or half duplex asynchronous operation up to 9600 baud. Supports 103 or 202 series modems or equivalent. Auto-detection parity and control character. Programmable selection of line speeds (2 of 6 speeds/line) and control characters. Includes VDM mating connectors	70-52XX
70-5302	Direct-Connection RS232/DT-TTL Line Adapter. Direct-connection (non-modem) devices within 50' of the processor, both low and high speed,	70-52XX

Model	Description	Prerequisites
	four full or half duplex lines with RS232 interface. Includes VDM mating connectors.	
70-5303	Direct-Connection Current Loop Line Adapter. Direct-connection (non-modem) devices located with current loop distance (50' to 5000') from processor dependent upon selected baud rates, four full or half duplex channels, 20/60 mA solid state current loop interface. Includes VDM mating connectors.	70-52XX
70-5304	Direct-Connection Relay Line Adapter. Direct-connection (non-modem) devices utilizing 20/60 mA signaling and an electro-mechanical type (relay) interface. Maximum rate is 300 bps. Telegraph network capability. Four full or half duplex channels. Includes VDM mating connectors	70-52XX
70-5305	Synchronous Line Adapter with RS232C, or CCITT V24 compatibility for four full or half duplex channels; line speed up to 20K baud. Programmable selection and auto-detection of control characters and sync characters. Supports 201 and 208 series modems or equivalent. Includes VDM mating connector	70-52XX
70-5306	Binary Synchronous Communication (BSC) Line Adapter for one communication channel. Supports full or half duplex operation with RS232C compatible interface for Bell 201, or 208 modems or equivalent. Bell 300 series modem interface option (70-5801) available for wide-band rates up to 50,000 baud. Includes programmable selection of EBCDIC, ASCII and transparent modes with integrated LRC and CRC error con-	70-5702 or 70-5712

Model	Description	Prerequisites
	trol and auto answer facilities. VDM mating connector.	
70-5307	Automatic Call Unit Line Adapter. Provides facilities for program control of four Bell 801 type (A/C) data auxiliary sets (or equivalent) for dialing any telephone number in the switched telephone network. Includes VDM mating connector.	70-52XX
70-5308	Programmable Asynchronous Line Adapter with RS232C or CCITT V24 compatibility for four channels of full or half duplex asynchronous operation up to 9600 baud. Supports Bell 103 or 202 series modem or equivalent. Provides auto answer, error reporting and programmable selection of; six line speeds, parity mode, character level (5, 6, 7, 8) and start/stop bits (1 or 2). Includes VDM mating connector.	
	ASYNCHRONOUS MODEM CONTROLLERS*	
70-5401	Data Set Controller - Bell 103 or 202 series modems or equivalent full, or half duplex asynchronous operation; speeds up to 9600 baud, auto-detection parity, overrun, framing; RS232C/CCITT V24 compatible, auto answer, both lines must operate at same speed. Includes VDM mating connector.	
70-5402	Dual Data Set Controller - Bell 103 or 202 series modems or equivalents; full or half duplex asynchronous operation, speeds up to 9600 baud, auto-detection parity, overrun, framing; RS232C/CCITT V24 compatible, auto answer, both lines must operate at same speed. Includes VDM mating connector	

Model	Description	Prerequisites
	SYNCHRONOUS MODEM CONTROLLERS*	
70-5501	Data Set Controller - Bell 201 type or equivalent full/half duplex synchronous operation. Speeds up to 2400 baud, single character buffered, software sync recognition, code transparency, automatic answer. Includes a 20-ft. data set cable.	
70-5502	Dual 70-5501 Data Set Controller. Includes two 20-ft. data set cables.	
70-5503	Data Set Controller - Bell 201 type or equivalent, full/half duplex synchronous operation. Speeds up to 50,000 baud, double character buffering, hardware line synchronization (sync characters may be changed under hardware control), and code transparency. The 70-5503 may be operated in conjunction with the DMA channel and buffer interlace controller (BIC). Includes a 20-ft. data set cable.	
70-5504	Data Set Controller - 70-5503 with extended control for optimized full duplex operation.	70-5911 or 7X-9004 or 7X-9005
70-5505	Binary Synchronous Communication facilities for one (1) communication channel. Includes message oriented control for Bell 201 or 208 series modem or equivalent; programmable mode selection of EBCDIC, ASCII and transparent text communications with integrated LRC and CRC error control, half or full duplex operation, and auto answer. VDM mating connector included. Bell 300 Series modem interface option (70-5801) available for wide-band rates up to 50,000 baud.	70-5913 or 70-5914

Model	Description	Prerequisites
70-5506	Binary Synchronous Communication facilities for two (2) communication channels.	70-5913 or 70-5914
70-5515	Binary Synchronous Communication facilities for one (1) communication channel. For use with systems containing memory map.	70-5913 or 70-5914
70-5516	Binary Synchronous Communication facilities for two (2) communication channels. For use with systems containing memory map.	70-5913 or 70-5914

*All controllers require ± 12V dc.

UNIVERSAL ASYNCHRONOUS SERIAL CONTROLLERS

70-5601	Universal Asynchronous Serial Controller with RS232C interface.	
70-5602	Universal Asynchronous Serial Controller with 20 or 60 mA current loop interface.	
70-5603	Universal Asynchronous Serial Controller with 20 mA relay interface.	

AUTOMATIC CALL UNIT CONTROLLER*

70-5701	Automatic Call Unit Controller (ACU). The ACU controller provides program control of the 801 (A/C) data auxiliary set and permits dialing any telephone number in the switched telephone network. Includes a 20-ft. data set cable.	

OPTIONS

70-5801	Binary Synchronous Communications wide-band interface option for Bell 300 series modems. Provides for line speeds up to 50,000 baud.	70-55X5, 70-55X6, or 70-5306

Model	Description	Prerequisites
*All controllers require a ± 12V dc.		
	COMMUNICATION CHASSIS	
70-5910	Communication I/O Chassis for 70-5201 5211, 70-5202, 5212 DCM or 70-5702, 5712 BSCM. Includes one backplane for up to eight (8) 70-530X line adapters, I/O cables, and power supply (17A).	73-CPU
70-5911	Communication I/O Chassis for 70-5201, 5211, 70-5202, 5212 DCM or 70-5702, 5712 BSCM. Includes one backplane for up to eight (8) 70-530X line adapters, one backplane with 12 I/O slots, I/O cables, and power supply (35A).	73-CPU
70-5912	Communication I/O Chassis for 70-5203, 5213 DCM. Includes two backplanes for up to sixteen (16) 70-530X line adapters, I/O cables, and power supply (35A).	73-CPU
70-5913	Communication I/O Chassis for 70-55X5 or 70-55X6 Binary Synchronous Communication controllers plus seven (7) additional I/O slots. Includes chassis, special backplane, I/O cables, and power supply (17A).	73-CPU
70-5914	Communication I/O Chassis for 70-55X5 or 70-55X6 Binary Synchronous Communication controllers plus nineteen (19) additional I/O slots. Includes chassis, 1 special and 1 I/O backplane, I/O cables, and power supply (35A).	73-CPU
	COMMUNICATION ACCESSORIES	
70-5901-XX	Modem Cable (single) for connection of EIA standard modems (Bell equivalent) to 70-5401 or 70-5402 data set controllers. One cable required for	70-5401 or 70-5402

Model	Description	Prerequisites
	each modem. Cable of optional length complete with mating connectors, with optional length in ft. specified by XX suffix of model number. Nominal cable length of 20 ft. supplied when no suffix specified.	
70-5902-XX	Modem Cable (dual) for connection of EIA standard modems (Bell equivalent) 70-5301 and 70-5305 line adapters. Each cable connects two modems to line adapter, complete with mating connectors. Cable of optional length with length in ft. specified by XX suffix of model number. Nominal cable length of 20 ft. supplied when no suffix specified.	70-5301 or 70-5305 or 70-5308
70-5903-XX	Modem Cable (single) for connection of EIA RS232 standard modem (Bell equivalent) to BSC interfaces on models 70-55X5, 55X6 and 70-5306. One cable required for each modem. Optional cable lengths available with length in feet specified by XX suffix of model number. Nominal cable length of 20 feet supplied when no suffix specified.	70-55X5 or 70-55X6 or 70-5306
70-5904-XX	Modem Cable (single) for connection of Bell 300 Series modems (or equivalent) to BSC interfaces using 70-5801 option. One cable required for each modem. Optional cable lengths available with length in feet specified by XX suffix of model number. Nominal cable length of 20 feet supplied when no suffix specified.	70-5801
70-5905-XX	Bell 801 ACU Line Adapter (70-5307) cable. Each cable provides interconnection for two (2) Bell 801 A/C units or their equivalent. Optional	70-5307

Model	Description	Prerequisites
	Cable lengths available with length in ft. specified by XX suffix or model number. Nominal cable length of 20 ft. supplied when no suffix specified.	
	Peripherals	
	TELETYPES	
70-6100	ASR-33	73-CPU
70-6102	KSR-35	73-CPU
70-6104	ASR-35	73-CPU
	CARD EQUIPMENT	
70-6200	Card Reader and Controller, 80 columns, 300 cards per minute.	
70-6201	Card Punch and Controller, 35 cards (80 columns each) per minute. (E-3174) (refurbished).	
	PAPER TAPE	
70-6300	Paper Tape Reader and Controller, 300 characters per second	
70-6310	Paper Tape Punch and Controller, 75 characters per second, table top.	
70-6311	Paper Tape Punch and Controller, 75 characters per second, 19-inch panel mounted.	
70-6320	Paper Tape System, includes time-share controller, 300 characters per second reader, 75 characters per second punch.	
	DISPLAYS AND TERMINALS	
70-6400	Oscilloscope Display (Tektronix Model 611), 11-inch storage scope.	70-8200

Model	Description	Prerequisites
70-6401	Keyboard and Alphanumeric CRT Display, conversational mode (Teletype replacement).	7X-CPU
70-6402	Model 70-6401 with 70-5602 controller.	70-5911 or 7X-9004 or 7X-9005
70-6403	Model 70-6401 with kit and instructions to connect to controllers, or a spare unit.	70-5301 or 03 70-5601 or 02

PRINTER/PLOTTERS
STATOS ®31 FAMILY

70-6606	8-1/2 inch wide Printer/Plotter with Controller, 80 styli per inch, 2.75 inches per second, 1320 alphanumeric lines per inch.	
70-6640	Model 70-6606 with 64 character 5 x 7 dot matrix hardware character generator and simultaneous print/plot options.	
70-6608	11 inch wide Printer/Plotter with Controller, 100 styli per inch, 2.2 inches per second, 1000 alphanumeric lines per inch.	
70-6641	Model 70-6608 with 123 character, upper and lower case, 7 x 11 dot matrix character generator and simultaneous print/plot options.	
70-6602	14-7/8 inch wide Printer/Plotter with Controller, 100 styli per inch, 2.2 inches per second, 1000 alphanumeric lines per inch.	
70-6642	Model 70-6602 with 123 character, upper and lower case, 7 x 11 dot matrix character generator and simultaneous print/plot options.	

Model	Description	Prerequisites
	STATOS 33 FAMILY:	
70-6611	**Bi-Scan Writing Head Models** 8-1/2 inch wide Printer/Plotter with Controller, 100 styli per inch, 1 inch per second, 460 alphanumeric lines per inch.	7X-3101 and 7X-3102 (VORTEX only)
70-6613	11 inch wide Printer/Plotter with Controller, 100 styli per inch, 0.9 inches per second, 410 alphanumeric lines per minute.	
70-6615	14-7/8 inch wide Printer/Plotter with Controller, 100 styli per inch, 0.8 inches per second, 370 alphanumeric lines per minute.	
70-6617	22 inch wide Printer/Plotter with Controller, 100 styli per inch, 0.6 inches per second, 210 alphanumeric lines per minute.	
	Linear Writing Head Models	
70-6621	8-1/2 inch wide Printer/Plotter with Controller, 100 styli per inch, 1.5 inches per second, 690 alphanumeric lines per minute.	
70-6623	11 inch wide Printer/Plotter with Controller, 100 styli per inch, 1.5 inches per second, 890 alphanumeric lines per minute.	
70-6625	14-7/8 inch wide Printer/Plotter with Controller, 100 styli per inch, 1.5 inches per second, 690 alphanumeric lines per minute.	
70-6627	22 inch wide Printer/Plotter with Controller, 100 styli per inch, 1.2 inches per second, 550 alphanumeric lines per minute.	

Model	Description	Prerequisites
	PRINTERS	
70-6701	Line Printer with controller, 245 to 1100 lpm, 132 columns, 64 characters segmented buffer, 460 lpm for first 72 columns.	
70-6720	Line Printer and controller, 300 lpm 136 columns, 64 characters, 11 position form length selector switch.	
70-6721	Line Printer and controller, 300 lpm, 136 columns, 64 characters, 12 channel paper tape vertical format unit.	
70-6722	Line Printer and Controller, 600 lmp, 136 columns, 64 characters, 11 position form length selector switch.	
70-6723	Line Printer and Controller, 600 lpm, 136 columns, 64 characters, 12 channel paper tape vertical format unit.	
70-6760	Static eliminator option	70-6720 or 70-6721
	MAGNETIC TAPE	
70-7100	Magnetic Tape Unit and Controller, 9-track, 800 bpi, 25 ips, read/write single density, includes control for up to four 9-track magnetic tape units.	
70-7101	Magnetic Tape Unit Slave, 9-track 800 bpi, 25 ips, read/write single density.	70-7100
70-7102	Magnetic Tape Unit and Controller, 9-track, 800 bpi, 37-1/2 ips, read after write, single density, includes control for up to four magnetic tape units.	
70-7103	Magnetic Tape Unit Slave, 9-track, 800 bpi, 37-1/2 ips, read after write, single density.	70-7102

Model	Description	Prerequisites
	ROTATING MEMORY	
70-7500	Disc Memory and Controller (2316 pack) moving head, single spindle capacity of 11.7 million 16-bit words, transfer rate 156K words per second, track to track 10.0 ms. Controller controls master and up to three slave units.	73-3100
70-7501	Disc Memory Slave Unit (2316 pack) moving head, single spindle capacity up to 11.7 million 16-bit words, transfer rate 156K words per second.	70-7500
70-7510	Disc Memory and Controller (2316 pack) moving head, dual spindle, (four spindles per controller), total capacity up to 46.7 million 16-bit words, transfer rates 156K words per second, track to track 10 ms.	73-3100
70-7511	Disc Memory Slave Unit (2316 pack) moving head, dual spindle, total capacity up to 46.7 million 16-bit words.	70-7510
70-7600	Disc Memory and Controller, moving head, total capacity 2.34 million 16-bit words, one fixed and one removable disc (5440 pack) transfer rate 92K words per second, track to track 10ms. Controller controls master and one slave.	73-3102
70-7601	Disc Memory Slave Unit, moving head, total capacity 2.34 million 16-bit words, one fixed and one removable disc (5440 pack) transfer rate 92K words per second.	70-7600
70-7610	Disc Memory and Controller, moving head (2315 pack) capacity 1.17 million 16-bit words, transfer rate 92K words	73-3102

Model	Description	Prerequisites
	per second, track to track 15 ms. Controller controls master and up to three slave drives.	
70-7611	Disc Memory Slave Unit, moving head (2315 pack) capacity 1.17 million 16-bit words, transfer rate 92K words per second.	70-7610
70-7700	61K-word Fixed Head Disc and Controller, 105KHz transfer rate, average access time 17 ms.	73-3102
70-7701	123K-word Fixed Head Disc and Controller, 105kHz transfer rate, average access time 17ms.	73-3102
70-7702	246K-word Fixed Head Disc and Controller, 105kHz transfer rate, average access time 17ms.	73-3102
70-7703	491K-word Fixed Head Disc and Controller, 105 kHz transfer rate, average access time 17ms.	73-3102
	ANALOG-TO-DIGITAL CONTROLLERS	
	High Level Analog-To-Digital Converter Modules: ± 10V full scale input, sample and hold, programmable time. Processor interface to I/O or BIC.	
70-8000	13 bits, single channel, 50kHz maximum.	70-8200
70-8001	10 bits, single channel, 100kHz maximum.	70-8200
	High Level Analog Input Systems: Analog-to-digital converter, sample and hold, programmable timer, multiplexor and control for 16 single ended or differential channels expandable to 256. Differential and single ended channels may be intermixed as well as	

Model	Description	Prerequisites
	input voltages. Sequential or non-sequential addressing. Processor interface to I/O or BIC. Multiplexor bus will also accept Digital Input Module 70-8410.	
70-8010	± 10V 13 bits, 50kHz maximum, 16 single ended channels.	70-8200
70-8011	± 10V, 13 bits, 50kHz maximum, 16 differential channels.	70-8200
70-8012	± 1V, 13 bits, 30kHz maximum, 16 differential channels.	70-8200
70-8013	± 10V, 10 bits, 100kHz maximum, 16 single ended channels.	70-8200
70-8014	± 10V, 10 bits, 100kHz maximum, 16 differential channels.	70-8200
70-8015	± 1V, 10 bits 50kHz maximum, 16 differential channels.	70-8200
	High Level Multiplexors: Multiplexor control module for 16 channels expandable to 256 channels by use of the expansion modules. Differential and single ended channels may be intermixed as well as input voltages. Sequential or non-sequential addressing. Processor interface to I/O or BIC. Multiplexor bus will also accept Digital Input Module 70-8410.	
70-8020	Basic Module with multiplexor control, ± 10V, 16 single ended channels,	70-8000 or 70-8001
70-8021	Expansion Module, ± 10V, 16 single ended channels.	70-8010,11, 13,14
70-8022	Basic Module with multiplexor control, ± 10V, 16 differential channels,	70-8000 or 70-8001

Model	Description	Prerequisites
70-8023	Expansion Module, ± 10V, 16 differential channels.	70-8010,11, 13,14
70-8024	Basic Module with multiplexor control, ± 1V, 16 differential channels.	70-8000 or 70-8001
70-8025	Expansion Module, ± 1V, 16 differential channels.	70-8010,11, 13,14
70-8090	I/O Expansion Chassis for high level analog systems. Does not include power supply.	73-CPU
	Low Level Analog Input System: 13 bits, analog-to-digital converter, multi-channel multiplexor with eight levels of full scale voltage inputs ranging from ± 9.77 mv to ± 1.25V, programmable gain amplifier with eight computer selectable gains, sample and hold amplifier, programmable timer, shielded chassis, power supplies and interconnection cables. Multiplexor bus will accept Digital Input Module 70-8410.	73-CPU
70-8101	16 channels.	
70-8102	32 channels.	
70-8103	48 channels.	
70-8104	64 channels.	
70-8105	80 channels.	
70-8106	96 channels.	
70-8107	112 channels.	
70-8108	128 channels.	
70-8109	144 channels.	

VARIAN 73 SYSTEM COMPUTER

Model	Description	Prerequisites
70-8110	160 channels.	
70-8111	176 channels.	
70-8112	192 channels.	
70-8113	208 channels.	
70-8114	224 channels.	
70-8115	240 channels.	
70-8116	256 channels.	

LOW LEVEL MULTIPLEXORS

70-8120	Low Level Expansion System with multiplexor control: 16 low level channels with eight levels of full scale input voltages ranging from ± 9.77 mv to ± 1.25V, programmable gain amplifier with eight computer selectables gains, shielded chassis, power supply.	70-8000,01, 10 through 15
70-8121	Low Level Expansion System without multiplexor control: 16 low level channels with eight levels of full scale voltage inputs ranging from ± 9.77 mv to ± 1.25V programmable gain amplifier with eight computer selectable gains, shielded chassis, power supply, and inter-connection cables.	70-8010 through 15 or 70-8101 through 16
70-8122	Low Level Expansion Module: 16 low level channels with eight levels of full scale voltage inputs ranging from ± 9.77 mv to ± 1.25V, programmable gain amplifier with eight computer selectable gains.	70-8101 through 15 or 70-8120,21

Model	Description	Prerequisites
	ANALOG POWER SUPPLY	
70-8200	Input: 115/230V ac ± 10 percent, 47 Hz to 63 Hz, 1.6A at full load. Output: +5V dc at 5A ± 15V dc at 1A, ± 20V dc at 250 mA, and +24V dc at 500 mA. Supply mounts on front or rear cabinet rails.	73-CPU
	DIGITAL-TO-ANALOG CONTROLLERS (DAC)	
	Expandable to 64 channels with expansion modules. Processor interface to I/O or BIC. Bus will also accept Digital Output Module 70-8310. DAC Module with controls, ±10V, full scale output at 5 mA maximum:	
70-8210	One 10-bit channel.	70-8200
70-8211	Two 10-bit channels.	70-8200
70-8212	DAC Expansion Module, two 10-bit channels. DAC Module with control, ± 10 volts, full scale output at ± 10 mA maximum:	70-8210 or 70-8211, 70-8213, and 70-8214
70-8213	One 12-bit channel.	70-8200
70-8214	Two 12-bit channels.	70-8200
70-8215	DAC Expansion Module, two 12-bit channels. DAC Module with control, ± 10V, full scale output at ± 10 mA maximum:	70-8213 or 70-8210, 70-8211, and 70-8214
70-8216	One 14-bit channel.	70-8200
70-8217	Two 14-bit channels.	70-8200

Model	Description	Prerequisites
70-8218	DAC Expansion Module, two 14-bit channels.	70-8216 or 70-8217
70-8219	DAC Module with control, one 10-bit channel at ± 10V full scale output at ± 5 mA maximum and one 12-bit channel at ± 10V full scale output at ± 10 mA maximum.	70-8200
70-8220	DAC Module with control, one 10-bit channel at ± 10V full scale output at ± 5 mA maximum and one 14-bit channel at ± 10V full scale output at ± 10 mA maximum.	70-8200
70-8221	DAC Module with control, one 12-bit and one 14-bit channel both channels have ± 10V full scale output at ± 10 mA maximum.	70-8200
	DIGITAL CONTROLLERS	
70-8301	Buffered I/O Controller, (printed-circuit version), general purpose interface, eight sense lines, eight control pulses, 16-bit output register, 16-bit input register. One of eight different pulse widths available (5-20, 20-73, 73-300 usec; 27-1.1, 1-4, 3.5-13, 12-50, 40-90 ms).	73-CPU
70-8310	Digital Output Module, two 16-bit output registers, one general purpose buffered input, eight control lines, eight sense lines and I/O-BIC interface.	73-CPU
70-8311	Digital Output Module Expansion, two 16-bit output register, one general purpose buffered input.	70-8310

Model	Description	Prerequisites
70-8410	Digit Input Module, four 16-bit input registers, control for sequential or random register addressing, expandable up to 256 input registers and I/O-BIC interface.	73-CPU
70-8411	Digital Input Module Expansion, four 16-bit input registers.	70-8410
70-8500	Relay Contact I/O Module, 16 contact points, voltage levels, 12 VA resistive, 0.5A of 200V maximum.	73-CPU
70-8501	Relay Contact I/O Module, 16 mercury wetted contact outputs, 50 VA resistive, 3A or 400V maximum.	73-CPU
70-8502	Relay Contact I/O Module, 16 mercury wetted contact outputs and 16 contact inputs, 50 VA resistive, 3A or 400V maximum.	73-CPU

Most of the equipment listed is also available with 50 Hz power.

SECTION 2 - PROCESSOR

Microprogramming Elements

In the conventional processor, the control section normally consists of large assemblies of gates and flip-flops interconnected to form timing counters, sequencers, and decoders to perform the following functions required by the specific instruction set:

- fetch instructions from memory
- decode machine instructions
- enable appropriate data paths
- change the state of the computer to that required by the next operation

In a microprogrammed processor, the control section is implemented in a less random fashion. All control signals are derived from information stored in a memory device (usually a read only memory). This memory, together with its buffers and control logic, form the control sections. The control words stored in the memory are known as microinstructions. Preparation of these instructions is known as microprogramming. These microinstructions bear no resemblance to the computer's own instruction set as they manipulate and control data at the most elementary level.

For a comparison between a microprogrammed and a conventional computer refer to Figure 2-1.

Advantages of microprogramming are.

- Provides an orderly method of implementing modifications and extensions to existing instruction sets.
- Permits easier troubleshooting through minimization of random logic.
- Permits optimum tailoring of computer systems to a specific task by implementing frequently used operations in microinstructions.

A more detailed explanation of microprogramming is given in the Varian Microprogramming Guide (document number 98A 9906 07x).

Functional Description

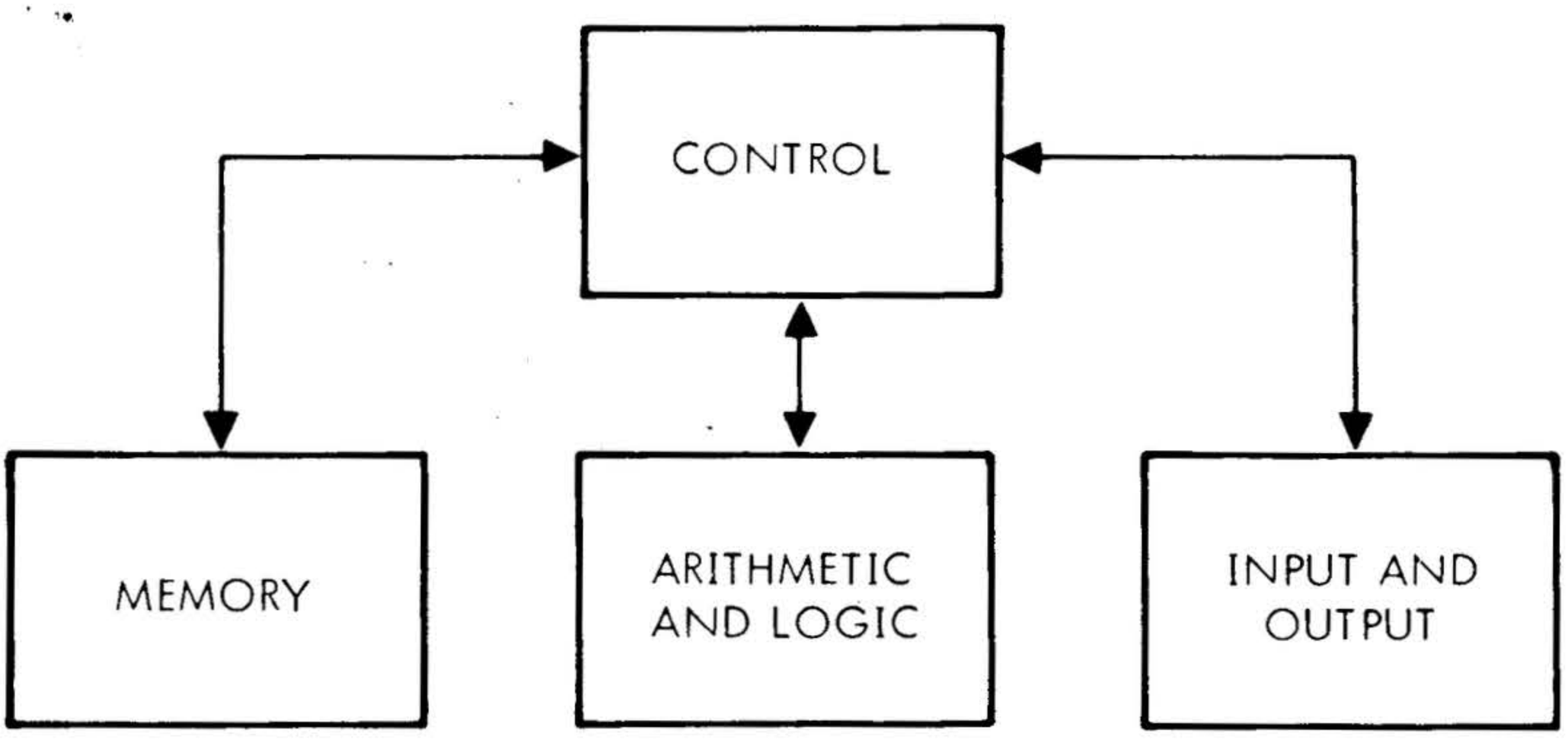
As illustrated in figure 2-2, the major functional sections of the Varian 73 processor are central control, data loop, memory control, I/O data loop, and I/O control. Except for the I/O control, which is located on the option board, these sections are on the processor board. The processor communicates with the control panel via the I/O bus.

Central Control

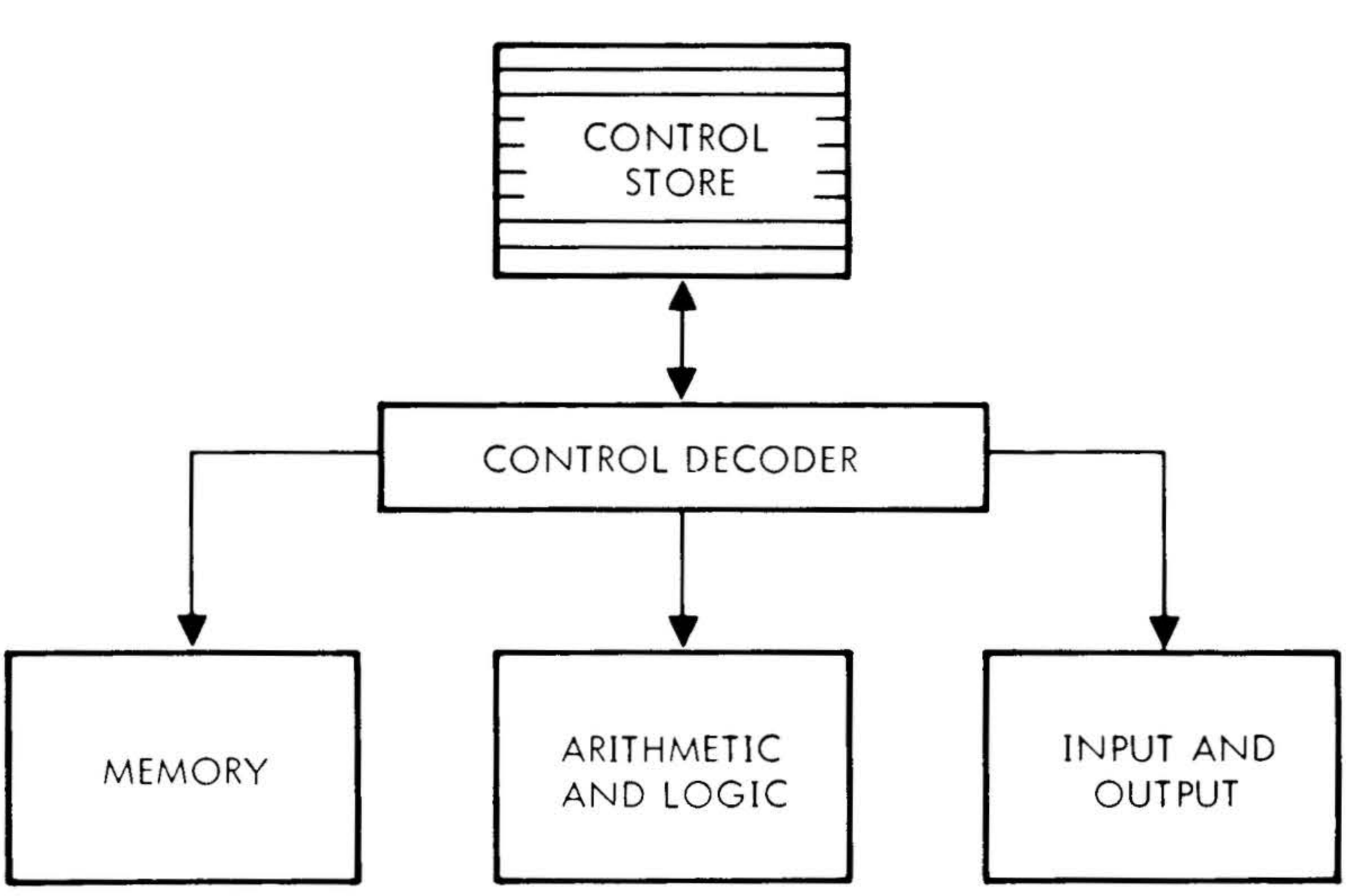
The central control is the heart of the processor. It contains the instruction register, control store, control store buffer, and control sequencing logic. The following

PROCESSOR

CONVENTIONAL CONTROL



SIMPLIFIED GENERAL MICROPROGRAMMING



VTI1-3240

Figure 2-1. Microprogrammed Versus Conventional Computer Simplified Block Diagram

functions are performed by the central control:

a. Initiates memory operations

b. Initiates I/O operations

c. Decodes instructions

d. Controls data transfers and manipulations

e. Tests internal data loop conditions

f. Responds to interrupts

The 16-bit instruction register receives instructions from an instruction buffer which is then free to accept new instructions. This double buffering of instructions provides a pipelining technique that allows the next instruction to be fetched during an otherwise unused memory cycle. The output of the instruction register can then be routed to the arithmetic and logic unit (ALU) or further decoding may be performed.

Data Loop

The data loop provides data transfer paths, data manipulation circuits, storage registers, and counters (figure 2-3). The data loop performs the following functions:

a. Selects both of the ALU inputs from the following sources:

1. 16 general purpose registers

2. Operand register

3. Memory input register, in memory control section

4. I/O data register, in I/O data loop section

5. Status word (signals displayed by control-panel STATUS switch)

6. Instruction register (masked), in central control section

7. Program counter

8. Control store literal which consists of a 16-bit mask field from the control store buffer.

b. Performs arithmetical and logical operations on the ALU inputs.

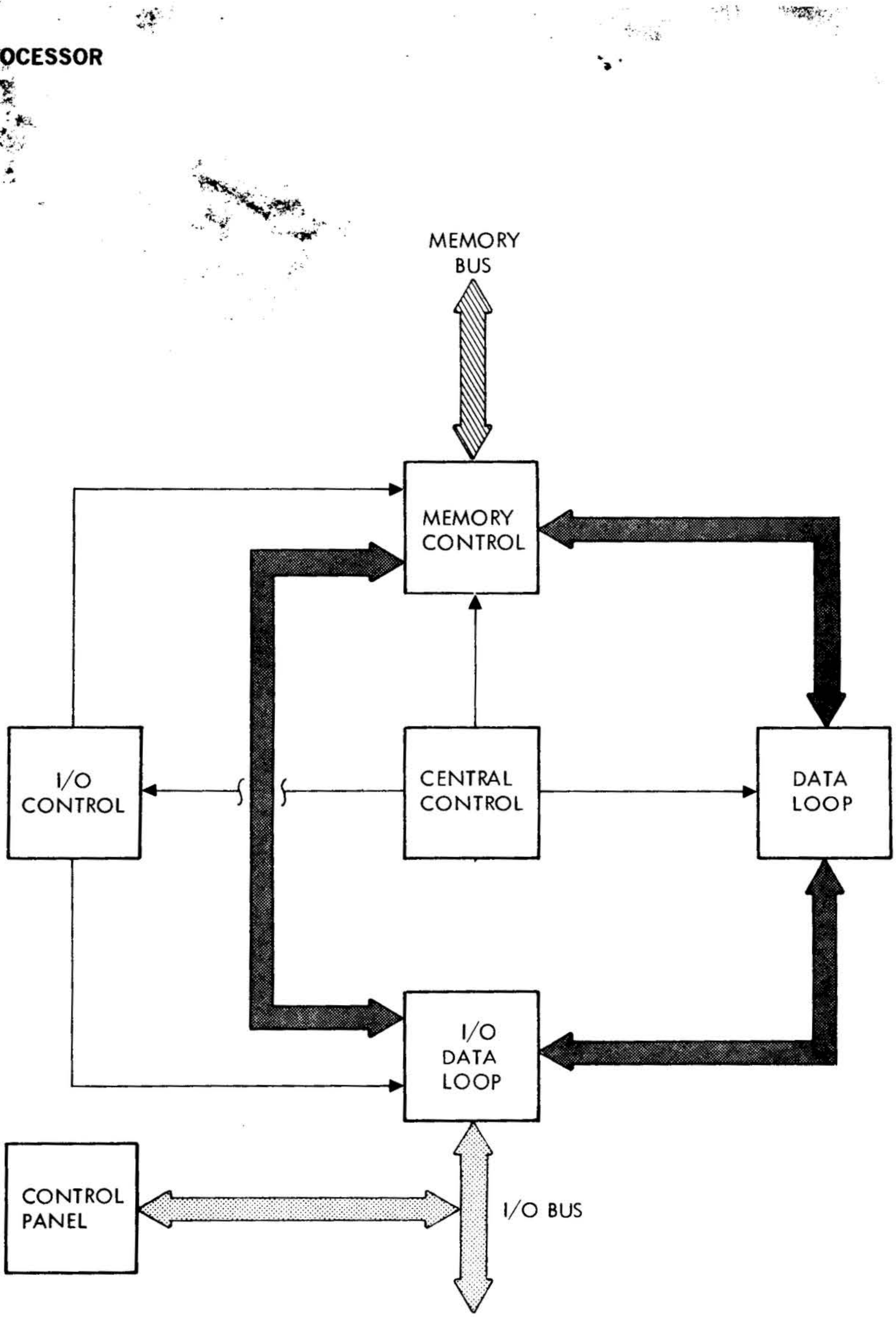
c. Performs single and double length, bidirectional, open or closed, arithmetical or logical shifts in accordance with the contents of the shift counter.

d. Stores and selects the desired test conditions such as ALU output zero, overflow, carry, SENSE switches, etc.

The ALU performs arithmetic and logical functions under control of the control store buffer. The ALU output is applied to the memory control and I/O data loop sections.

Memory Control

The memory control initiates memory operations requested by the central control, I/O control, or options (on option board). It acknowledges acceptance of each request and signals completion of the requested memory operation to the requesting section. Once a request is accepted, no further requests are acknowledged (one exception to this rule permits the central control to override a previous



VTII-3228

Figure 2-2. Varian 73 Processor Block Diagram

request before it has been completed). Priority memory access (PMA) requests have a higher priority than I/O requests, and I/O requests have a higher priority than central control requests.

The following functions are performed by the memory control:

a. Accepts memory requests from central control and stores the following information to complete the memory operation:

   1. Read/write
   2. Word/byte
   3. Address source

b. Accepts memory requests from the PMA option.

c. Accepts memory requests from I/O control (for DMA transfers or programmed data transfers).

d. Acknowledges receipt of memory requests.

e. Resolves priority of simultaneous requests deferring lower priority requests until memory operations for higher priority requests have been completed.

f. Provides asynchronous operation and drivers/receivers for the memory bus.

g. Signals completion of scheduled operation.

Since the memory control operates asynchronously, the central control is free to perform one other non-memory operation while the scheduled operation is being executed. The memory control's ability to accept memory requests from the I/O control permits direct memory access (DMA) operations to cycle steal without interfering with non-memory operations in the remainder of the processor.

I/O Data Loop

The I/O data loop (see figure 2-3 for I/O data paths) contains a multiplexor, I/O data register, and drivers and receivers. Three sources of data are applied to the I/O data loop: data from the I/O bus, data from the ALU and data from the memory I/O latch. The input data is selected by the I/O multiplexer under control of I/O control signals and transferred onto the bidirectional I/O bus.

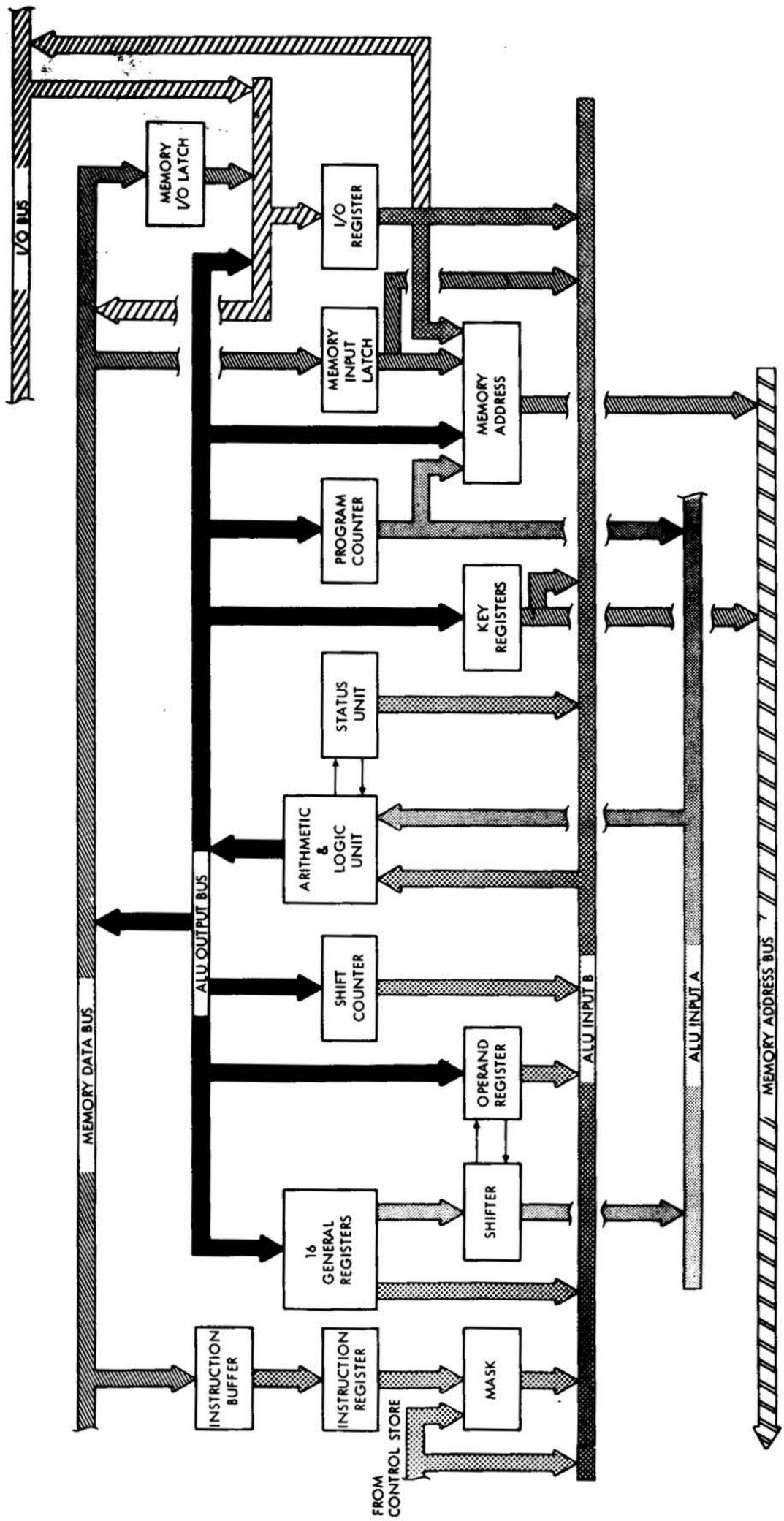
In addition to being applied to the I/O drivers, the output of the I/O data register is applied to the data loop and memory control sections.

I/O Control

The I/O control operates under control of an independent I/O read only memory (random access memory when controlled by the WCS) and performs I/O operations initiated either by the central control or I/O device activity. This permits I/O operations to proceed with minimum impact on other internal processor functions. The I/O control performs the following functions:

a. Programmed I/O initiated by the central control.

b. DMA trap-in/trap-out operations (up to 969,600 words per second with semiconductor memory).



VTI2-453

Figure 2-3. Varian 73 Processor Data Paths

c. DMA (620 compatible) trap-in/trap-out operations (up to 372,900 words per second with semiconductor memory).

d. I/O interrupts.

Standard Features

The following standard features are included with the Varian 73 processor:

- Hardware multiply/divide
- Power failure/restart
- Teletype controller
- Real-time clock
- Automatic bootstrap loader (TTY)
- Memory protection
- Hardware priority interrupt
- Direct memory access (DMA)

The hardware multiply/divide is on the processor board; the power failure/restart, memory protection, Teletype controller, and real-time clock are on the option board.

Hardware priority interrupt and DMA are discussed in section 6.

Hardware Multiply/Divide

The hardware multiply/divide feature enhances the computational capabilities of the processor by reducing the number of steps required for multiplication and division operations.

During multiplication, the contents of the effective memory address are multiplied by the contents of the B register, then the A register contents are added to the product. The result is placed in the A and B registers, with the most significant half in the A register and the least significant half in the B register. The sign of the result is in bit 15 (sign bit) of the A register. The B register sign bit is always set to zero. The largest positive multiplier or multiplicand in an operation register in the processor is thus 15 binary bits.

During division, the dividend is contained in the combined A and B registers with the sign in bit 15 of the A register. The B register sign bit is not used. The divisor is in the effective memory address. The quotient and its sign are placed in the B register and the remainder (with the sign of the dividend), in the A register.

Power Failure/Restart

The power failure/restart (PF/R) protects, during loss or reduction of ac line voltage, the program in progress and the contents of computer memory and registers. Upon restoration of power, the PF/R automatically restarts the computer and causes it to reenter the interrupted program at the point of interruption.

Power reduction, failure, or turn-off initiates a power-down cycle during which the PF/R sustains execution of the current instruction and then interrupts the processor, directing it to the address of the user prepared SAVE subroutine. This SAVE subroutine loads the contents of the volatile registers (A, B, X, P, and overflow) into preselected addresses in memory. After the execution of SAVE, the PF/R disables the processor and memory until power is restored.

PROCESSOR

When power is restored so that all power-up conditions are satisfied, the PF/R enables the processor and memory, initiates the system-start signal, and directs the processor to the address of the RESTORE subroutine. This service subroutine reloads the registers with the saved data, and contains a jump instruction that directs the processor to reenter the program at the point of interruption and continue execution.

Teletype Controller

The Teletype controller (TC) is a serial, asynchronous, full-duplex, data-transfer interface between the processor and a Model 33 or 35 Teletype, using the I/O bus. The TC also provides for the buffering of data, the sensing of status by the processor, an interrupt capability for priority interrupt module (PIM) control, and the

Table 2-1. Teletype Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0401	100401	Initialize Teletype controller
	Data Transfer	
OAR 01	103101	Transfer A register contents to input register
OBR 01	103201	Transfer B register contents to input register
OME 01	103001	Transfer memory contents to input register
INA 01	102101	Transfer output register contents to A register
INB 01	102201	Transfer output register contents to B register
IME 01	102001	Transfer output register contents to memory
CIA 01	102501	Transfer output register contents to cleared A register
CIB 01	102601	Transfer output register contents to cleared B register
	Program Sense	
SEN 0101	101101	Ready to write
SEN 0201	101201	Ready to read

Table 2-1. Teletype Controller Instructions *(continued)*

Mnemonic	Octal Code	Description

Teletype Commands

Function	Symbol	Octal Code	Type As:
Enable printer	SOM	201	CONTROL and A
Suppress printer	EOT	204	CONTROL and D
Reader on	XON	221	CONTROL and Q
Punch on	TAPE	222	CONTROL and R
Reader off	XOFF	223	CONTROL and S
Punch off	TAPE OFF	224	CONTROL and T

initialization of the system through program control or by use of the RESET switch on the control panel. Table 2-1 lists the instructions for the TC.

Real-Time Clock

The real-time clock (RTC) provides the following real-time functions:

- Variable-interval interrupt
- Memory-overflow interrupt
- Readable free-running counter

The variable-interval interrupt has three preselectable hardware timing sources: (1) a 10KHz signal (standard unless otherwise specified), (2) line frequency from the power supply, or (3) a user-supplied external source. The rate of the variable-interval interrupt is selectable under program control as follows. The program loads a number n between 1 and 4095 into the variable-interval logic. The logic repeatedly counts down from n to 0, issuing an interrupt each time 0 is reached. The variable-interval interrupt rate is thus 1/nth of the timing source rate.

The memory-overflow interrupt, which operates in conjunction with the variable-interval interrupt, is implemented by loading an increment-memory-and-replace instruction into the address of the variable-interval interrupt. This is monitored by the overflow-detection logic, which triggers the memory-overflow interrupt when the contents of the variable-interval interrupt are incremented to 040001.

The 16-bit readable free-running counter is continually updated and may be read under program control. Counter timing is based on the 10KHz clock, the variable-interval interrupt rate, the line frequency, or a user-supplied external source. Table 2-2 lists the instructions for the RTC.

Automatic Bootstrap Loader (TTY)

The automatic bootstrap loader (Teletype) automatically loads the bootstrap program into computer memory from an integrated-circuit read only memory (control store in the processor). The program then executes and reads the loader program into memory from the Teletype paper tape reader.

Table 2-2. RTC Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 047	100047	Clear FRC (the FRC cannot otherwise be reset)
EXC 0147	100147	Enable RTC interrupts (pending RTC interrupts are immediately processed)
EXC 0247	100247	Inhibit memory overflow interrupts (inhibits only overflow interrupts)
EXC 0347	100347	Enable variable interval interrupts (inhibits overflow interrupts)
EXC 0447	100447	Initialize RTC (inhibits all interrupts and resets the interrupt control logic and clock control logic)
EXC 0647	100647	Initialize variable-interval counter (loads the contents of the interval selection register in the interval counter and resets the clock control)
EXC 0747	100747	Inhibit variable-interval interrupt
	Data Transfer	
OAR 047	103147	Output A register to interval selection register
OBR 047	103247	Output B register to interval selection register
OME 047	103047	Output memory to interval selection register
INA 047	102147	Input FRC contents to A register
INB 047	102247	Input FRC contents to B register
IME 047	102047	Input FRC contents to memory
CIA 047	102547	Clear and input FRC contents to A register
CIB 047	102647	Clear and input FRC contents to B register

Memory Protection

Memory protection (MP) is contained on the Varian 73 option board. MP prevents unauthorized or unintentional program access to, or modification of, protected areas of memory.

Memory is divided into 512-word segments, each of which can be either protected or unprotected. Segments not designated as protected are, by definition, unprotected. The protected/unprotected status of each segment is stored in the MP in eight 16-bit mask registers that are loaded by I/O instructions from the processor. These mask registers can store the status of up to 128 memory segments (65K).

MP monitors the address of the instruction being processed, the address specified by the next instruction in sequence, and the address specified by the effective address. Using this information and the status of the stored segment, the MP detects and operates on errors.

When a program is executed in unprotected memory, any of the following operations constitutes an error:

- Overflowing into a protected segment
- Writing into a protected segment
- Jumping into a protected segment
- Executing an I/O instruction in an unprotected segment
- Executing a halt instruction in an unprotected segment

When MP detects an error, execution of the current instruction is allowed to complete. However, if the instruction specifies writing or I/O operations, the contents of the A, B, and X registers, and memory are not modified and the I/O command is not issued to the I/O bus. The program is then interrupted and directed to one of five preassigned memory addresses. From this address, the program is directed to a user-written service subroutine for error analysis and correction. Table 2-3 lists the instructions for MP.

Table 2-3. MP Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 045	100045	Select mask register 0/4
EXC 145	100145	Select mask register 1/5
EXC 245	100245	Select mask register 2/6
EXC 345	100345	Select mask register 3/7
EXC 645	100645	Enable memory protection
EXC 745	100745	Disable memory protection
	Data Transfer	
CIA 045	102545	Clear and input to A register from instruction address register
CIB 045	102645	Clear and input to B register from instruction address register
INA 045	102145	Input to A register
INB 045	102245	Input to B register
IME 045	102045	Input to memory
OAR 045	103145	Output from A register to mask register
OBR 045	103245	Output from B register to mask register
OME 045	103045	Output from memory to mask register

SECTION 3 - CORE MEMORY

General Description

Two types of magnetic core memory modules are available with the Varian 73 computer. A standard 8K core memory is supplied with the Varian 73 and a 16K core memory module is available as an option. The first part of this section describes the features common to both memory types; the remaining text describes the 8K core memory and 16K core memories, respectively. Both core memory types are physically interchangeable and use a standard 16-bit word or optional 18-bit word.

The basic information storage element of both types is the magnetic core. The core is a toroid of ferrite that can be magnetized in two discrete directions representing a binary one or zero. The core is magnetized by a current passing through it. The direction of current flow through the core determines the direction of magnetization, either read or write.

The core memory uses the coincident-current technique for core magnetization. Two perpendicular wires (X drive and Y drive) pass through each core. During memory operations, the current on each drive wire is approximately on-half the current necessary to magnetize a core; thus, only the core at the intersection of two activated drive wires is magnetized.

Operations

A memory full-cycle consists of a reading sequence followed by a writing sequence.

The full-cycles are:

a. **Clear/write:** a word is transferred from the processor to memory.

b. **Read/restore:** a word is transferred from memory to the processor.

Both full-cycle operations require 660 nanoseconds (990 or 1200 nanoseconds for 16K memory), and memory access time is 350 nanoseconds (or 600 nanoseconds for 16K memory).

A clear/write operation loads the memory. The clear half-cycle of the operation resets the addressed cores to zero (read current); the write sequence then loads the data in the cores.

A read/restore operation reads information from the memory. The read half-cycle of the operation unloads the addressed data word from memory; the restore sequence immediately reloads (writes) the word in the same location.

Wrap-Around Addressing

Wrap-around addressing is a standard feature of the core memory. Wrap-around addressing techniques automatically prevent the processor from trying to address data to or from nonexistent memory addresses. Without this capability data read out of a nonexistent address results in zero and data written in are lost.

Since the processor registers are used as address sources, an address to memory can specify a nonexistent memory cell.

Thus, a computer with a 32,768 word memory (cells 0 through 32,768) can generate an address specifying nonexistent cells 32,768 through 65,536. The wrap-around address features makes it possible to predict the result when this occurs. In a computer with 32K of memory, the wrap-around feature directs all nonexistent cells to corresponding addresses between 0 and 32,768. Therefore, wrap-around addressing only words on 32K, 64K, 128K, or 256K increments. Addressing a nonexistent cell results in one of the following actions:

a. A predictable cell of existing memory is accessed if the addressing sequence has reached a wrap-around point and has started back through memory.

b. An imaginary memory cell is accessed and transfers zeros as data if the addressing sequence has reached a value above memory capacity but not the next wrap-around point.

Memory Interleaving

Memory interleaving is an optional feature that increases the performance of the memory. With memory interleaving, even addresses are located in one 8K module (or 16K module) and odd addresses in another 8K module (or 16K module). This permits instruction-execution cycles to overlap thus decreasing the time required to run a program. Memory interleaving is normally installed at the factory and consists of changing address jumpers on the memory PC card.

During interleaving, memory modules are active simultaneously and thus draw more current. To prevent power supply overloading. Special power supply and chassis configurations may be required.

8K Memory

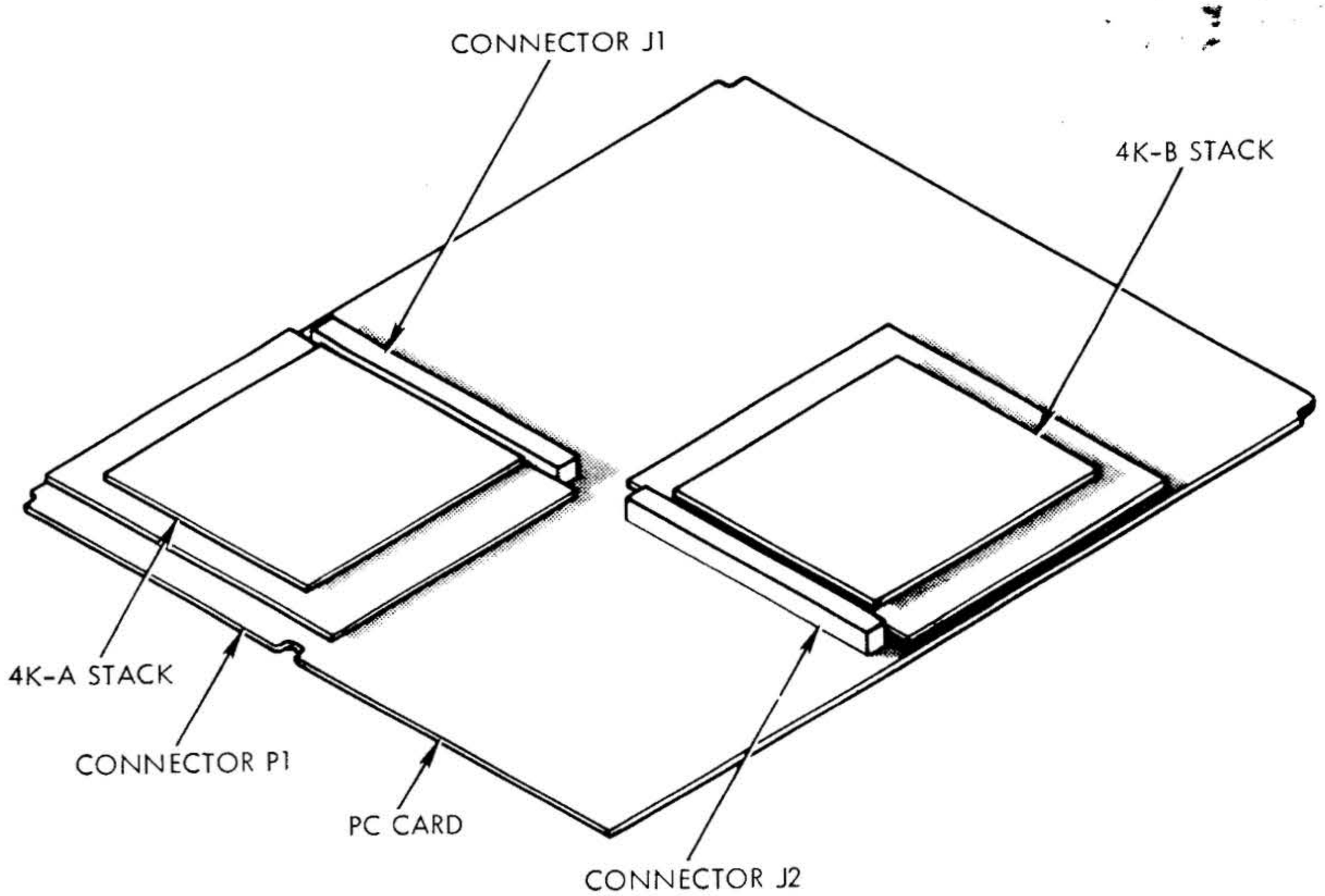
The standard 8K core memory is a dual-port, random-access, three-wire/three-dimensional, memory for storing instructions and data. This core memory is expandable in increments of 8K modules. Through jumper selection, 8K modules of semiconductor and core memory may be intermixed in any order. Jumper selection may also provide odd/even address interleaving between core memory modules.

The standard 16-bit word contains two 8-bit bytes which, in some systems, can be operated on independently. An optional 18-bit word is available to provide a parity bit for each byte.

The two memory ports allow simultaneous access to different memory modules from multiple sources such as main and peripheral processors.

An 8K memory module (figure 3-1) consists of a 15.6-by-19-inch (39.6 by 48.3 cm) printed-circuit (PC) card and two 4K stack cards. One side of the PC card contains two 130-pin connectors (J1 and J2) to accommodate the two 4K stack cards. Each memory stack card contains a diode-decoding matrix and a planar magnetic core array composed of sixteen 4K mats (eighteen 4K mats for 18-bit memory).

The other side of the PC card contains the electronic components of the memory module. The PC card also contains a 132-pin card-edge connector (P1) that interconnects the memory module with the processor. The height of the memory module is approximately one inch (2.54 cm).



VTI1-3233

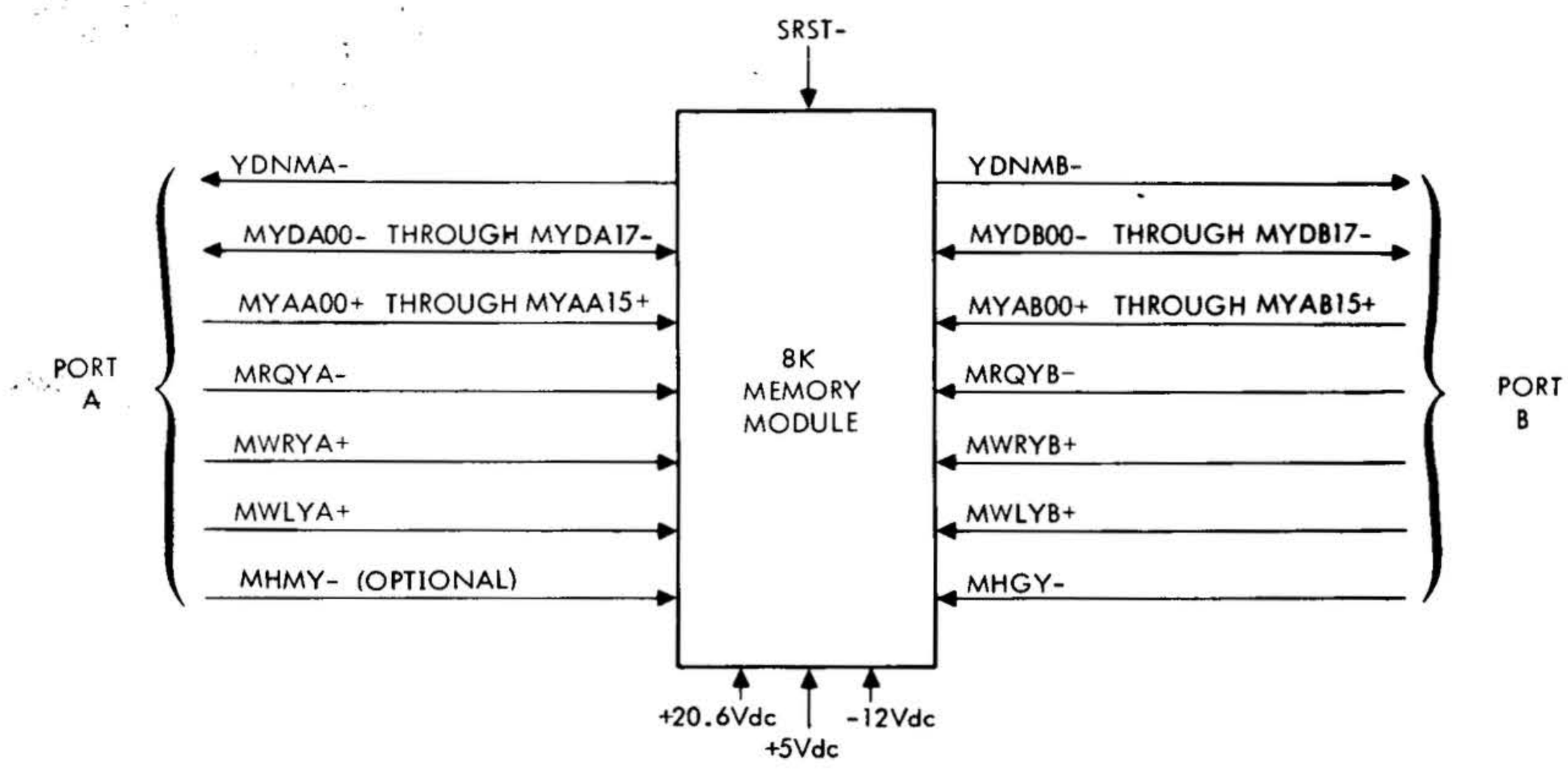
Figure 3-1. 8K Core Memory Module

Interfacing and Timing

Figure 3-2 is a block diagram showing the input and output signals for a single 8K memory module. Except for the port B override signal (MHGY–), the same type of signals are used for ports A and B. This makes each port independent allowing two memory modules to operate simultaneously, one on each port. Figure 3-3 shows the signal lines of ports A and B for an expanded memory system.

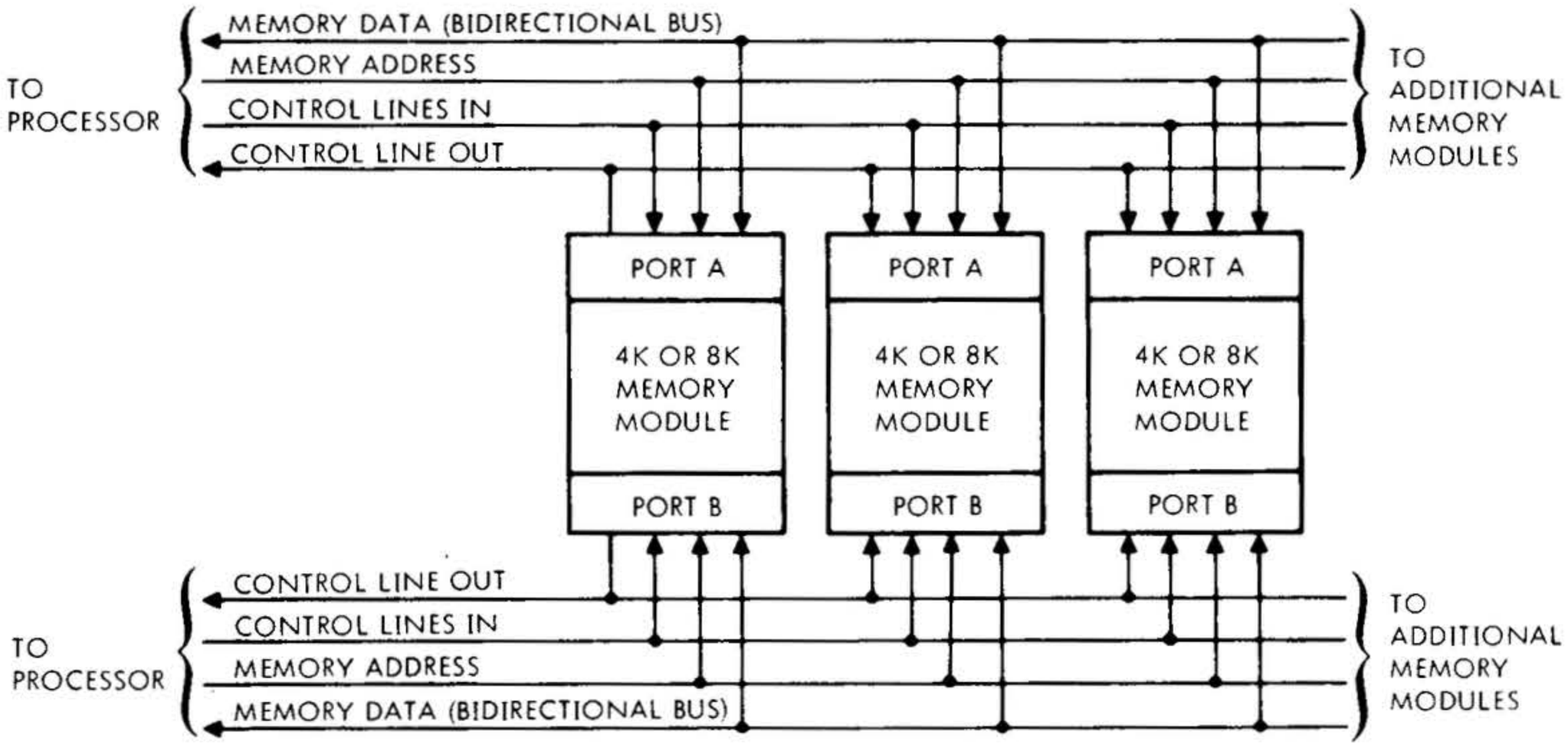
Typical memory interface waveforms are illustrated in figure 3-4. The memory cycle is initiated when the negative transition of the memory start request (MRQYx–) is received by port A or B. The memory first resolves priority in case of conflicting requests on the two ports, and then begins its memory timing sequences.

A memory acknowledge signal (YDNMx–) is generated 140 nanoseconds (maximum) before read data is stable on the memory data bus. The pulse width of YDNMx– is typically 180 nanoseconds, and its trailing edge (positive-going transition) is used to clock the memory read data into a data register in the processor. Data is stable 15 nanoseconds (minimum) before the trailing edge of YDNMx– and remains stable for 35 nanoseconds (minimum) after the trailing edge of YDNMx– .



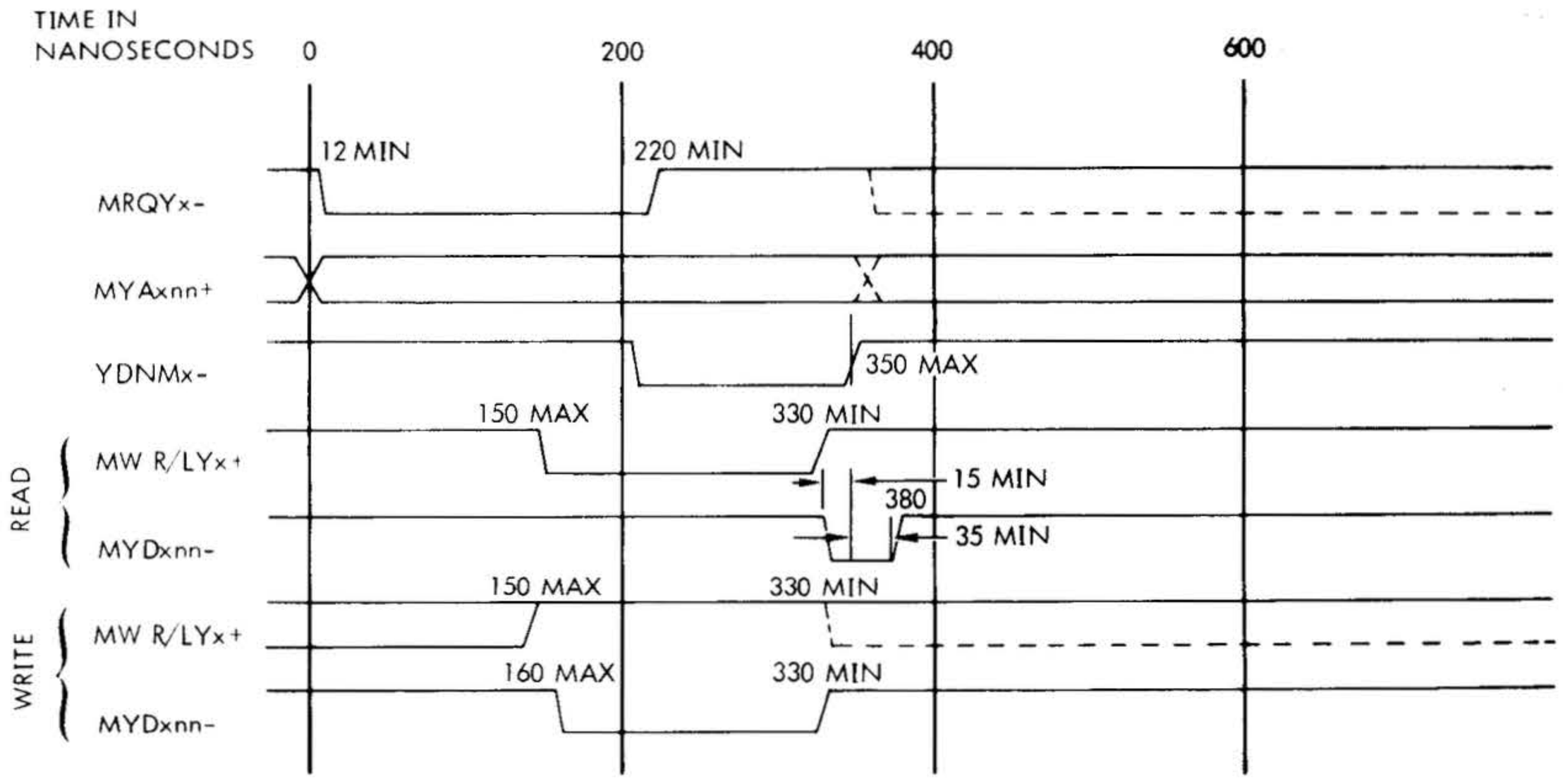
VTII-3234

Figure 3-2. 8K Memory Module Interface Block Diagram



VTII-3235

Figure 3-3. 8K Memory Module Expansion Block Diagram



VTII-3236

Figure 3-4. Typical Memory Interface Waveforms (8K Module)

For a clear/write operation, the read/write control signal (MWRYx+ or MWLYx+) must be received by the memory no later than 140 nanoseconds after the leading edge of the memory start request signal.

The memory writes full or half words (bytes) as commanded. Bytes which are not written into retain previously written data.

Specifications

Core memory specificiations are listed in table 3-1.

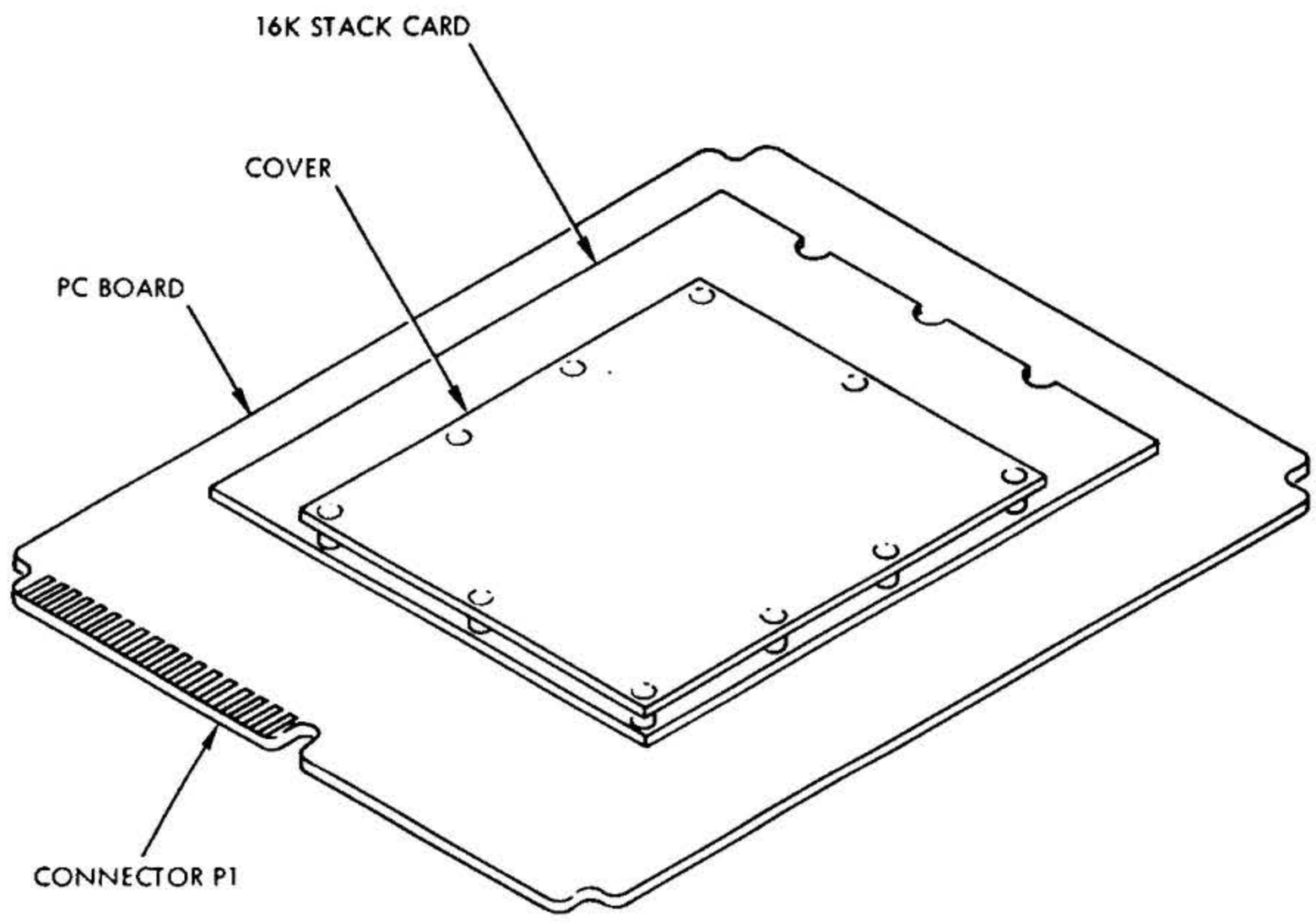
16K Memories

The optional 16K core memories are single-port, expandable, random-access, three-wire/three-dimensional memories. There are 16K by 16-bit (or 18-bit) memories for storing instructions and data. Each 16K memory module is contained on a single board. They have cycle times of 990 and 1200 nanoseconds and can be interleaved with the 660 nanosecond memory.

Each 16K memory module (figure 3-5) consists of a 15.6 by 19-inch printed-circuit (PC) board and one 16K stack card. The height of the complete module is 1.1 inch. On the underside of the stack card, groups of pin terminals of a 126-pin connector (J1) are located around the perimeter of the card and plug into corresponding jack terminals located on the etched surface of the PC board. The other side of the PC board contains the electronic components of the memory module. The PC board has a 132-pin edge connector (P1) that connects the module to the processor through a cable.

Table 3-1. Specifications for 8K Core Memory

Parameter	Description
Timing	Cycle time: 660 nanoseconds (minimum) Access time: 350 nanoseconds (maximum) Write data available from 150 (minimum) to 330 nanoseconds (maximum).
Word Length	16 bit words containing two 8-bit bytes. An optional 18-bit word provides a parity bit for each byte.
Size	Expandable to maximum memory size of system (see system specifications in section 1) in 8,192 word increments.
Temperature	
Operating	0 to 50 degrees C
Storage	-20 to 70 degrees C
Humidity	
Operating	To 90 percent without condensation
Storage	To 95 percent without condensation



NOTE:
PINS OF J1 ARE LOCATED AROUND PERIMETER OF STACK CARD

VTII-3237

Figure 3-5. 16K Core Memory Module

Interfacing and Timing

Figure 3-6 is a block diagram showing the input and output signals for a single memory module. Figure 3-7 shows the signal lines for an expanded memory system.

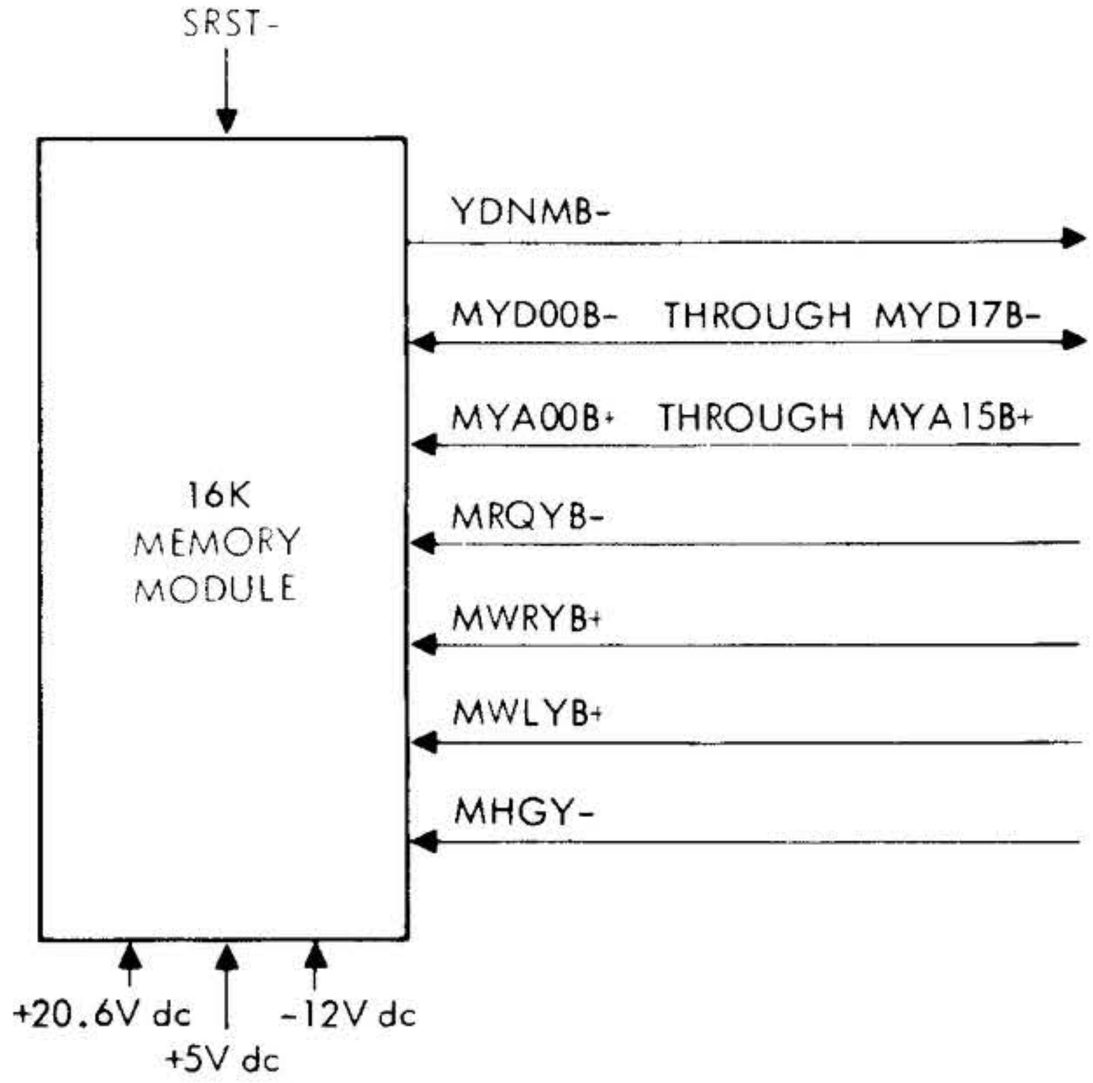
Typical interface waveforms for the 1200-nanosecond memory are illustrated in figure 3-8. The memory cycle is initiated when the negative transition of the memory start request (MRQYB—) is received.

The pulse width of YDNMB– is typically 185 nanoseconds, and its trailing edge (positives going transition) is used to clock the memory read data into a data register in the processor. Data is stable 30 nanoseconds (minimum) before the trailing edge of YDNMB– and remains stable for 60 nanoseconds (minimum) after the trailing edge of YDNMB– .

For a clear/write operation, the read/write control signal (MWRYB + or MWLYB +) must be received by the memory no later than 265 nanoseconds after the leading edge of the memory start request signal. The memory writes full or half words (bytes) as commanded. Bytes which are not written into retain previously written data.

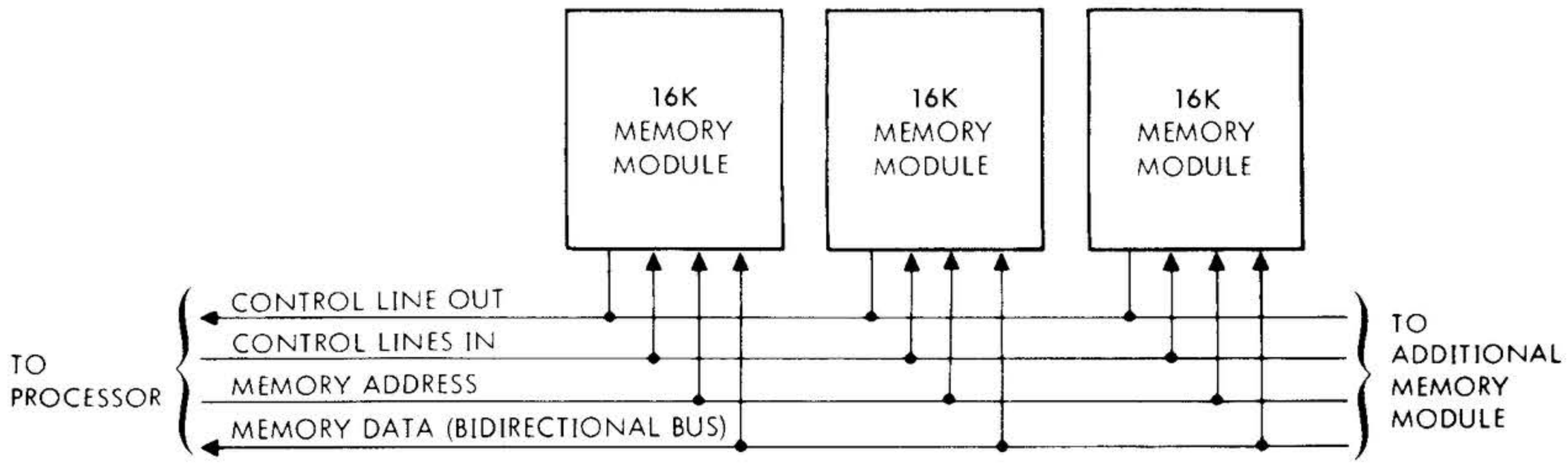
Specifications

The optional core memory specifications are listed in table 3-2.



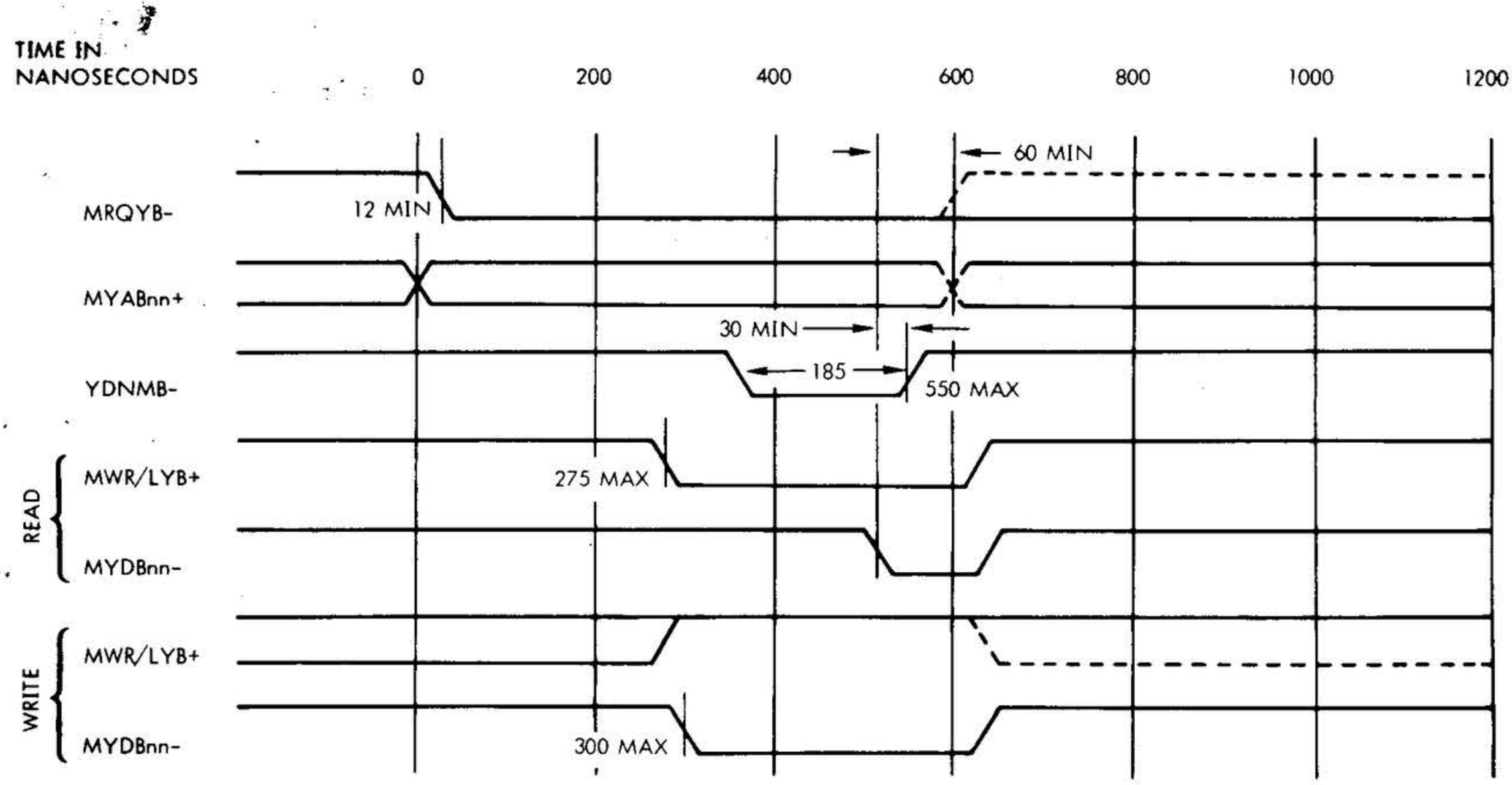
VTI1-3239

Figure 3-6. 16K Memory Module Interface Block Diagram



VTI1-3238

Figure 3-7. 16K Memory Module Expansion Block Diagram



VTI1-3241

Figure 3-8. Typical Memory Interface Waveforms (16K Module 1200 ns)

Table 3-2. Specifications for 16K Core Memory (990 and 1200 ns)

Parameter	Description
Timing	Cycle time: 990 or 1200 nanoseconds (minimum) Access time: 600 nanoseconds (maximum) Write data available from 275 (minimum) to 610 nanoseconds (maximum)
Word Length	16-bit words containing two 8-bit bytes. An optional 18-bit word provides a parity bit for each byte Expandable to 65,536 words in 16,384 word increments.
Temperature	
Operating	0 to 50 degrees C
Storage	−20 to 70 degrees C
Humidity	
Operating	To 90 percent without condensation
Storage	To 95 percent without condensation

SECTION 4 - SEMICONDUCTOR MEMORY

The **Varian 73 system semiconductor memory** is an expandable, random-access, semiconductor, asynchronous memory system with an internal cycle time of 330 nanoseconds.

Memory Design

The memory storage arrays are packaged in 22-pin dual-in-line ceramic or plastic cases. The memory elements are dynamic in operation and the data stored is volatile with respect to the power. However, there is a Data Save mode (battery backup source) that is programmed to refresh the memory during low power or power loss.

Each printed circuit (PC) board can accommodate up to 8K of 16-bit words in increments of 1K, 2K, 4K, or 8K. An option 18-bit word provides a parity bit for each 8-bit byte.

The memory system is dual port, i.e., two independent sets of input/output terminals to access memory. Port B has priority over port A. Port B can also request continuous service by a signal that modifies the priority logic. A PC board memory bus at the rear of the mainframe or memory expansion chassis interconnects all circuit boards for maximum speed at minimum length.

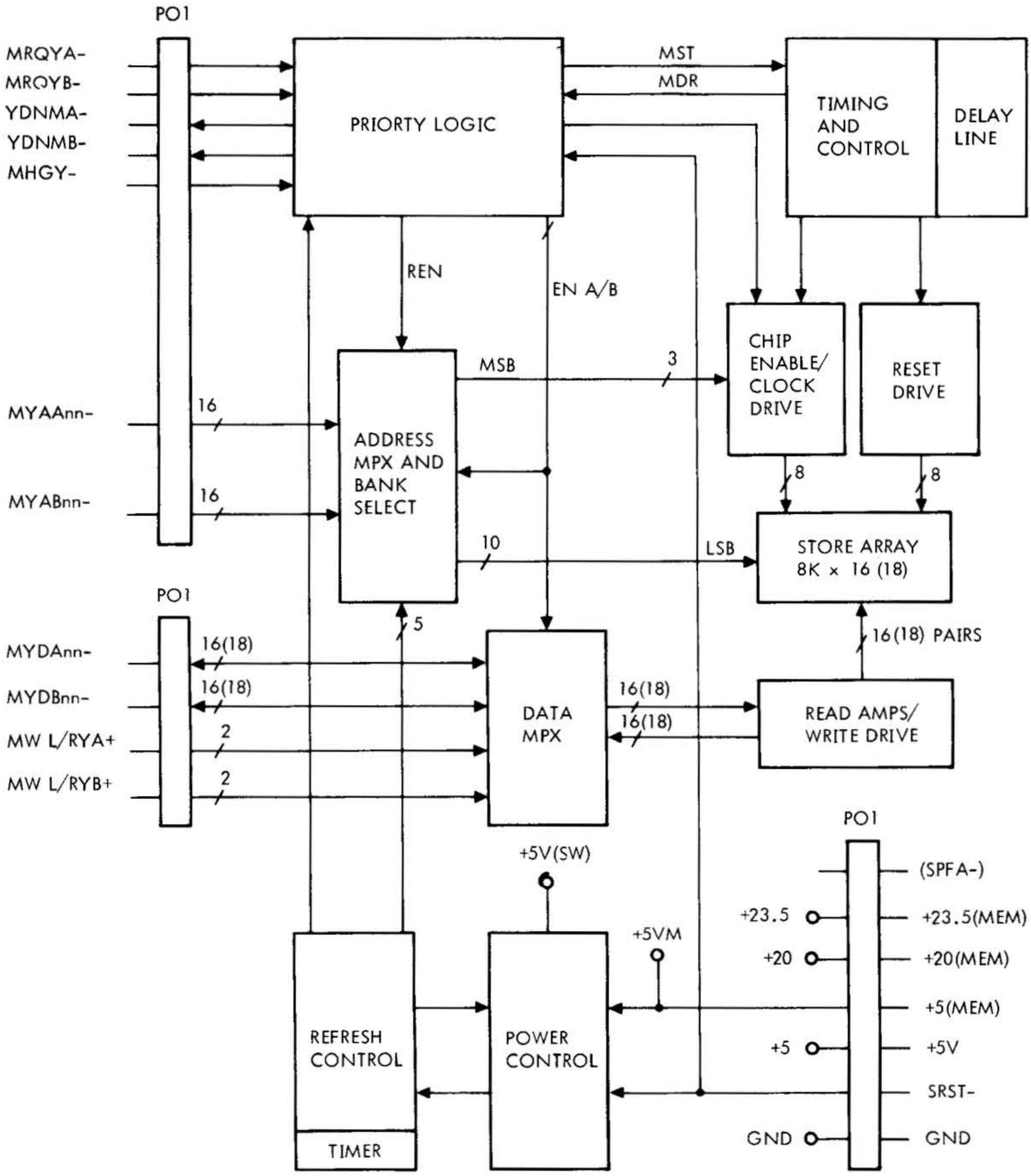
The major functional logic blocks of the semiconductor memory are: priority, timing and control, delay line, address multiplex and bank select, data multiplex, refresh, power control, storage array, reset drive, chip enable/clock driver, write drivers, and read amplifiers (figure 4-1).

Memory Operations

A memory full cycle consists of a read and write sequence. Because the memory storage arrays are dynamic, they are read from and written into nondestructively (no clear/write, read/restore operation). Thus, the cycle and access times are nearly equal with full cycle at 330 nanoseconds and access at 260 nanoseconds.

Instruction words read from memory are transferred to the control section for execution. Words are transferred, under program control, from memory to the arithmetic unit, operational register, or the I/O bus. Also under program control, they are transferred to memory from the operation register or the I/O bus.

When the semiconductor memory receives a start request, it determines the requesting address, examines the HOG line for priority modification, and initiates a memory cycle, if appropriate. The write information is converted from the data bus of the requesting port to high levels (+19V) for the write inputs to the memory arrays. The write drivers are sectored to provide 8-bit data storage. The read amplifiers contain sixteen sense amplifiers that discriminate the one and zero signals from the memory arrays by time and amplitude. Since there is no data register in the memory, output pulses from memory to data bus are determined by the width of the signals and gated by a timed strobe.



VTII-1467

Figure 4-1. Semiconductor Memory System Block Diagram

Interfacing and Timing

The semiconductor memory interfaces via the memory bus, with the processor, option board, core memory, another memory bus (dual port), and the writable control store, if used. All signals on the memory bus are buffered through the interfacing TTL logic to help eliminate noise problems and cross-talk. The address and control lines are unidirectional and the data lines are bidirectional. All interconnection of the boards in the mainframe or memory expansion chassis is made with a PC board that connects at the rear. This keeps the input/output lines to a minimum length.

A typical timing sequence is shown in figure 4-2. The memory cycle is initiated by the falling edge of request line MRQYA – after the bank addresses have been stable for at least 12 nanoseconds. Status is examined and priority resolved for conflicting requests on the two ports. After priority is resolved, the memory begins its timing sequences by driving a voltage pulse down the delay line initiating the read or write sequences.

The read data sequence output is activated by lowering the YDNM – signal 130 nanoseconds before the data is stable at the output. The width of YDNM – is at least 100 nanoseconds and the rising (trailing) edge is a strobe that latches the data into the processor data register. During a write sequence, the write command signal MWR/LY must be received at the memory no later than 140 nanoseconds after the leading edge of MRQYA –. The memory writes half (8-bit) or full (16-bit) words, as commanded. Bytes not written into will retain previously written data.

Wrap-Around Addressing

Wrap-around addressing is available (upon request) as a standard feature of the semiconductor memory. Wrap-around addressing techniques automatically prevent the processor from trying to address data to or from nonexistent memory addresses. Without this capability, data read out of a nonexistent address result in zero and data written in are lost.

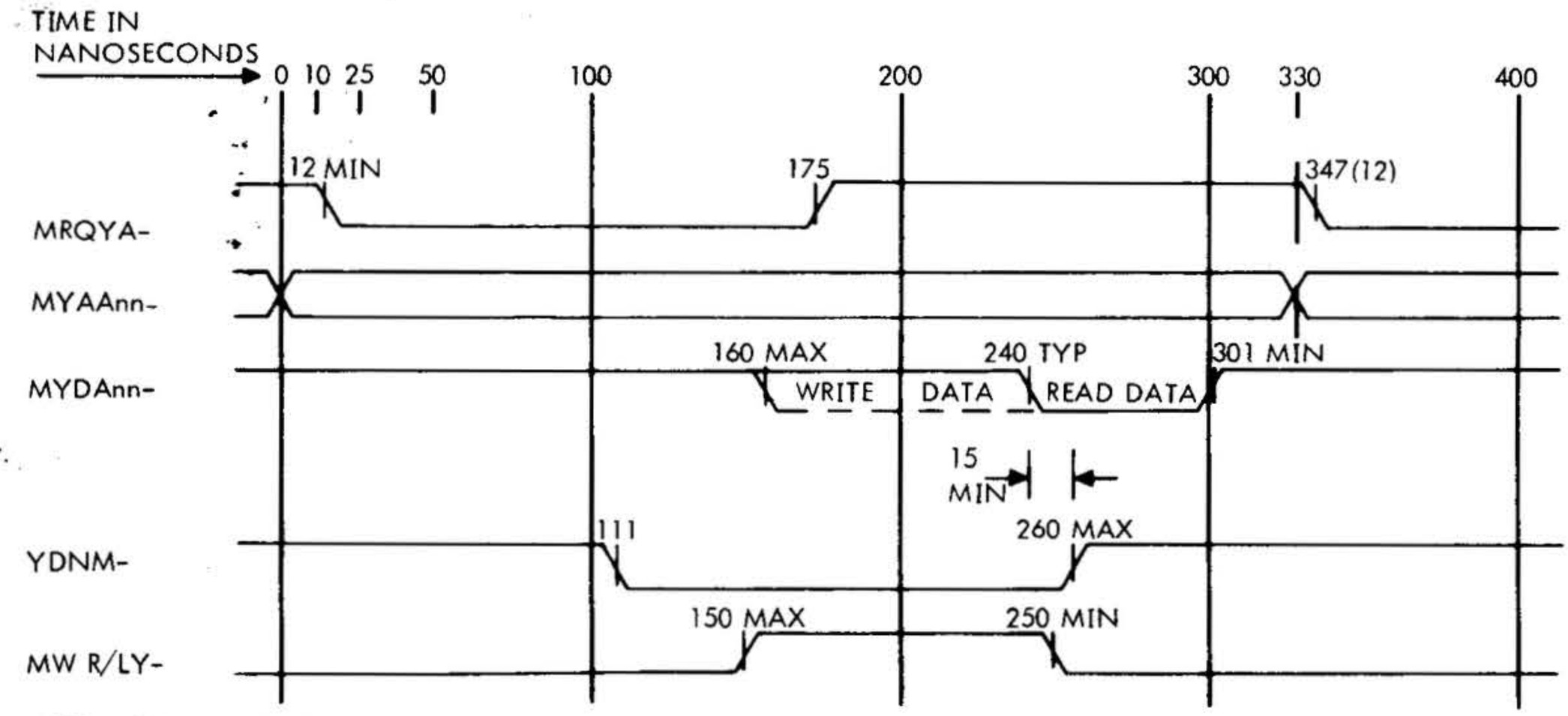
Since the processor registers are used as address sources, an address to memory can specify a nonexistent memory cell. Thus, a computer with a 8,192 word memory (cells 0 through 8,192) can generate an address specifying nonexistent cells 8,192 through 32,768. The wrap-around address feature makes it possible to predict the result when this occurs. In a computer with 8K of memory, the wrap-around feature directs all nonexistent cells to corresponding addresses between 0 and 8,192 Therefore, wrap-around addressing only works on 8K, 16K, 32K, 64K, 128K, or 256K increments. Addressing a nonexistent cell results in one of the following actions:

a. A predictable cell of existing memory is accessed if the addressing sequence has reached a wrap-around point and has started back through memory.

b. An imaginary memory cell is accessed and transfers zeros as data if the addressing sequence has reached a value above memory capacity but not the next wrap-around point.

Specifications

Performance specifications for the semiconductor memory system are listed in table 4-1.



NOTE: TIMES ARE RELATIVE TO STABLE ADDRESS AT MEMORY CONNECTOR.

VTII-1504

Figure 4-2. Typical Timing Sequence

Table 4-1. Memory Specifications

Parameter	Description
Design	Parallel, random-access, expandable, metal oxide semiconductor (MOS)
Cycle time	330 nanoseconds
Access time	260 nanoseconds
Execution times	Write data available from 160 to 240 nanoseconds; memory start pulse width 163 nanoseconds
Operating modes	Continuous at any address, and in burst mode (i.e., memory cycle series interspersed by non-operating time)
Operating environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

SECTION 5 - OPTIONS

Several options are available to enhance the performance of the Varian 73 system. The options are:

a. Automatic Bootstrap Loader (paper tape or rotating memory)

b. Memory Map

c. Memory Parity

d. Writable Control Store (WCS)

e. Priority Interrupt Module (PIM)

f. Buffer Interlace Controller (BIC)

g. Priority Memory Access (PMA)

h. Block Transfer Controller (BTC)

i. Floating Point Processor (FPP)

Automatic Bootstrap Loader

The automatic bootstrap loader option automatically loads the bootstrap program into computer memory from an integrated-circuit read only memory (control store in the processor). This option saves the time and effort involved in manually loading the bootstrap program. Three models of the automatic bootstrap loader are available allowing a choice of input devices:

a. TTY (standard)

b. Model 73-3001 paper tape

c. Model 73-3002 rotating memory

Memory Map

The memory map option performs address relocation and memory protection for up to 256K words of physical memory by translating the 16-bit logical memory address and a 4-bit key into an 18-bit physical address. Mapping operations can be performed independently in up to sixteen 32K logical memory areas. A 64K mode of operation is available to provide eight 64K logical memory areas. Map numbers 0 through 15 are used to identify the logical memory areas, with map 0 being reserved for the VORTEX II operating system. All memory addresses are mapped into physical memory pages consisting of 512 words each. Page assignments are controlled by the VORTEX II page-allocating routine.

The memory map specifications are listed in table 5-1.

Memory Parity

The memory parity option generates and checks parity for left and right bytes when accessing memory. Parity bits are generated during the memory writing sequence and are checked during the reading sequence. This option contains its own control and interrupt logic. The memory parity specifications are listed in table 5-2.

There are two external control instructions that enable (0445) and disable (0545) parity. Parity is generated and checked, but an interrupt is not generated if PARITY is disabled.

Table 5-1. Memory Map Option Specifications

Parameter	Description
Physical Memory Size	Up to 256K words
Logical Memory Size	Up to 32K words
Page Size	512 words
Number of Logical-Memory Name Spaces	Up to 16 with 32K words
Memory Access Times	With the memory map active, memory access is delayed 104 nanoseconds for the first 64K of memory and 156 nanoseconds for the memory above 64K. With the memory map inactive, memory access is delayed by 27 nanoseconds. These are worst-case delays for standard operating modes
System Configuration	Provides mapping of address for processor, DMA, and PMA on one memory port. Mapping on more than one port requires one memory map for each port (not supported by VORTEX II)
Loading	The memory map array is loaded and read via high-speed DMA operations. The loading word rate is 715 kHz; the reading word rate is 358 kHz
Operating Modes	User mode, executive mode, and inactive mode
Priority Assignments	The memory map's memory-protection feature is assigned the highest system priority. Priority assignment for DMA operation is made independently
PIM Capability	The memory map is not used with a priority interrupt module (PIM)

Table 5-1. Memory Map Option Specifications *(continued)*

BIC Control	Mapping operations can be performed using the buffer interlace controller (BIC)
BTC Control	Mapping operations can be performed using the block transfer controller (BTC)
Logic Levels (internal)	High = +2.4 to +5.0V dc Low = 0 to +0.4V dc
Logic Levels (I/O bus)	High = +2.8 to +3.6V dc Low = 0 to +0.5V dc
Dimensions	Contained on a 15.6-by-19-inch (39.6 by 48.3 cm) printed-circuit board
Installation	Plugs into a Varian 72 mainframe chassis using one module slot
Input Power	+5V dc at 17 amperes
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 5-2. Memory Parity Specifications

Parameter	Description
Mode of Operation	Interrupt logic controls the interrupt cycle
I/O	I/O interface via the option bus

Table 5-2. Memory Parity Specifications *(continued)*

Logic Levels	Positive logic:	
	True:	+2.40 to +5.25V dc
	False:	0.00 to +0.45V dc
	Negative logic:	
	True:	0.00 to +0.45V dc
	False:	+2.40 to +5.25V dc
Input Power	+5V dc at 0.5 ampere	
Temperature		
Operating	0 to 50 degrees C	
Storage	−20 to 70 degrees C	
Humidity		
Operating	To 90 percent without condensation	
Storage	To 95 percent without condensation	

Writable Control Store

The writable control store (WCS) option extends the Varian 73 processor's read-only control store to permit addition of new instructions, development of microdiagnostics, and optimum tailoring of the system computer to its application. Unlike the read-only control store, which contains the Varian 73 basic instruction set and cannot be altered, the writable control store can be loaded from the computer system's main memory under control of certain I/O instructions. The capability of altering the contents of the writable control store gives the user complete access to the resources of the Varian 73 system computer.

Supporting software for the writable control store includes a utility loader, a machine micro simulator, and a micro assembler. The writable control store can operate in a stand-alone environment or under control of the MOS or VORTEX operating system. A test program is also provided to assist in maintaining the writable control hardware. The writable control store specifications are listed in table 5-3. Table 5-4 lists the instructions for the WCS.

Table 5-3. WCS Option Specifications

Parameter	Description
Cycle time	190 nanoseconds.
Capacity	Up to three WCSs (2048 words including read-only control store) for each computer system.
Selection	WCSs can be selected for operation over the I/O bus by an I/O instruction, BCS instruction, or by a processor page-branch microinstruction
Control	WCSs are loaded and read under programmed I/O control. WCSs can also be loaded under control of a buffer interlace controller (BIC). Loading can be performed in one WCS while executing from another.
Dimensions	Contained on a 15.6 by 19 inch printed-circuit board.
Installation	Plugs into a Varian 70 series mainframe chassis using one module slot
Input power	+5 V dc, 14 amperes for model 7x-4000, 20 amperes for model 7x-4001, 26 amperes for model 7x-4002
Operational environment	0 to 50 degrees C, 0 to 90 percent relative humidity without condensation.

Table 5-4. WCS Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 07X*	10007X*	Initialize
EXC 17X	10017X	Enables processor clock
EXC 27X	10027X	Step operation of central control store
EXC 37X	10037X	Initialize BIC load
EXC 47X	10047X	Not used
EXC 57X	10057X	Not used
EXC 67X	10067X	Not used
EXC 77X	10077X	Resets stack pointer
	Data Transfer	
OAR 07X	10317X	Transfers address and control functions from A register to writable control store
OBR 07X	10327X	Transfers address and control functions from B register to writable control store
OME 07X	10307X	Transfers address and control functions from main memory to writable control store
OAR 07Y**	10317Y**	Transfers data from A register to writable control store
OBR 07Y	10327Y	Transfers data from B register to writable control store
OME 07Y	10307Y	Transfers data from main memory to writable control store
INA 07Y	10217Y	Transfers data from writable control store to A register
INB 07Y	10227Y	Transfers data from writable control store to B register
IME 07Y	10207Y	Transfers data from writable control store to main memory
CIA 07Y	10257Y	Transfers data from writable control store to cleared A register
CIB 07Y	10267Y	Transfers data from writable control store to cleared B register

Table 5-4. WCS Instructions *(continued)*

Program Sense		
SEN 07X	10107X	Senses if writable control store is busy
SEN 17X	10117X	Senses if subroutine stack is empty
SEN 27X	10127X	Senses if subroutine stack is full
SEN 37X	10137X	Senses if BIC load is in process

* X equals 0, 2, or 4
**Y equals 1, 3, 5, when X equals 0, 2, 4, respectively

Priority Interrupt Module (PIM)

The PIM provides for the orderly servicing of peripheral-initiated interrupts of a program in progress in the computer. It does so by:

a. Establishing up to eight levels of interrupt priority for selected peripheral controllers.

b. Storing interrupt requests originated by associated peripheral controllers and placing the requests on the I/O bus in the order of the established priority.

In effect, the PIM organizes a "priority-within-a-priority" system. Peripheral controllers that cannot normally initiate an interrupt because of their inability to generate memory addresses can do so when connected, via a line in the interrupt cable, to the PIM. PIM-controlled priority assignments are prewired at the factory to user specifications. The PIM specifications are listed in table 5-5. Table 5-6 lists the instructions for the PIM.

Table 5-5. PIM Specifications

Parameter	Description
Organization	Contains line, synchronization, and mask registers; an interrupt address generator; priority and control logic; and line drivers and receivers
Control Capability	Establishes and implements eight levels of interrupt priority (user-assigned) for system peripherals
I/O Capability	Five external control and three transfer instructions
Standard Device Address	040 through 043
Interrupt Addresses	First PIM: 0100 through 0117 Second and succeeding PIMs: 0120 through 0177
System Priority Assignment	Determined by location in the priority chain (user-selected)
Logic Level	
Internal	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
I/O Cable	Negative logic: True: 0.0 to +0.45V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 by 30.3 cm) etched-circuit card
Interconnection	Plugs into the mainframe I/O backplane; (additional PIMS interface via the I/O cable) connects to peripheral controllers via the backplane and/or I/O expansion cable
Input Power	+5V dc at 0.45 ampere
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 5-6. PIM Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 014*	10014*	Clear interrupt registers
EXC 024*	10024*	Enable PIM
EXC 0244	100244	Enable all PIMs in system
EXC 034*	10034*	Clear interrupt registers and enable PIM
EXC 044*	10044*	Disable PIM
EXC 0444	100444	Disable all PIMs in system
EXC 054*	10054*	Clear interrupt registers and disable PIM
	Data Transfer	
OME 04*	10304*	Transfer contents of memory to mask register
OAR 04*	10314*	Transfer contents of A register to mask register
OBR 04*	10324*	Transfer contents of B register to mask register

* Represents the last octal digit of the device address

Buffer Interlace Controller (BIC) (for 620 compatible DMA)

The BIC implements the transfer of blocks of data directly to and from computer memory and peripheral controllers. Cycle-stealing trap requests inhibit the processing of a stored program for only the memory cycle required to transfer one word of data directly between memory and a system peripheral, i.e., 660 nanoseconds.

Operation register contents are not changed by the transfer, thus freeing the CPU to execute an instruction from the stored program between successive data word transfers.

The BIC monitors trap requests initiated by the stored program, or by magnetic-tape units, disc memories, card and paper-tape readers and punches, and analog-to-digital system controllers. Up to ten such devices can be connected to the BIC. The computer system can include up to four BICs.

The BIC is designed to operate on the 620 compatible DMA.

The BIC will perform DMA transfers at the peripheral device rate up to a maximum rate defined as follows:

$$R_{max} = \frac{1}{\frac{1}{R_{CPU\ max}} + T_{IUCX}}$$

where:

R_{max} is the maximum rate through a BIC (words/second)
R_{CPU} is the maximum DMA rate for the processor (words/second)
T_{IUCX} is the period of interrupt clock (seconds)

As an example, the maximum DMA rate for a Varian 70 series computer with core memory and a 990 nanosecond interrupt clock period is 361,800 words/second. The maximum rate through the BIC is then:

R_{max}:

$$\frac{1}{\frac{1}{361{,}800} + (990 \times 10^{-9})} = 266{,}383 \text{ words/second}$$

Note that the BIC is included on the system priority chain; peripheral controllers connected to it have no priority of their own.

When the memory map option is used in conjunction with the BIC an additional register is used in the BIC to store the key bits. The BIC specifications are listed in table 5-7. Table 5-8 lists the instructions for the BIC.

Priority Memory Access (PMA)

The priority memory access (PMA) is a mainframe option for the Varian 73 system computer. The PMA option interfaces with PMA controllers to provide logic functions to interface four data transfer channels with the memory in a hardware-fixed priority. All signals can be asynchronous to free the PMA option/PMA controller interface from circuit and cable speed dependence. Also there is the capability to handle a variety of circuits and data rates, or synchronous operation for maximum rate of data transfer. Specifications for the PMA option are given in table 5-9.

The PMA option can interface with up to eight PMA controllers distributed in any manner among the four priority levels although only one PMA controller per level can be active at a time. Each PMA controller interfaces with the PMA logic circuits on the option board in the mainframe and with the I/O bus for program control.

Block Transfer Controller (BTC)

The block transfer controller (BTC) interfaces the PMA option to a peripheral controller requiring fast access, high-speed transfer of data in or out of processor memory. The BTC provides memory address control when the transferred data are organized into "blocks" of 16-bit words.

The BTC contains a key-bit register for use with the memory map option and an odd parity generator which generates parity bits for PMA input transfers in parity systems.

Table 5-7. BIC Specifications

Parameter	Description
Organization	Contains an initial address register, a final address register, sequence controller, and drivers and receivers
Control Capability	Implements trapping operations for up to ten peripheral controllers
I/O Capability	Two external control, eleven transfer, and two program sense instructions
Transfer Rate	Synchronized with peripheral
I/O Signal Limits	Rise/fall: 10 nanoseconds (minimum), 100 nanoseconds (maximum)
System Priority	Determined by location in the priority chain (user-selected)
Standard Device Address	First BIC: 020-021 Second BIC: 022-023 Third BIC: 024-025 Fourth BIC: 026-027
Logic Level	
Internal	Positive logic: True: +2.4 to +5.5V dc False: 0.0 to +0.5V dc
I/O Cable	Negative logic: True: 0.0 to +0.5V dc False: +2.8 to +3.6V dc
Size	One 7-3/4-by-12-inch (19.7 by 30.3 cm) etched-circuit card
Interconnection	Plugs into the mainframe I/O backplane, with control lines between the BIC and associated peripheral controllers in a separate chassis connect to the BIC via control lines in the I/O cable
Input Power	+5V dc at 0.6 ampere
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 5-8. BIC Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 020	100020	Activate BIC
EXC 021	100021	Initialize
EXC 0321	100321	Enable loading of key bits
	Data Transfer	
OAR 020	103120	Load initial register from A
OBR 020	103220	Load initial register from B
OME 020	103020	Load initial register from memory
OAR 021	103121	Load final register from A
OBR 021	103221	Load final register from B
OME 021	103021	Load final register from memory
INA 020	102120	Read initial register into A
INB 020	102220	Read initial register into B
IME 020	102020	Read initial register into memory
CIA 020	102520	Read initial register into cleared A
CIB 020	102620	Read initial register into cleared B
	Program Sense	
SEN 020	101020	Sense BIC not busy
SEN 021	101021	Sense abnormal device stop
SEN 0121	101121	Senses if BIC has been stopped due to a memory map error

Table 5-9. PMA Specifications

Parameter	Description
Organization	Contains control logic, priority logic, data transfer logic, data drivers, and input data receivers and address receivers
Priority Assignment	Exceeded only by the PF/R interrupt
Number of Priority Levels	Four with hardware-assigned priority
Maximum Number of Controllers per PMA	Eight distributed in any manner among four levels of priority (only one active controller per priority level at a time)
Maximum Cable Length	20 feet
Maximum Latency	1354 nanoseconds (20 feet)*
Minimum Latency	740 nanoseconds (20 feet)*
Maximum Transfer Rate	1,347,000 words/second (write)* 1,212,000 words/second (read)*
Control Signals	Control can be by interlocked responses allowing true asynchronous or synchronous operation
Logic Levels on PMA Bus	Positive logic True: +2.2 to +3.2V dc False: −0.5 to +1.4V dc Negative logic True: −0.5 to +1.4V dc False: +2.2 to +3.2V dc
Size	Circuits occupy a portion of the option board in the mainframe chassis

*Assumes that the time from acknowledge (ACK1−) to GO− equals approximately 130 nanoseconds.

Table 5-9. PMA Specifications *(continued)*

Interconnection	PMA bus connectors available at rear of mainframe on option board (J07A and J07B)
Power Consumption	+5V dc from the mainframe power supply
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 5-10. BTC Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 002x	10002x	Activate BTC
EXC 002y	10002y	Initialize BTC
EXC 012x	10012x	Reset BTC
EXC 012y	10012y	Test data transfer
EXC 022x	10022x	Test input (writing) cycle
EXC 022y	10022y	Test output (reading) cycle
	Data Transfer	
OAR 02x	10312x	Load BTC initial-address register
OBR 02x	15322x	Load BTC initial-address register
OME 02x	10302x	Load BTC initial-address register
OAR 02y	10312y	Load BTC final-address register
OBR 02y	10322y	Load BTC final-address register
OME 02y	10302y	Load BTC final-address register
INA 02x	10212x	Read initial-address register
INB 02x	10222x	Read initial-address register
IME 02x	10202x	Read initial-address register
CIA 02x	10252x	Read initial-address register
CIB 02x	10262x	Read initial-address register
INA 02y	10212y	Read final-address register
INB 02y	10222y	Read final-address register
CIA 02y	10252y	Read final-address register
CIB 02y	10262y	Read final-address register

Table 5-10. BTC Instructions *(continued)*

	Program Sense	
SEN 002x	10102x	Sense BTC not busy
SEN 002y	10102y	Sense abnormal device-stop
SEN 012x	10112x	Test sense-end-of-block

x is the last digit of the even address
y is the last digit of the odd device address (y = x + 1)

The BTC buffers one word of data into or out of the PMA option so that the peripheral controller timing requirements can be relaxed with respect to the fast PMA bus.

Peripheral controllers used with the BTC must be designed for PMA use since the PMA bus is separate from the E bus shared by I/O and DMA. Table 5-10 lists the instructions for the BTC.

Floating Point Processor (FPP)

The floating point processor option performs high-speed floating point arithmetic on single and double precisicn numbers. When installed in a V73 system computer the 56-bit floating point accumulator and all floating point instructions are fully integrated into the computer architecture both at the machine language programming level and at the FORTRAN level.

The FPP contains a direct memory access to increase data acquisition and storage speed. It implements pipelining of instructions to increase throughput. The floating point accumulator is a full 56 bits wide so that fully parallel arithmetic operations can be performed on double precision real numbers. The basic processor clock period is 165ns, however, all shifting operations occur in an 82.5ns period. Operations such as addition, where most of the operate time consists of accumulator shifts, can be accomplished at a high rate.

The FPP implements pipelining of instructions to increase throughput. The floating point accumulator is a full 56 bits wide so that fully parallel arithmetic operations can be performed on double precision real numbers. The basic processor clock period is 165n, however, all shifting operations occur in an 82.5ns period. Operations such as addition, where most of the operating time consists of accumulator shifts, can be accomplished at a high rate.

The FPP interrupt is disabled whenever either a jump and mark instruction is executed from another interrupt, or when EXC 0444 (disable PIM's) is executed. EXC 0244 (enable PIM's) will subsequently enable the FPP interrupt. This allows the FPP interrupt to be placed anywhere in the priority chain.

Table 5-11 lists the FPP specifications. The FPP instructions are listed in table 5-12.

Table 5-11. Floating Point Processor Specifications

Parameter	Description
Cycle time	165 ns
Interrupt address	016
Priority assignment	May be placed anywhere in the priority chain.
Dimensions	Contained on a 15.6 by 19 inch (39.6 by 48.3 cm) wire-wrap board.
Installation	Plugs into V73 mainframe chassis using three module slots.
Input power	+5V dc at 16 amps
Operational environment	0 to 50 degrees C (32 to 122 degrees F). 0 to 90 percent relative humidity without condensation.

Table 5-12. Floating Point Processor Instructions

Mnemonic	Octal Code	Description
		Memory Reference
FLD	105420	Load floating point accumulator with single precision number.
FLDD	105522	Load floating point accumulator with double precision number.
FST	105600	Store single point precision floating point accumulator in memory.
FSTD	105710	Store double precision floating point accumulator in memory.
FLT	105425	Reformat single precision integer and load into floating point accumulator.
FIX	105621	Reformat floating point accumulator and store in memory.
		Arithmetic Instructions
FAD	105410	Add single precision memory to floating point accumulator.
FADD	105503	Add double precision memory to floating point accumulator.
FSB	105450	Single precision floating point subtraction.
FSBD	105543	Double precision floating point subtraction.
FMU	105416	Single precision floating point multiply.
FMUD	105506	Double precision floating point multiply.
FDV	105401	Single precision floating point divide.
FDVD	105535	Double precision floating point divide.

V75 Instruction Set Option

The V75 instruction set option provides 27 instructions that permit programmer access to 8 general-purpose registers and operate on 8-, 16-, and 32-bit operands. These instructions are listed below.

Register-to-Memory Instructions

LD	Load
ST	Store
AD	Add
SB	Subtract

Byte Instructions

LBT	Load Byte
SBT	Store Byte

Jump-If Instructions

JZ	Jump if Register Zero
JNZ	Jump if Register Not Zero
JN	Jump if Register Negative
JP	Jump if Register Positive
JDZ	Jump if Double-Precision Register Zero
JDNZ	Jump if Double-Precision Register Not Zero

Double-Precision Instructions

DLD	Double Load
DST	Double Store
DADD	Double Add
DSUB	Double Subtract
DAN	Double AND
DOR	Double OR
DER	Double Exclusive OR

Register-to-Register Instructions

T	Transfer
ADR	Add Register
SBR	Subtract Register

Single Register Instructions

INC	Increment Register
DEC	Decrement Register
COM	Complement Register

Immediate Instructions

LDI	Load Immediate
ADI	Add Immediate

SECTION 6 - INPUT/OUTPUT SYSTEM

The Varian input/output (I/O) system allows the computer to interface with a large variety of peripheral devices. Interface circuitry to control a specific peripheral is contained on one or more controller cards that plug into a peripheral controller slot in the mainframe or expansion chassis. One controller can control one or more similar peripherals.

The I/O system comprises the following principal elements:

- Timing and control logic and internal bus interface in the processor
- Various peripheral controllers
- A party-line, time-shared I/O bus connecting the processor and controllers
- I/O options that enhance the I/O capabilities

The processor and control panel are described in sections 2 and 9. Information on the controllers for specific peripherals is given in section 7. I/O instructions are described in detail in section 13. This section concentrates on the role of the I/O bus and the interaction between the I/O bus and system peripheral controllers.

System Priority

A typical Varian 73 system priority structure is shown in figure 6-1. The following three priority levels are established for interfacing with the main memory. PMA has first priority (except for PF/R interrupt) and accesses memory directly. Next is the I/O bus to memory via DMA (BIC and interrupt PIM). Third, processor access to memory.

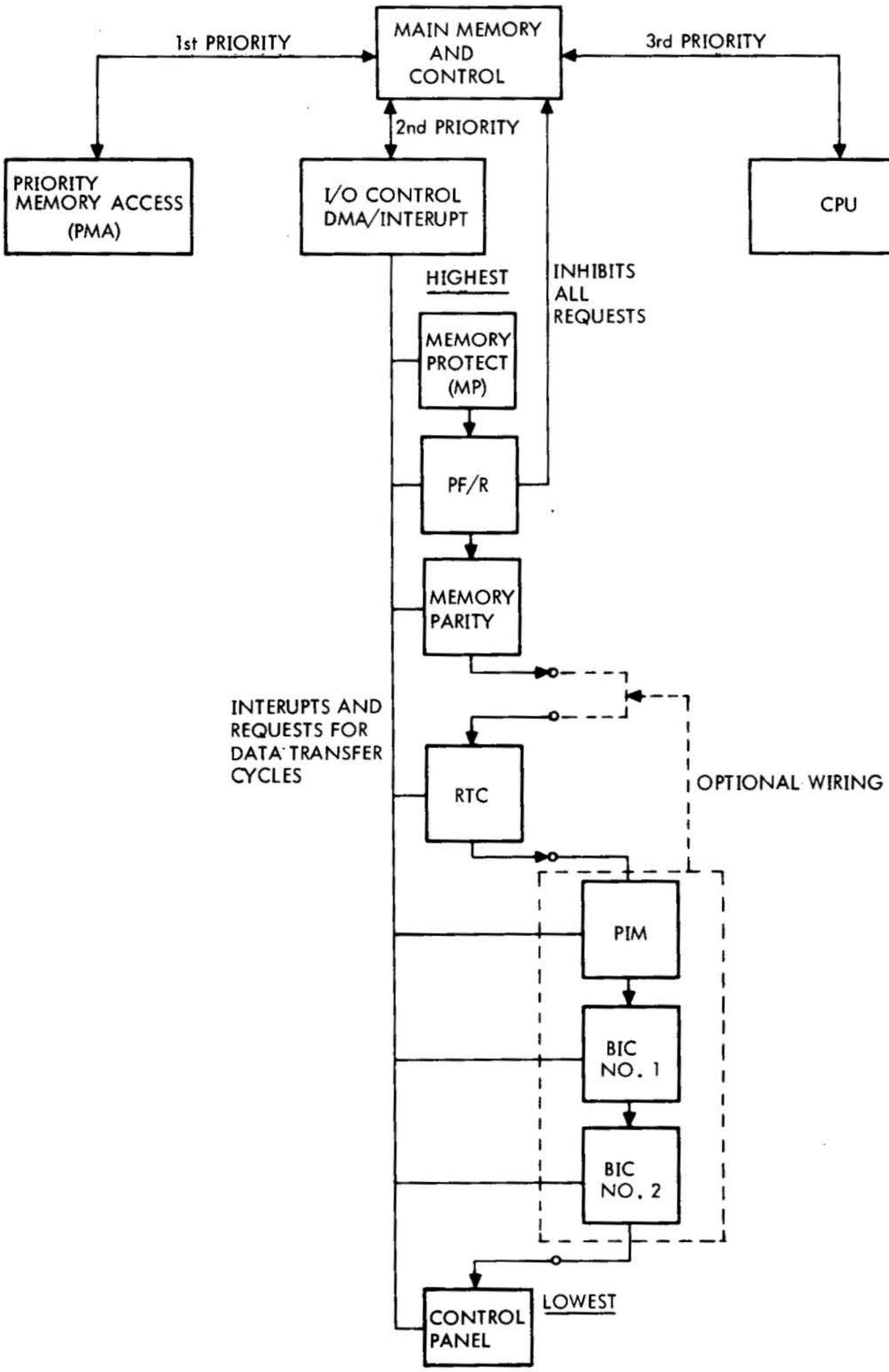
The highest DMA/interrupt priority begins with memory protection (MP) and is followed by PF/R, memory parity, real-time clock (RTC), priority interrupt module(s) (PIM), and the buffer interlace controller(s) (BIC) as required by the system application. The control panel interrupt priority follows all others.

Organization

The I/O system utilizes a bidirectional I/O bus, which allows one set of data and control lines to communicate with all system peripherals. The organization of the I/O system is illustrated in figure 6-2. All peripheral controllers and I/O options connect to the I/O bus.

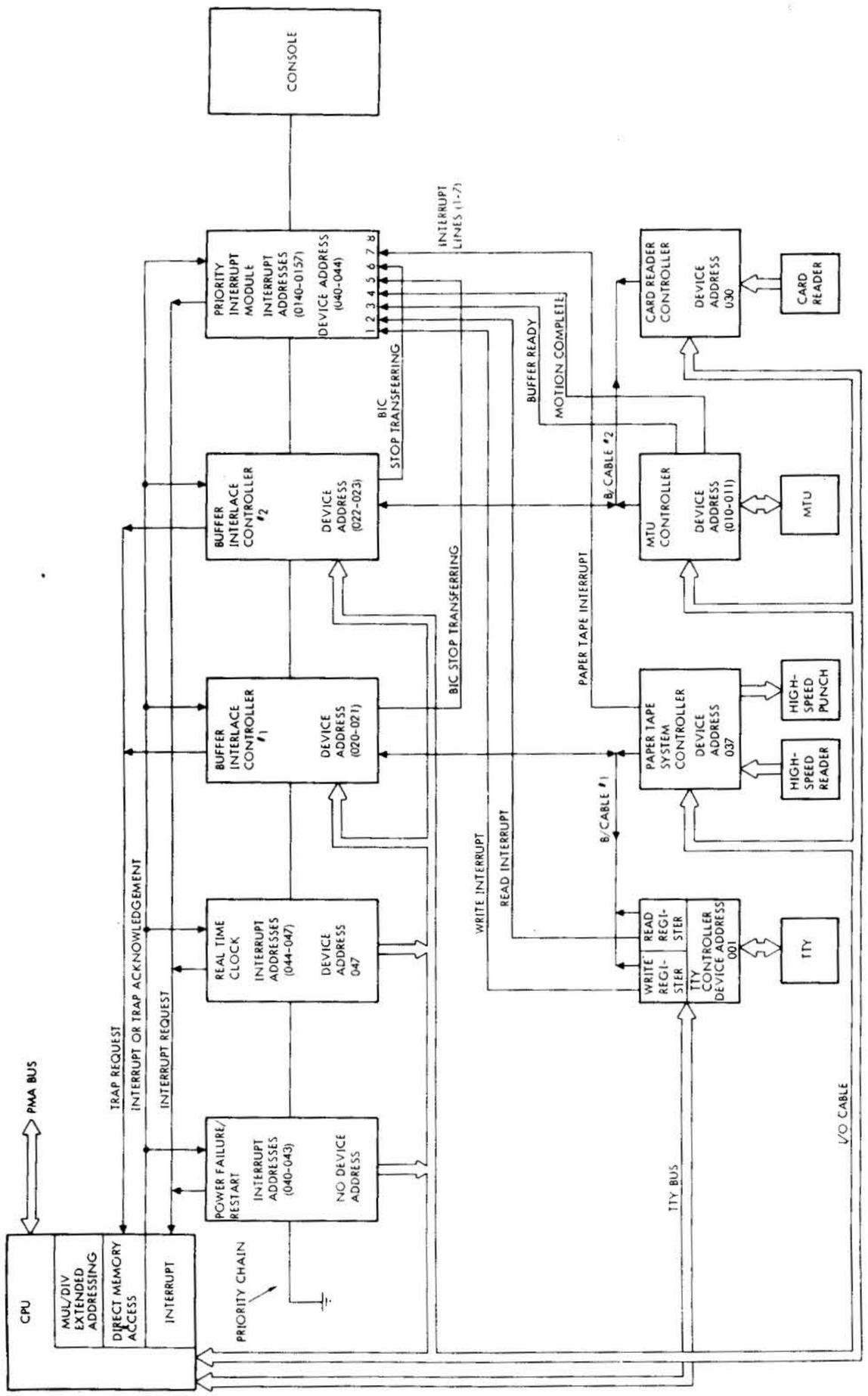
Note: The I/O bus contains two bus connections, one internal and one external. The option board, containing the first Teletype controller and various I/O options, is connected to the internal I/O bus. All other controllers are connected to the external bus.

The E bus, priority chain, BIC control, interrupt request, trap request, and acknowledgment lines shown in figure 6-3 are contained in the I/O bus; they are separated in the illustration for clarity. However, the interrupt lines to the PIM are separate entities, installed as needed during system installation or expansion.



VTII-1480

Figure 6-1. Typical System Priority



VTI3-0279

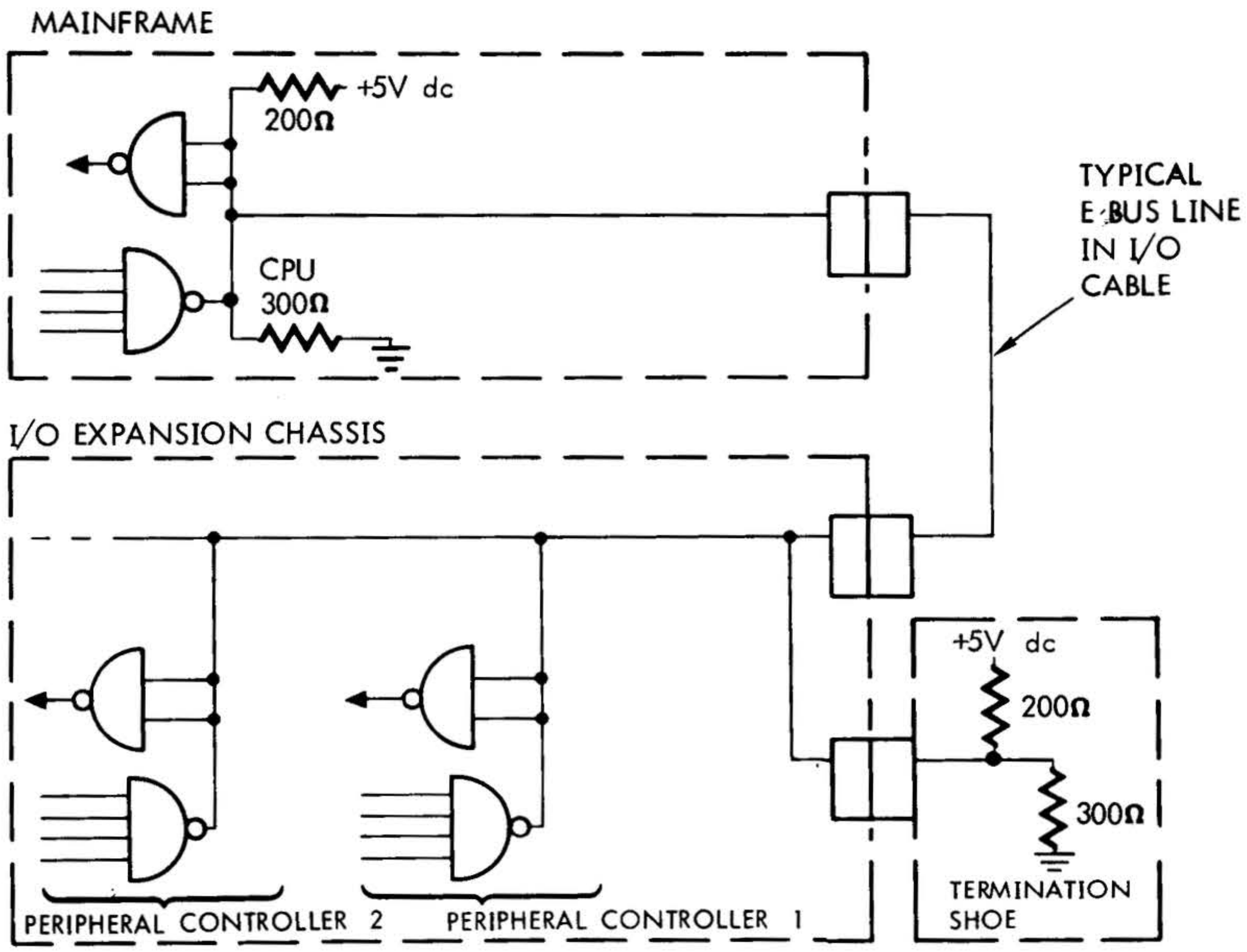
Figure 6-2. Typical Varian 73 I/O System Block Diagram

I/O Bus Structure

The Varian 73 system communicates directly with peripherals by transmitting an external control instruction and peripheral device address to the selected controller via the I/O bus. When a peripheral is ready to send or receive information, as indicated by its associated sense line, the computer requests the device to place a word of data, or to accept one placed by the computer, on the I/O bus. The I/O bus lines are described in the following subsections. System interconnection details are given in section 8.

E Bus

This 16-bit, parallel, bidirectional I/O channel (EB00– I through EB15– I) is used to transmit I/O instructions, device addresses, and data from the computer to the peripherals. In turn, the E bus is used by the peripherals to transmit data to the computer. Ten drivers and ten receivers can be connected to each line to service up to ten peripheral controllers. When more than ten controllers are included in the system, they are controlled through an I/O party-line expander card. An E bus signal is true when it is at 0V dc, and false at



VTI1-1481

Figure 6-3. Typical E Bus Line Configuration

+3V dc. Figure 6-3 shows a typical E bus configuration.

Control Lines

The following data and control signals are used during I/O instructions, interrupts, and in 620 compatible DMA. Figures 6-4 and 6-5 show typical control lines to and from the computer. The following signals are true at 0V dc and false at +3V dc.

FRYX-I: This signal is generated by the computer to indicate that an I/O instruction and a device address have been placed on the E bus. Each peripheral controller examines the device address, and, upon the true-to-false transition of FRYX-I, the addressed peripheral responds to the I/O instruction.

DRYX-I: This signal indicates that the computer has placed data on the E bus, or that it has accepted the data placed on the E bus by the peripheral. The transfer occurs upon the true-to-false transition of DRYX-I.

IUAX-I: This signal is generated by the computer to acknowledge the receipt of an interrupt, trap-in, or trap-out request. The interrupting or trapping peripheral controller can communicate an address to the computer and can receive data from or send data to the computer when IUAX-I is true. IUAX-I also inhibits device address decoding in all controllers during the address phase of an interrupt or trap operation to prevent the controllers from interpreting any part of the memory address as a device address.

SYRT-I: This signal is used to initialize all the peripheral controllers connected to the I/O bus. SYRT-I is true when the RESET switch on the control panel is pressed.

SERX-I: This signal is a controller response to a program sense instruction, during the execution of which the computer places a function code and a device address on the E bus. The addressed controller is instructed to indicate the status of a peripheral device action. If the status (sensed condition) is true, the controller responds by setting SERX-I true.

IUCX-I: This signal is an interrupt and/or trap synchronization clock from the computer that is disabled when IUAX-I is true. The true-to-false transition of IUCX-I sets the interrupt and/or request flip-flops in the appropriate peripheral controllers. IUCX-I is jumper selectable for one of two basic clock rates: 660 nanosecond or 990 nanosecond periods. The 990 nanosecond rate is standard.

IURX-I: An interrupting peripheral controllers (e.g. PIM) requests the computer to execute an instruction by setting this signal true. The address of the instruction is placed on the E bus when the computer acknowledges the interrupt request (IUAX-I).

IUJX-I: This signal is generated by the computer to inhibit all interrupts following a jump-and-mark instruction when that instruction is the result of an interrupt request.

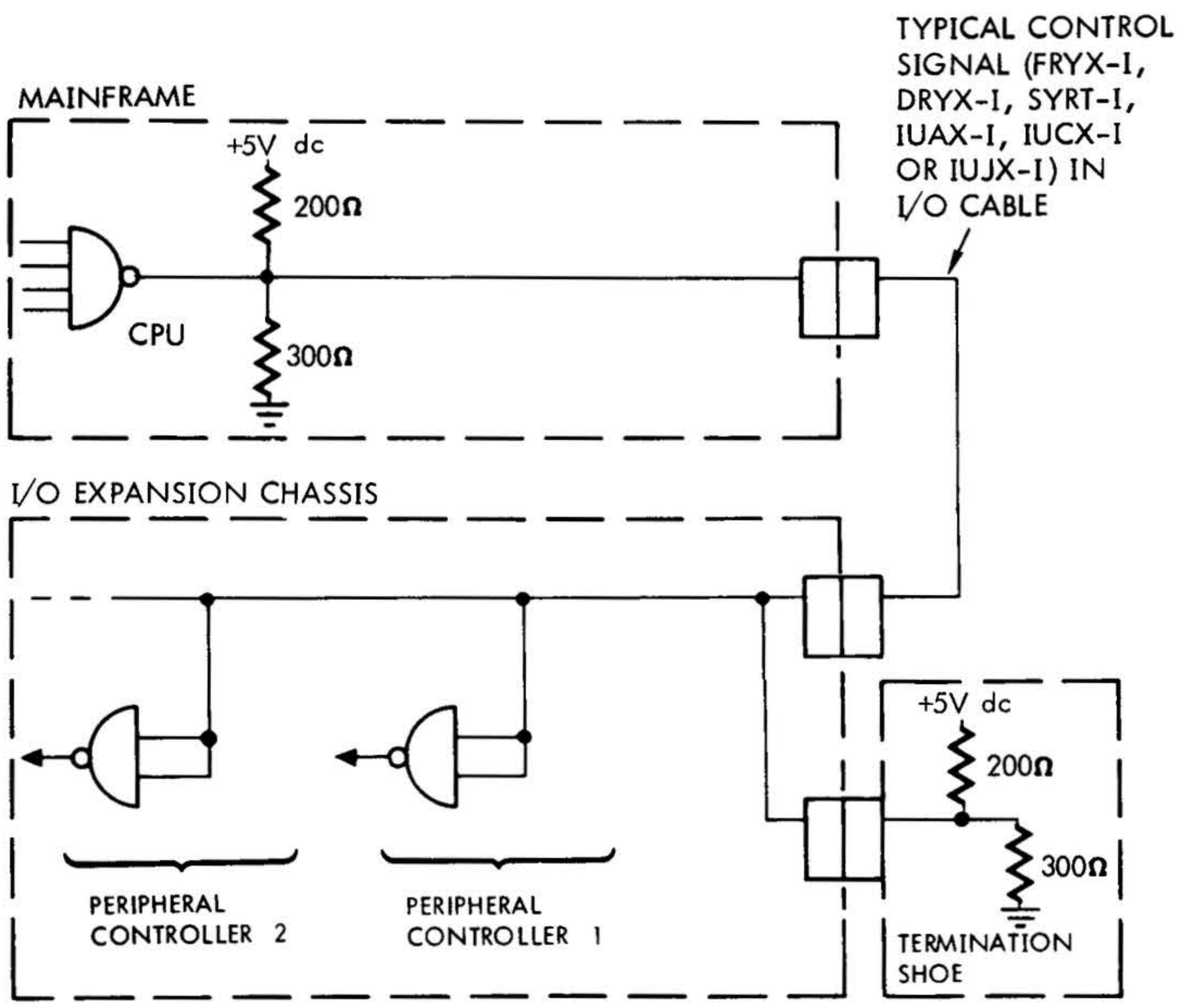
TPIX-I: A trapping peripheral controller (e.g. BIC) sets TPIX-I true to request the computer to input one word of data to the memory. The address is placed on the E

bus by the controller when the computer acknowledges the request.

TPOX-I: A trapping peripheral controller (e.g. BIC) sets TPOX-I true to request the computer to output one word of data from memory. The address is placed on the E bus by the controller when the computer acknowledges the request.

In addition to the E bus, the following signals are used by the DMA and are analogous to the corresponding signals described above: TPIF-I, TPOF-I, IUAF-I, IUCF-I, FRYF-I and DRYF-I.

Figures 6-4 and 6-5 show typical control lines to and from the computer.

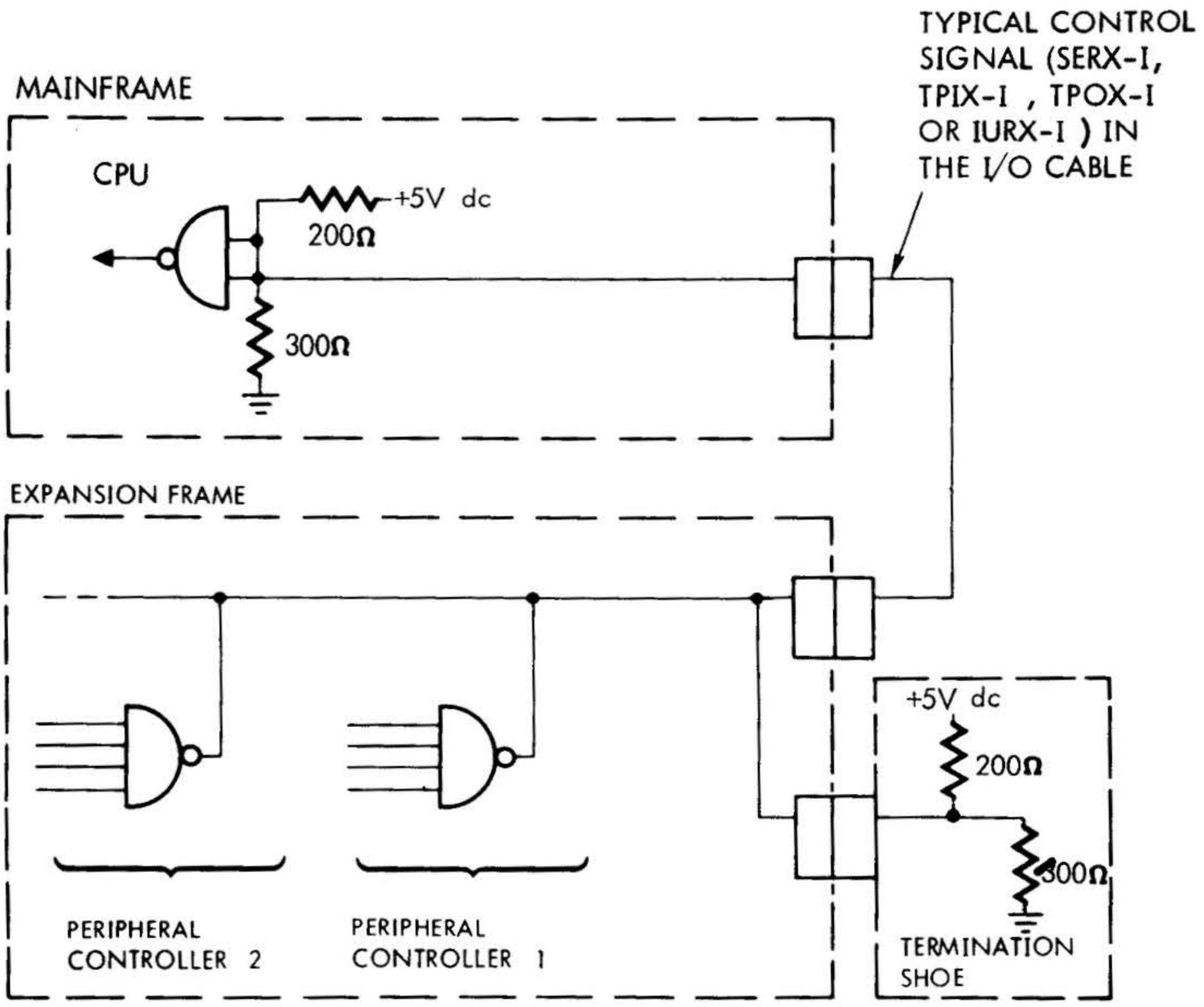


VTI1-1482

Figure 6-4. Typical Control Line from the Computer

Priority Lines (Interrupt and 620 Compatible DMA)

PR1X-I through PR10X-I are used to establish the priority of system interrupts by connecting devices in a priority chain. The devices that can be included in this priority chain (in order from high to low priority) are: MP, PF/R, memory parity, RTC, PIM, BIC, and interrupt (INT) switch on the control panel. These devices can be switched on in any order. Peripheral controllers cannot, of themselves, generate interrupt requests. They do so only through interaction with the PIM. System interrupts are discussed in greater detail later in this section.



VTII-1483

Figure 6-5. Typical Control Line to the Computer

Priority Lines (DMA)

PRMA+I through PRMC+I are used to establish the priority of DMA controllers. They connect controllers in a parallel priority scheme. The output of each higher priority device is connected directly to a priority input of each of the lower priority devices attached to the DMA (figure 6-6).

BIC Control Lines

The seven BIC control lines DCEX-B, DESX-B, TAKX-B, CDCX-B, BCDX-B, TRQX-B, and TROX-B are used for communication between a BIC and the peripheral controllers it monitors.

I/O Operations

In the Varian 73 system, information transfers can occur under the control of a stored program, they can be interrupt-initiated, or they can occur on a cycle-stealing (trapping) basis. Specific details of these operations are contained in the Varian 70 series processor Maintenance Manual (document 98 A 9906 02x). The following paragraphs briefly outline the capabilities of the system.

Program-Controlled I/O

The I/O system provides four types of I/O operations under program control:

a. **External Control.** An external control code, specifying a specific peripheral function and a device address, is transmitted from the computer to a peripheral controller.

b. **Program Sense.** The status of a selected peripheral controller sense line is interrogated by the computer.

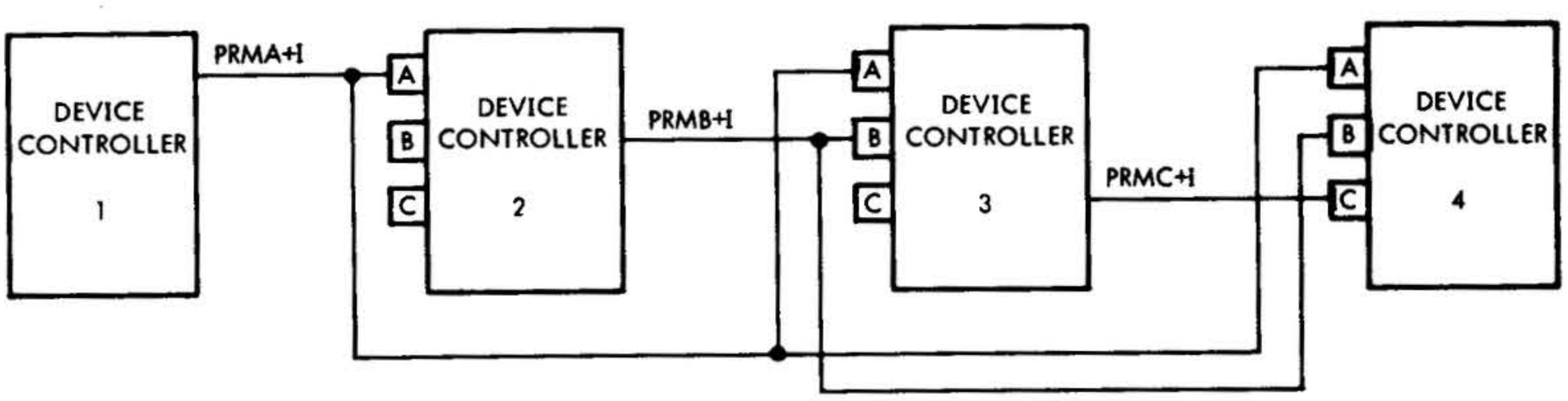
c. **Input Data Transfer.** One word of data is transferred from a peripheral controller to the A register, B register, or a location in memory.

d. **Output Data Transfer.** One word of data is transferred to a peripheral controller from the A register, B register, or a location in memory.

The instructions implementing these operations are described in section 13.

Under program control, the I/O system communicates directly with all peripherals.

The computer can initiate peripheral operations by transmitting an external control



VTI1-3229

Figure 6-6. DMA Priority Block Diagram

function code and a proper device address to the selected controller via the I/O bus. The computer can determine when a peripheral is ready to send or receive information by interrogating its associated sense line. A peripheral can be requested to place a word of data on the I/O bus during a computer input transfer, or to accept a word of data placed on the bus by the computer during an output transfer.

Table 6-1 summarizes how the E bus and control lines are connected (routed) to the controller. Figures 6-7 through 6-10 show the timing for the program controlled operations.

Only part of an I/O instruction is transmitted intact over the E bus. The device address (bits 0 through 5) and function code (bits 6 through 8) of the instruction are transmitted unchanged. Bits 9 through 15, however, are decoded in the processor first so that one of the lines EB11-I through EB15-I is set to produce specified operation.

Interrupt-Initiated I/O

The Varian 73 I/O system includes an interrupt capability by which certain devices and options, on a priority basis, can request the computer to execute an instruction (or a series of instructions) independent of the program in progress. During an interrupt, the computer is directed to a memory address specified by the interrupting device and executes the instruction at that address. Normally, the instruction at the interrupt address is a jump-and-mark instruction that results in the processing of an I/O service subroutine. The computer returns to the original program through an appropriate jump instruction at the conclusion of the interrupt subroutine.

Standard Varian peripheral controllers normally are not capable of generating an interrupt because they cannot provide the necessary memory address. The PIM supplies this addressing capability; thus implementing an **external interrupt system** within the computer interrupt system. A peripheral controller connected to a PIM directs an interrupt request to the PIM, which in turn sets IURX-I true. The PIM then waits for the computer to acknowledge the request (IUAX-I true), and then places the appropriate interrupt address on the I/O bus.

Up to eight interrupt levels can be serviced by one PIM. PIM priority logic establishes, on a hard-wired basis, the order in which interrupt requests are serviced. Normally the priority assignment is wired at the factory before equipment delivery. The user can, however, change ths system to suit specific system requirements.

Figure 6-11 shows a typical timing sequence for an interrupt.

Standard Varian interrupt addresses are:

Address	Device and Function
020,021	MP halt error
022,023	MP I/O error
024,025	MP write error
026,027	MP jump error
030,031	MP overflow error
040,041	Power failure
042,043	Power restart
044,045	RTC interval
046,047	RTC overflow
100-117	PIM, first modules
120-177	PIM, remaining modules

Note: Parity interrupt address is configured by jumpers.

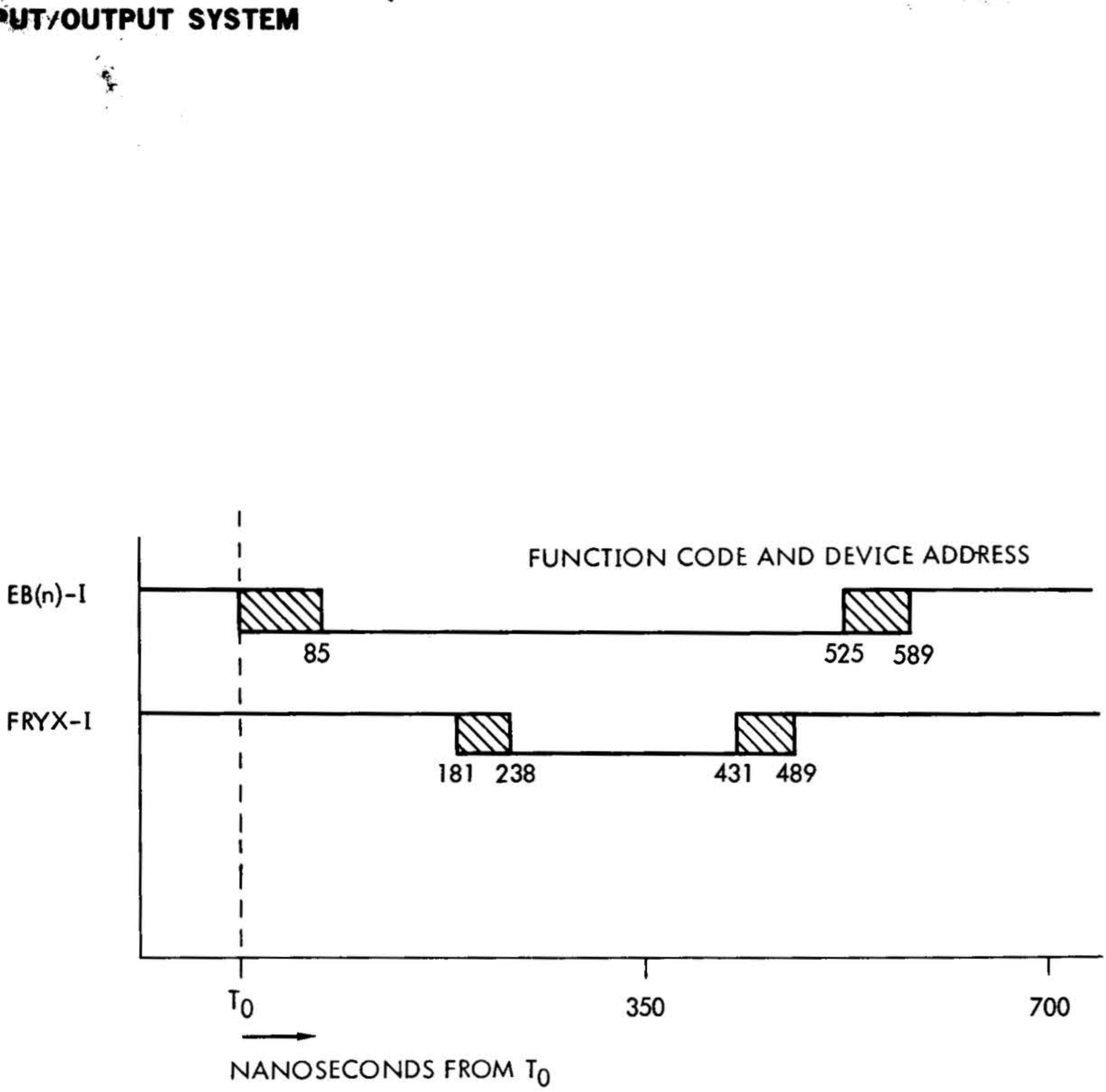
Table 6-1. E Bus and I/O Control Signals

OPERATION >	External Control	Sense	Data Transfer		Trapping Sequence		Interrupt Sequence
CONTROL LINE >	FRYX-I* (Phase 1)	FRYX-I* SERX-I* (Phase 1)	FRYX-I* (Phase 1)	DRYX-I (Phase 2)	TPOF-I, TPOX-I IUAF-I, IUAX-I, FRYF-I, FRYX-I, (Phase 1)	or TPIF-I, TPIX-I IUAF-I, IUAX-I, DRYF-I, DRYX-I (Phase 2)	IURX-I IUAX-I (Phase 1)
EB00-I EB01-I EB02-I EB03-I EB04-I EB05-I	Device address	Device address	Device address	Data	Address	Data	Pairs of signals used for specific interrupts
EB06-I EB07-I EB08-I	Function code	Function code	Not used				

OPERATION >	External Control	Sense	Data Transfer		Trapping Sequence		Interrupt Sequence
CONTROL LINE >	FRYX-I* (Phase 1)	FRYX-I* SERX-I* (Phase 1)	FRYX-I* (Phase 1)	DRYX-I (Phase 2)	TPOF-I, TPOX-I IUAF-I, IUAX-I, FRYF-I, FRYX-I, (Phase 1)	or TPIF-I, TPIX-I IUAF-I, IUAX-I, DRYF-I, DRYX-I (Phase 2)	IURX-I IUAX-I (Phase 1)
EB09-I EB10-I	Not used	Not used	Not used				
EB11-I	External control command	Zero	Zeros				
EB12-I	Zeros	Sense command					Pairs of signals used for specific interrupts
EB13-I		Zeros	Data in	Data	Address	Data	
EB14-I			Data out				
EB15-I	See note 3		Zero				

NOTES:
1. Phase 1 is device or memory selection.
2. Phase 2 is the data transmission.
3. For extended external control, control and data lines are the same as external control except EB11-I is zero and EB15-I is one.

* IUAX interlock; used in address decoding.



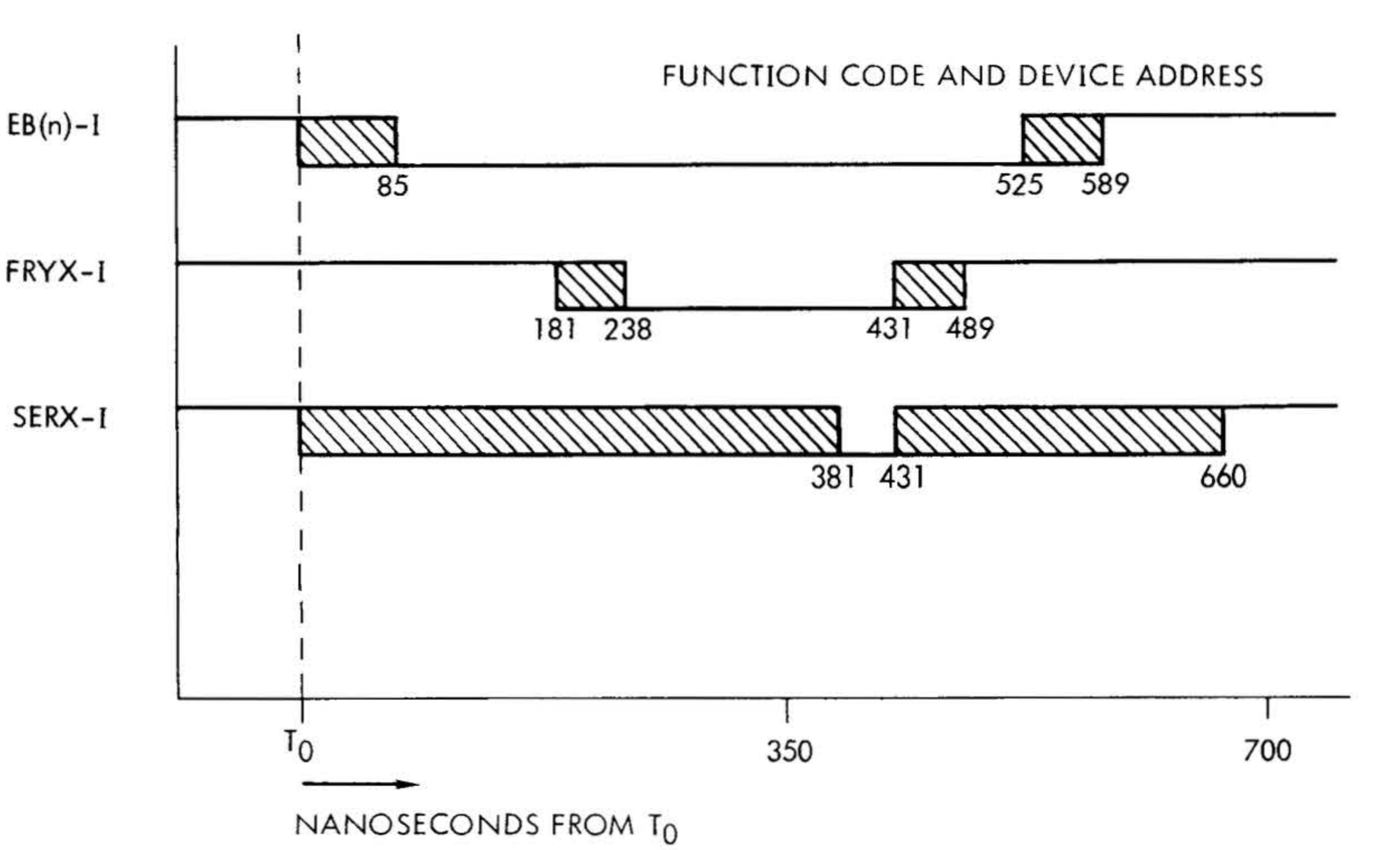
T_0 is the start of the execute phase of the external control instruction.

Logic levels: true = 0V dc,
false = +3V dc.

▨ = time when signal is settling.

VTII-1506

Figure 6-7. External Control Timing



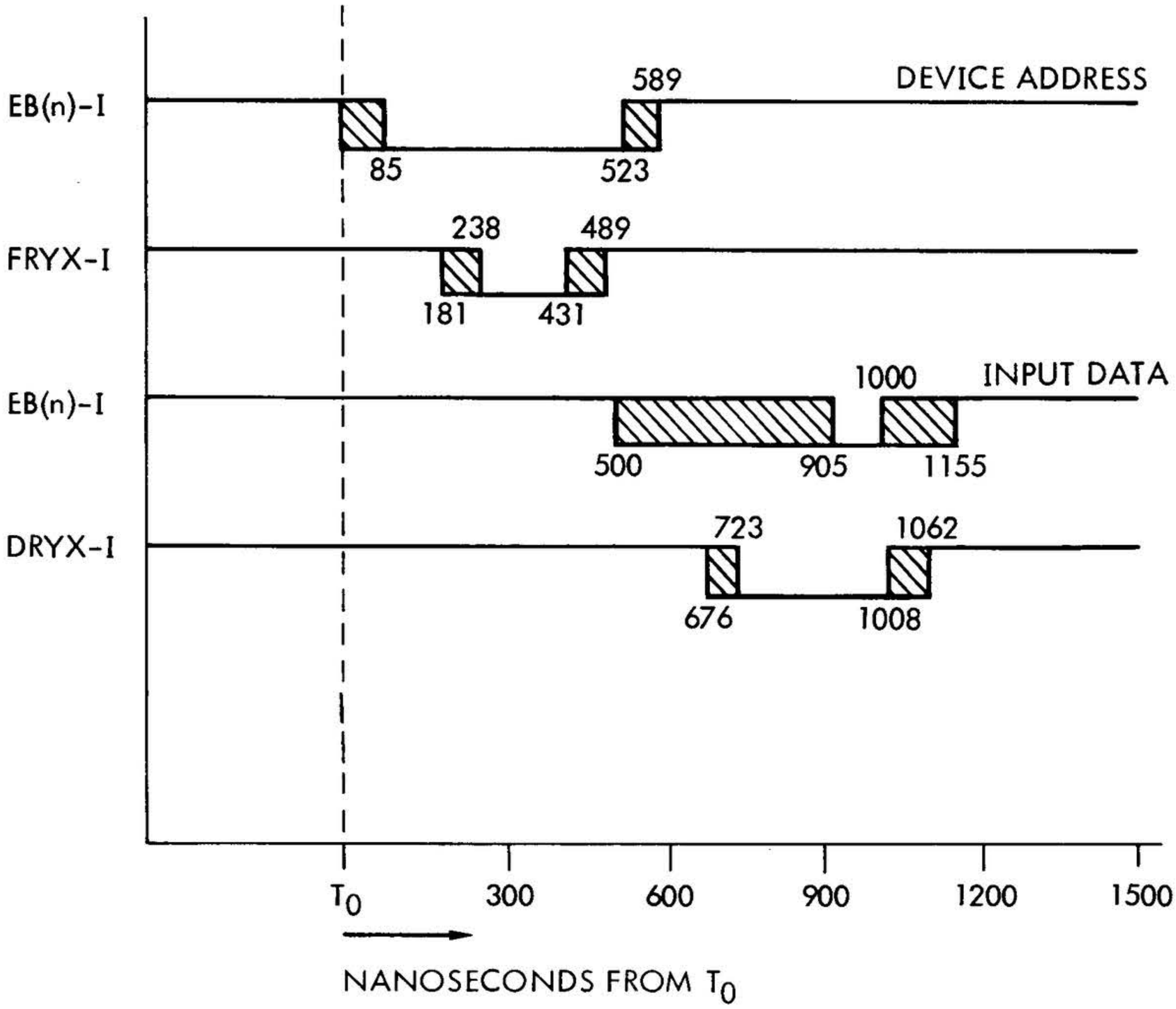
T_0 is the start of the execute phase of the sense instruction.

Logic levels: true = 0V dc,
false = +3V dc.

▨ = time when signal is settling.

VTI1-1507

Figure 6-8. Sense Response Timing



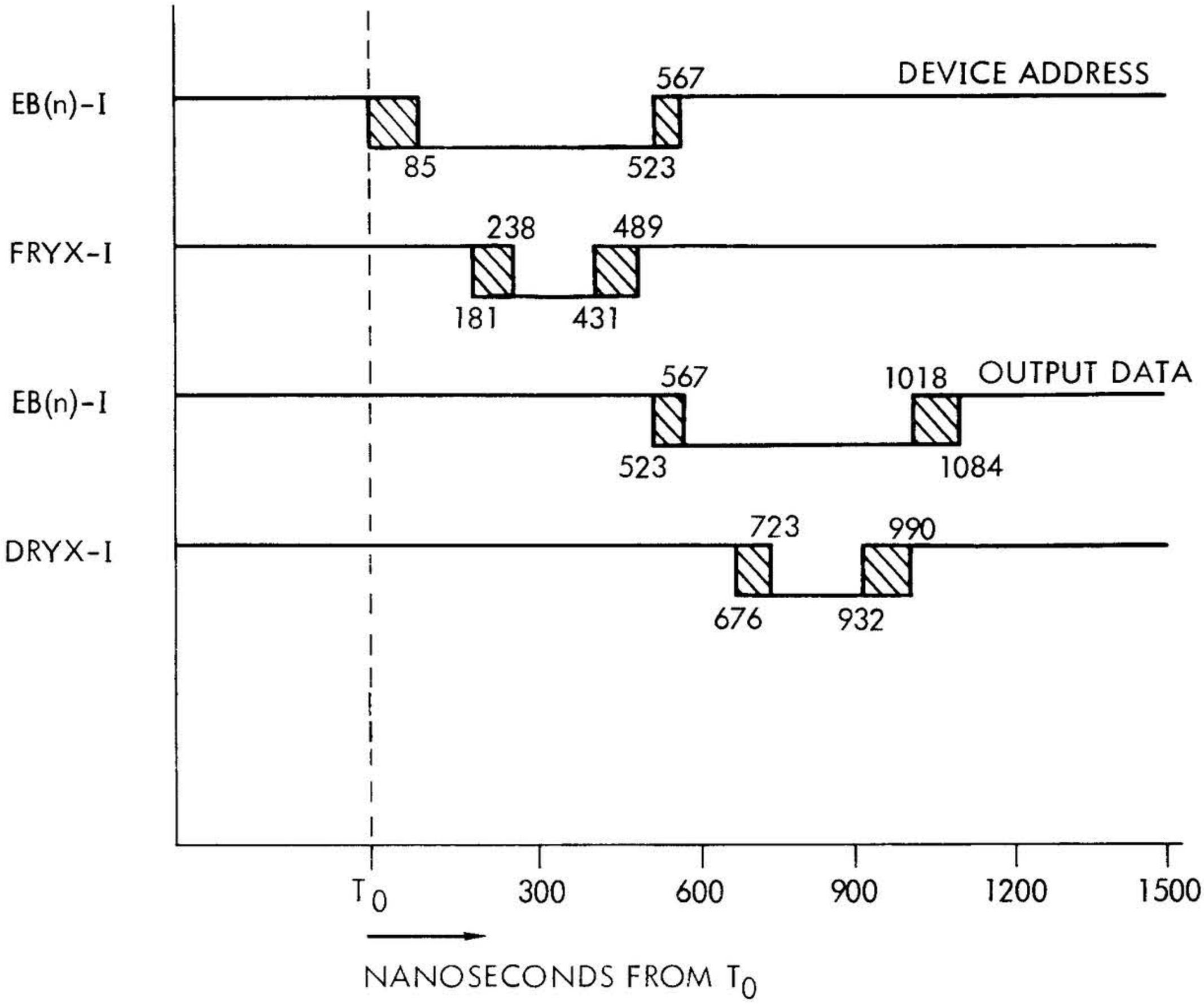
T_0 is the start of the execute phase of the data transfer in instruction.

Logic levels: true = 0V dc,
false = +3V dc.

= time when signal is settling.

VTI1-1508

Figure 6-9. Data Transfer-In Timing



T_0 is the start of the execute phase of the data transfer out instruction.

Logic levels: true = 0V dc,
false = +3V dc.

= time when signal is settling.

VTI1-1509

Figure 6-10. Data Transfer-Out Timing

DMA (620 Compatible)

Cycle-stealing I/O operations are implemented by the addition of one or more (up to four) BICs. Cycle-stealing I/O is combined with the features of program-controlled and interrupt-initiated I/O. This mode of operation allows peripherals on the I/O bus to transfer data to or from memory while temporarily halting the processing of the stored program. This process is also referred to as "trapping."

Trap requests differ from interrupt requests in two ways:

a. Interrupts direct the computer to the address of a subroutine, whereas the trapping requests require the computer to transfer data to or from memory. The data to be transferred is placed on the E bus after the memory address has been transmitted.

b. The subroutine specified by an interrupt returns the computer to the main program, whereas trapping operations halt the program execution when both program and I/O request memory usage. This method allows data to be transferred between memory and peripherals at rates up to 361,800 words per second.

Cycle-stealing traps do not disturb the contents of the operation registers (A, B, X, and P), thus freeing the CPU to perform other operations during data transfers.

Trapping operations are initiated by the stored program. The program signals the controller (via an I/O command) to request the BIC to issue a trap request. The BIC service subroutine establishes the initial and final addresses for the transfer, identifies the peripheral controller, and initializes both the selected controller and the BIC.

When the BIC receives a trap request from a controller (TRQX-B) it issues a trap request (TPIX-I or TPOX-I) to the processor. When the processor sends an acknowledgment (IUAX-I) to the BIC, the BIC places the initial memory address on the E bus and increments the initial address buffer by one. When a data word has been transferred, the controller again sends a trap request to the BIC. The sequence is repeated until the initial buffer contents equals the final address. The processor utilizes the memory cycles between the trap memory cycles to continue processing the stored program.

If trap requests are present continuously for a period of time and the processor executes an I/O instruction, the processor will stop until the trap requests are no longer continuously present.

Figure 6-12 shows the input and output timing for 620 compatible DMA operations. Figure 6-13 shows the 620 compatible DMA and interrupt request timing for a 660 and 990 nanosecond sample rate.

DMA

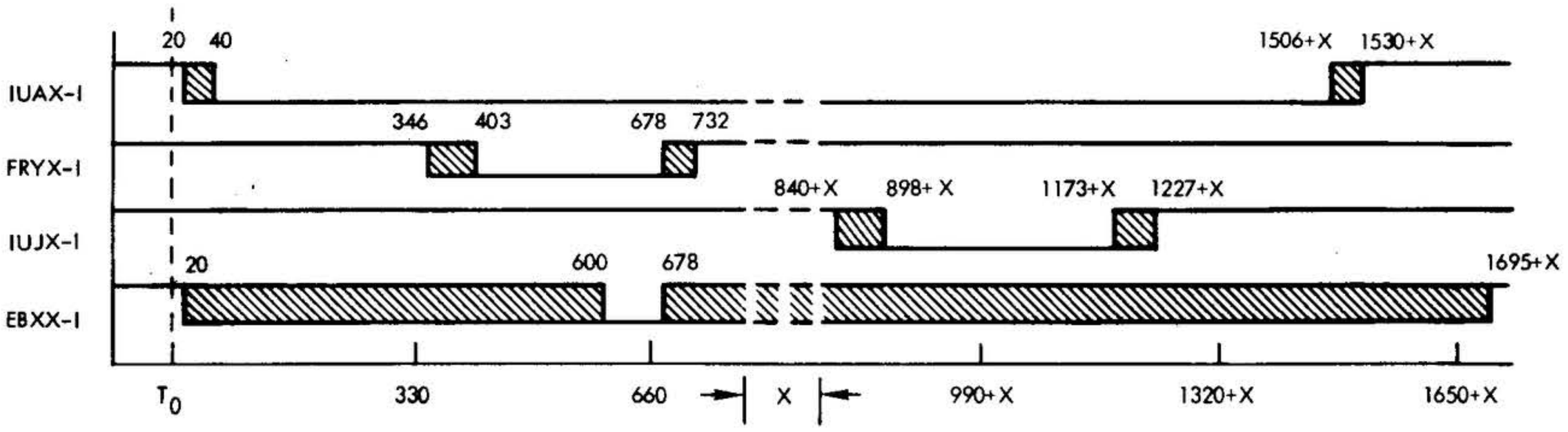
The DMA functions in a similar manner to the 620 compatible DMA except a faster interrupt clock and I/O timing is used. A separate set of function ready, data ready, interrupt acknowledge, interrupt clock, trap in, and trap out lines are used. Figure 6-14 shows the high-speed DMA input and output timing.

Device Addresses

Standard device addresses assigned to options and peripherals used in I/O system operations are listed in table 6-2, grouped according to their function (i.e., class).

Table 6-2. Standard Device Addresses

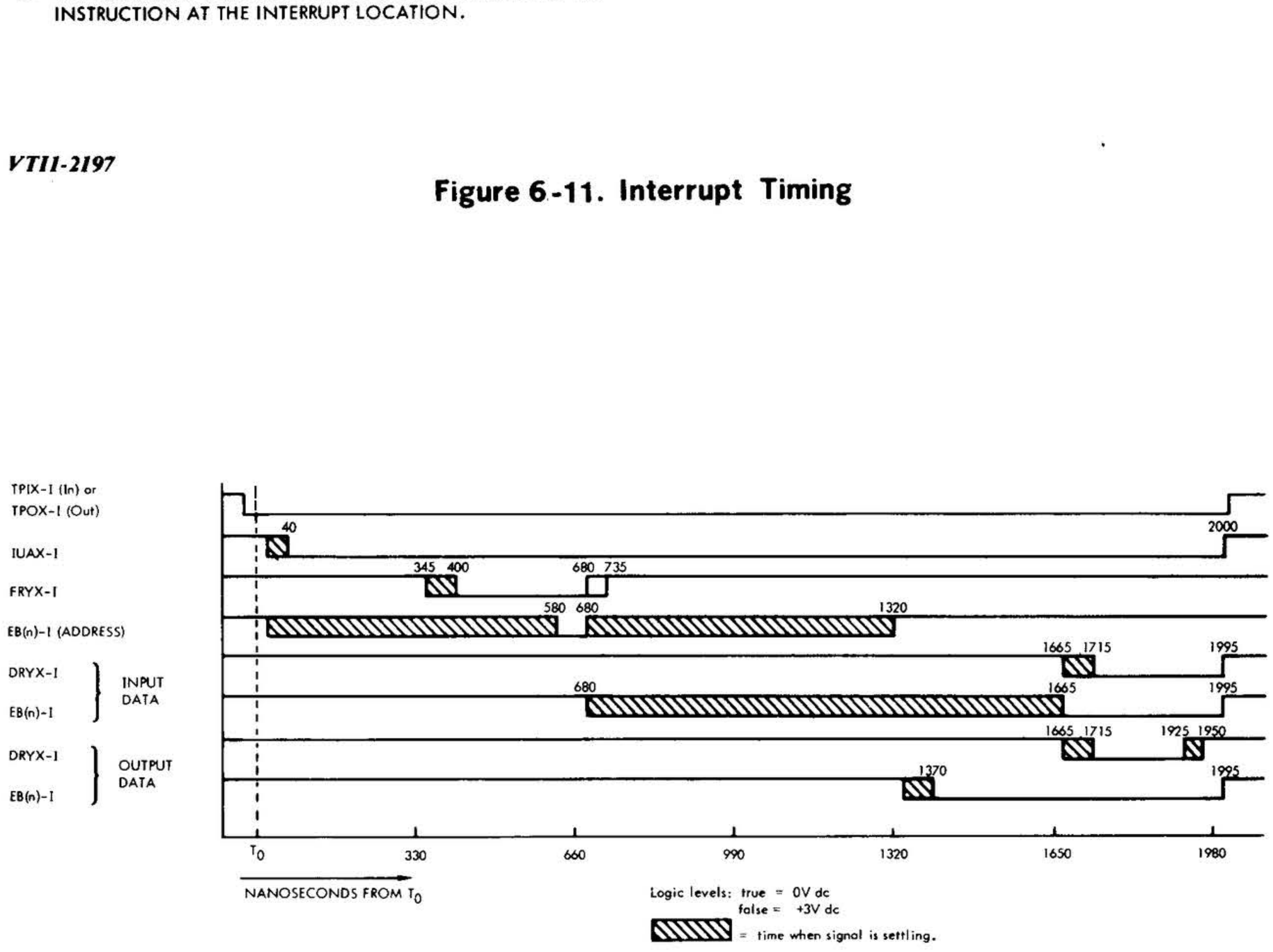
Class Code	Addresses	Option or Peripheral
00-07	01-07	Teletype (620-06, -07, -08), or CRT device
010-017	010-013	Magnetic tape unit (620-30, -31)
	014	Fixed-head rotating memory (620-38, -42 through -49)
	015	Movable-head rotating memory (620-35)
	016,017	Movable-head rotating memory (620-37, -36)
020-027	020,021	First BIC (620-20)
	022,023	Second BIC
	024,025	Third BIC
	026,027	Fourth BIC
030-037	030	Card reader (620-25)
	031	Card punch (620-27)
	032	Digital plotter (620-72)
	033	Electrostatic plotter
	034	Second paper tape system
	035,036	Line printer (620-77, or 620-74)
	037	First paper tape system (620-53, -55, -55A, -51, -51A)
040-047	040-043	PIM (620-16)
	044	All PIM enable/disable
	045	MP
	047	RTC
050-047	050-053	Special applications, and Digital-to-analog converter (620-870 through -875)
	054-057	Analog system (620-85A, -850, -851)
060-067	060-067	Digital I/O controller (620-81), or Buffered I/O controller (620-80)
070-077	070-073	Data communications system (620-60, -61, -65, -66, -68)
	074-076	Relay I/O controller (620-83), or Special applications
	077	Varian 73 console



X = THE PROCESSOR TIME REQUIRED TO FETCH AND DECODE THE INSTRUCTION AT THE INTERRUPT LOCATION.

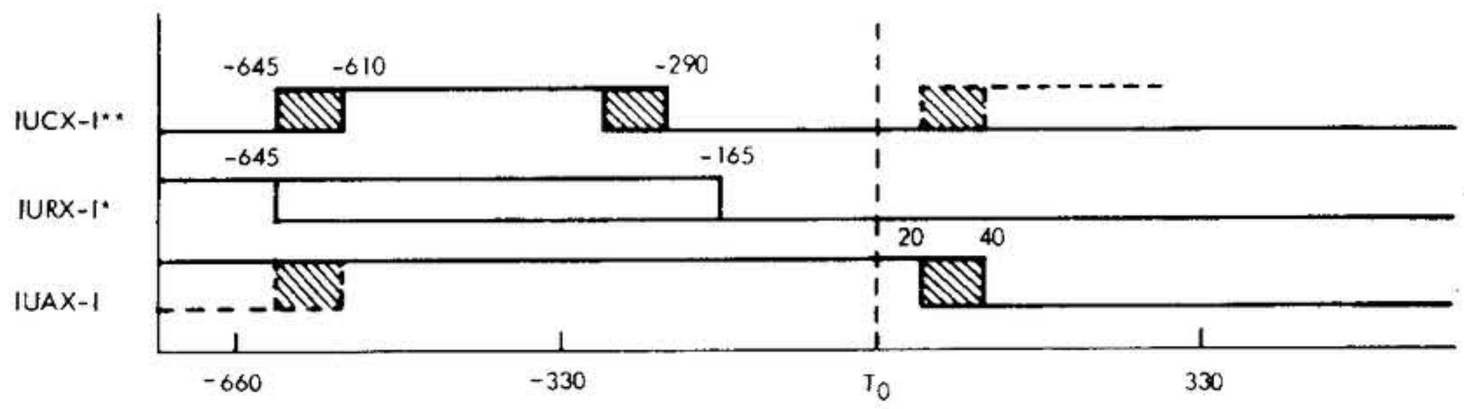
VTII-2197

Figure 6-11. Interrupt Timing



VTII-2192

Figure 6-12. 620 Compatible DMA Input and Output Timing

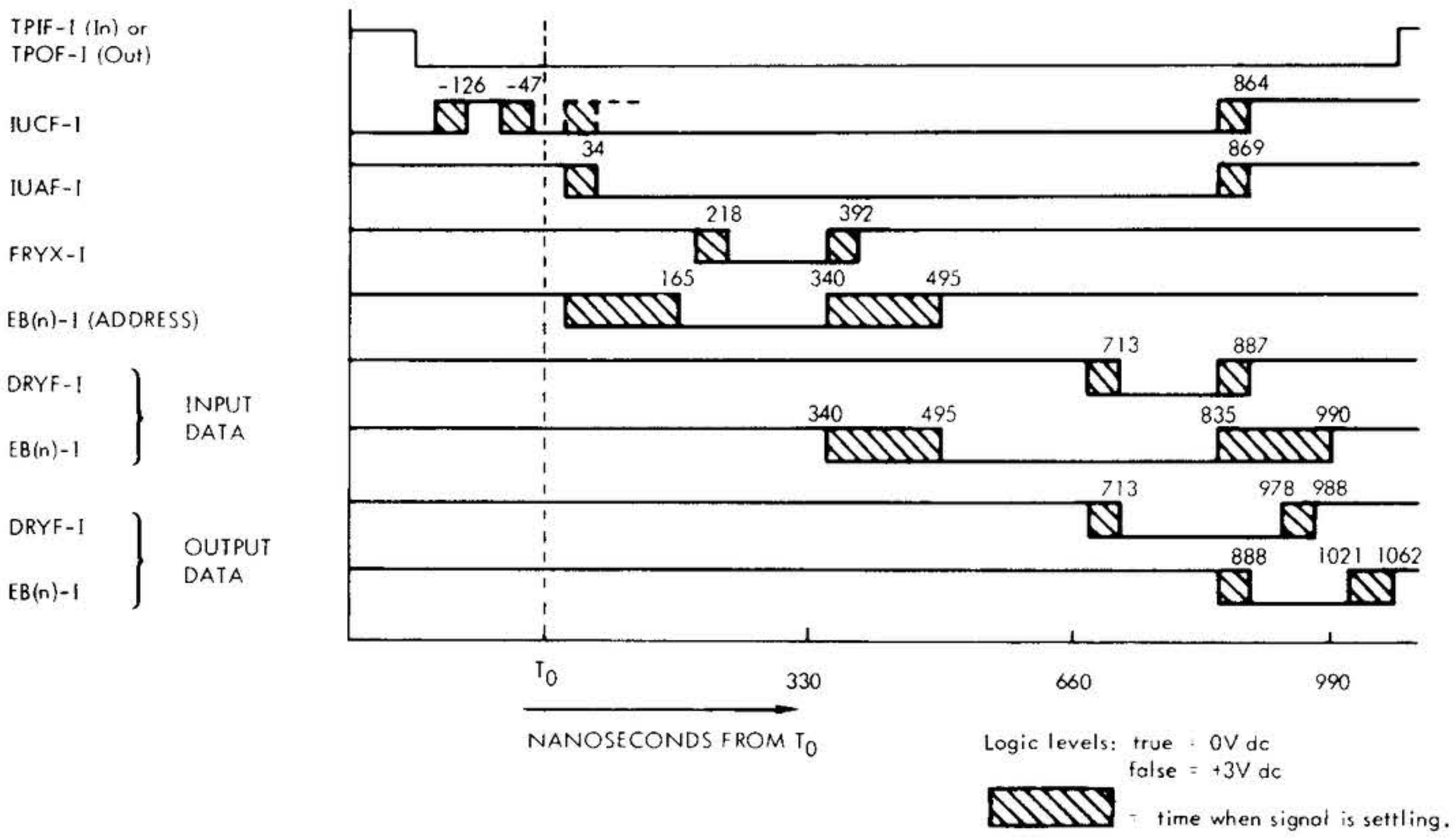


* THE ABOVE TIMING ALSO APPLIES TO DMA REQUEST

** THE INTERRUPT CLOCK PERIOD (TPOK-I, TPIK-I) IS JUMPER SELECTABLE VIA CLIPS, FOR 660 NANOSECONDS OR 990 NANOSECONDS. LONGER LENGTH PRIORITY STRINGS MAY REQUIRE A SLAVE INTERRUPT SAMPLE RATE (990 NANOSECONDS)

VTII-3162

Figure 6-13. 620 Compatible DMA and Interrupt Request Timing



VTII-2194

Figure 6-14 DMA Input and Output Timing

SECTION 7 - PERIPHERALS AND I/O INTERFACES

A complete line of peripherals and I/O interfaces is available for use with the Varian 73. This section presents descriptions of a sampling of the peripheral device/controller combinations offered.

- Teletypes
- Paper Tape System
- Magnetic Tape Equipment
- Disc Memories
- Line Printers
- Punched Card Equipment
- Oscilloscope Display Units
- Digital Plotters
- Analog I/O Systems
- Data Communications Equipment
- Buffered I/O Controllers
- Relay I/O Controllers
- Digital I/O Controllers
- Universal Controller
- Disc Memory Systems

The circuits of the Varian 73 peripheral controllers and I/O interfaces are contained on 7-3/4-by-12-inch (19.7 by 30.3 cm) etched-circuit cards that can be installed in any peripheral controller slots in the expansion chassis.

This section briefly discusses peripheral device and controller capabilities. Refer to the applicable Varian Data Machines technical manual for details of peripheral specifications, installation, operation, and maintenance.

General controller specifications are listed in table 7-1. Specific controller specifications are listed in the tables for specific models.

Teletypes

The peripheral controller for the first Teletype in the Varian 73 system is provided on the option board. Refer to sections 4 and 8 for interconnection details.

If more than one Teletype is required, peripheral controllers (Models 70-6100,-6102, and -6104) can be installed in any I/O slot in the mainframe or expansion chassis. The additional Teletypes also require the universal asynchronous controller (Model 70-5602). As many as eight Teletypes can be included in one 73 system.

The factory-modified Teletype unit controlled by the Teletype controller can be a Teletype Model ASR-33, ASR-35, or KSR-35. The ASR models include paper-tape reader and punch capabilities; the KSR model uses only keyboard-entered instructions and data.

Specifications of the Teletype controller are given in table 7-2. Table 7-3 lists the Teletype instructions. Refer also to the Teletype section of the Option Board Manual (98 A 9906 05x).

Table 7-1. General Controller Specifications

Parameter	Description
Dimensions	Each wire-wrap circuit board is 7.75 by 12 inches (19.7 by 30.3 cm) and requires 3 slots. Each etched circuit board is 7.75-by-12-inches (19.7 by 30.3 cm) and requires 1 slot
Interconnection	Each wire-wrap and etched circuit board contains one 122-terminal connector (P1) and two 44-terminal connectors (J1 and J2). The boards interconnect with the processor via connector P1.
Logic Levels (internal)	High = +2.4 to +5.0V dc Low = 0 to +0.4V dc
Logic Levels (I/O bus)	High = +2.8 to +3.6V dc Low = 0 to +0.5V dc
Operational Environment	0 to 50 degrees C (32 to 122 degrees F); 0 to 90 percent relative humidity without condensation

Table 7-2. Teletype Controller Specifications

Parameter	Description
Organization	Contains input and output registers, timing control circuitry for simultaneous two-way communication, and processor/Teletype interface logic
Control Capability	One factory-modified Teletype Model ASR-33, ASR-35, or KSR-35, including cable
I/O Capability	One external control, eight transfer, and two program sense instructions

Table 7-2. Teletype Controller Specifications *(continued)*

Parameter	Description
Operation Modes	Input: from keyboard or paper-tape reader Output: to Teletype printer or paper-tape punch
Interrupt Capability	Ready to write and ready to read interrupt available to PIM
Standard Device Address	01 through 07
Size	First controller on option board Second and subsequent controllers each on printed circuit boards
Interconnection	Interfaces with the I/O bus and Teletype unit via option board connectors
Operational Environment	0 to 50 degrees C; 0 to 90 percent relative humidity without condensation

Table 7-3. Teletype Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0401	100401	Initialize Teletype Controller
	Data Transfer	
OAR 01	103101	Transfer A register contents to input register
OBR 01	103201	Transfer B register contents to input register
OME 01	103001	Transfer memory contents to input register
INA 01	102101	Transfer output register contents to A register
INB 01	102201	Transfer output register contents to B register
IME 01	102001	Transfer output register contents to memory

Table 7-3. Teletype Controller Instructions *(continued)*

Mnemonic	Octal Code	Description
CIA 01	102501	Transfer output register contents to cleared A register
CIB 01	102601	Transfer output register contents to cleared B register
Program Sense		
SEN 0101	101101	Ready to write
SEN 0201	101201	Ready to read

Teletype Commands

Function	Symbol	Octal Code	Type As:
Enable printer	SOM	201	CONTROL and A
Suppress printer	EOT	204	CONTROL and D
Reader on	XON	221	CONTROL and Q
Punch on	TAPE	222	CONTROL and R
Reader off	XOFF	223	CONTROL and S
Punch off	TAPE OFF	224	CONTROL and T

High-Speed Paper Tape Equipment

The high-speed paper tape system (Model 70-6320) comprises a controller, perforator, and reader. A paper tape spooler is also available for use with the reader.

The controller card contains a data register that buffers the data words being transferred, a decoder section that interprets instructions received from the computer, a timing and control section that synchronizes operation of the peripheral equipment with the computer, and necessary interface hardware.

The controller can transfer data from the reader to the computer. It can also transfer data to the perforator from the computer. It can be used to reproduce paper tapes. The controller can transfer data into the computer in a continuous read mode, which places the reader in continuous slew until an instruction to stop is received, or it can operate in a step read mode, requiring a new instruction from the computer for each transmitted data word.

Computer control of the paper tape system is accomplished through the I/O bus. The controller can also be operated under the direction of the BIC.

Each controller is capable of operating one perforator and one reader on a time-shared basis.

Specifications of the paper tape system controller are given in table 7-4, and table 7-5 lists the paper tape system instructions. Refer also to the paper tape controller manual (document number 98 A 9902 15x).

Table 7-4. High-Speed Paper Tape Controller Specifications

Parameter	Description
Organization	Contains a timing and control section, instruction decoder, and an eight-bit data buffer register
Control Capability	One reader and one perforator, operated on a time-shared basis
I/O Capability	Five external control, eight transfer, and one program sense instructions
Operational Modes	Continuous read: 300 characters per second Step read: 1 to 300 characters per second Punch: 1 to 75 characters per second
Standard Device Address	037
Size	One etched-circuit card
Input Power	+5V dc at 540 milliamperes

Table 7-5. High Speed Paper Tape Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 037	100037	Connect punch to BIC
EXC 0437	100437	Stop reader and initialize controller
EXC 0537	100537	Start reader
EXC 0637	100637	Enable punch buffer
EXC 0737	100737	Read one character
	Data Transfer	
OAR 037	103137	Transfer A register contents to data buffer register
OBR 037	103237	Transfer B register contents to data buffer register
OME 037	103037	Transfer memory contents to data buffer register
INA 037	102137	Transfer data buffer contents to A register
INB 037	102237	Transfer data buffer contents to B register
IME 037	102037	Transfer data buffer contents to memory
CIA 037	102537	Transfer data buffer contents to cleared A register
CIB 037	102637	Transfer data buffer contents to cleared B register
	Program Sense	
SEN 0537	101537	Sense buffer ready

Magnetic Tape Equipment

The magnetic tape system consists of up to four magnetic tape transports and a controller (Model 70-7100). The system can read and record a magnetic tape that is IBM-2400-compatible for nine-track systems.

The magnetic tape controller provides a buffered interface between the I/O bus and a nine-track magnetic tape transport. The controller accommodates up to four transports, but only one transport is in use at any one time.

The controller comprises two circuit cards and contains all read/write data buffer registers and timing and control logic required to control one tape transport.

Computer control of the magnetic tape system is accomplished through the I/O bus. The controller can also be operated under the direction of the BIC.

When more than one tape transport is used with the controller, the transports are connected to the controller in party-line configuration. However, only one transport can be operated at a time. Transport selection is under the control of the stored program.

The controller specifications are given in table 7-6, and table 7-7 lists the magnetic tape system instructions. Refer also to the magnetic tape controller manual (document number 98 A 9902 12x).

Table 7-6. Magnetic Tape Controller Specifications

Parameter	Description
Organization	Contains instruction decoder and storage logic, sense logic, read/write data controller, read/write motion controller, read/write data storage and error-checking logic, and timing and control logic
Control Capability	Four tape transports, any one of which can be selected for connection; system reset automatically selects transport 1
I/O Capability	Eight external control, eight transfer, eight program sense, and four transport selection instructions
Data Word	Provides buffering for two 16-bit words, each containing two 8-bit bytes
Error-Checking	Longitudinal redundancy check (LRC) and cyclic redundancy check (CRC) characters regenerated during reading; correction not provided
Standard Device Address	010 through 013
Size	Two wire-wrap circuit boards
Interconnection	Each card interfaces with the computer and BIC via a 122-terminal connector; two 44-terminal connectors interface each card with the tape transport

Table 7-7. Magnetic Tape Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 010	100010	Read one binary record
EXC 0210	100210	Write one binary record
EXC 0410	100410	Write a file mark
EXC 0510	100510	Skip forward one record
EXC 0610	100610	Backspace one record
EXC 0710	100710	Rewind
	Data Transfer	
OAR 010	103110	Transfer A register contents to buffer register
OBR 010	103210	Transfer B register contents to buffer register
OME 010	103010	Transfer memory contents to buffer register
INA 010	102110	Transfer buffer register contents to A register
INB 010	102210	Transfer buffer register contents to B register
IME 010	102010	Transfer buffer register contents to memory
CIA 010	102510	Transfer buffer register contents to cleared A register
CIB 010	102610	Transfer buffer register contents to cleared B register
	Program Sense	
SEN 010	101010	Sense parity error
SEN 0110	101110	Sense buffer ready
SEN 0210	101210	Sense magnetic tape unit ready
SEN 0310	101310	Sense file mark
SEN 0410	101410	Sense odd-length record/high density
SEN 0510	101510	Sense end of tape
SEN 0610	101610	Sense beginning of tape
SEN 0710	101710	Sense rewinding
	Transport Selection	
EXC2 0110	104110	Select tape drive 1
EXC2 0210	104210	Select tape drive 2
EXC2 0310	104310	Select tape drive 3
EXC2 0410	104410	Select tape drive 4

Rotating Memories

Models 70-7500 and 70-7501

Models 70-7500,-7501 disc memory and controller is a peripheral mass storage option for Varian 70 series computers. The system which includes a disc drive unit, an interface controller capable of operating up to four disc units, a removable disc pack, and all required interconnecting cables for one drive; provides storage of up to 11.7 million 16-bit words for each disc drive unit. In addition, the multiple-disc capability of the controller allows additional disc drives (up to four per controller) to be added at any time to increase the storage capability of the system. The "add-on" disc drive units are assigned model number 70-7501.

Specifications for the 70-7500,-7501 disc drives and controller are given in table 7-8. Disc memory controller instructions are listed in table 7-9.

Table 7-8. Model 70-7500,-7501 Disc Memory Specifications

Parameter	Description
Controller	
Control Capability	Up to four disc drive units
Cables to Disc(s)	TTL, twisted pair terminated
Construction	Two wire-wrap circuit boards
Disc Drive Unit	
OEM Model Number	Century Data CDS-114
Storage Capacity	11.7 million 16-bit words per disc drive unit
Recording Mode	Double frequency
Data Transfer Rate	2.50 MHz (bit rate), 312,000 bytes per second
Rotation Speed	2400 rpm ± 2 percent
Average Latency Time	12.5 milliseconds
Tract-to-Track Access Time Maximum (Full-Stroke)	10 milliseconds
Access Time	65 milliseconds
Input Power	208V ac ± 10 percent, 3-phase, 4-wire, wye-connected, 60Hz (+1 percent, -2 percent), 1000 watts
Motor Starting Current	25 amperes for 4 seconds
Power Factor	0.8
Head Dissipation	3500 BTU/hour
Construction	Free-standing unit, 40-inches high, 30-inches wide, 24-inches deep
Weight	425 pounds
Space Requirements	3 feet clear at front and back for access to service panels
Operating Environment	15.6 to 32.2 degrees C (60 to 90 degrees F) with a maximum gradient of 20 degrees F per hour; 10 to 80 percent relative humidity (no condensation)
Disc Pack	Removable disc, IBM 2316 or equivalent

Table 7-9. Model 70-7500,-7501 Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 015	100015	Stop transfer and initialize
EXC 0115	100115	Select cylinder (seek)
EXC 0215	100215	Read address
EXC 0315	100315	Write track format
EXC 0415	100415	Read/write one record
EXC 0515	100515	Recalibrate, return heads to zero
	Data Transfer	
IME 015	102015	Input to memory
INA 015	102115	Input to A register
INB 015	102215	Input to B register
CIA 015	102515	Clear and input to A register
CIB 015	102615	Clear and input to B register
OME 015	103015	Output from memory
OAR 015	103115	Output from A register
OBR 015	103215	Output from B register
	Program Sense	
SEN 015	101015	Sense disc drive 0 selecting a cylinder
SEN 0115	101115	Sense disc drive 1 selecting a cylinder
SEN 0215	101215	Sense disc drive 2 selecting a cylinder
SEN 0315	101315	Sense disc drive 3 selecting a cylinder
SEN 0415	101415	Sense DCU not busy

Models 70-7510 and 70-7511

Models 70-7510,-7511 disc memory and controller is a peripheral mass storage option for Varian 70 series computers. The system includes two individually addressable double-density drive units, and interface controller capable of operating up to four double-density spindles, power supply, and removable IBM 2316-type disc packs. The multiple-disc capability of the controller allows a slave drive unit (Model 70-7511) to be added to increase the storage capacity of the system to a maximum of four spindles. Specifications for the 70-7510,-7511 disc drives and controller are given in table 7-10. Disc memory controller instructions are listed in table 7-11.

Table 7-10. Model 70-7510,-7511 Disc Memory Specifications

Parameter	Description
Controller	
Control	Up to two disc drive units
BTC Control	Reading and writing takes place automatically via the PMA channel under BTC control
PIM Capability	The disc memory system has an interrupt capability when connected to a PIM
Construction	Two wire-wrap circuit boards
Input power	+5V dc, at 3.4 amperes
Disc Drive Unit	
OEM Model Number	CDS-215
Capacity	116 million 8-bit bytes
Transfer Rate	156,000 words per second
Positioning Time	10 milliseconds, track-to-track 55 milliseconds, full stroke
Rotational Speed	2400 rpm ± 2 percent
Average Latency	12.5 milliseconds
Pack Start Up	90 seconds
Pack Stop	11 seconds
Disc Pack Characteristics	IBM 2316 or approved equivalent. 20 surfaces each pack. Must be certified for 200 tracks per inch
Interconnection	Two cables connect disc-drive unit to the two controller boards

Table 7-10. Model 70-7510,-7511 Disc Memory Specifications *(continued)*

Parameter	Description
Power Requirements:	
Input Voltage	208 or 230V ac, ± 10 percent, 3-phase
Line Frequency	60 Hz +0.4 Hz (50 Hz optional)
Operating Current	8.6A per phase
Starting Current	20A per spindle for 7 seconds
Heat Dissipation	3500 Btu per hour per cabinet, with a 0.8 power factor
Dimensions	32-inches wide, 33-inches deep, and 61-inches high (81.4 by 83.8 by 155 cm)
Weight	Approximately 850 pounds
Space Requirements	3 feet of clearance is required in the front and back of unit for access to service panels
Operational Environment	15 to 32 degrees C; 10 to 80 percent relative humidity without condensation

Table 7-11. Model 70-7510,-7511 Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 015	100015	Stop transfer and initialize
EXC 0115	100115	Select cylinder (seek)
EXC 0215	100215	Read address
EXC 0315	100315	Write track format
EXC 0415	100415	Read/write one record
EXC 0515	100515	Recalibrate, return heads to zero
	Data Transfer	
IME 015	102015	Input to memory
INA 015	102115	Input to A register
IMB 015	102215	Input to B register
CIA 015	102515	Clear and input to A register
CIB 015	102615	Clear and input to B register
OME 015	103015	Output from memory
OAR 015	103115	Output from A register
OBR 015	103215	Output from B register
	Program Sense	
SEN 015	101015	Sense disc drive 0 selecting a cylinder
SEN 0115	101115	Sense disc drive 1 selecting a cylinder
SEN 0215	101215	Sense disc drive 2 selecting a cylinder
SEN 0315	101315	Sense disc drive 3 selecting a cylinder
SEN 0415	101415	Sense disc drive controller not busy

Models 70-7600 and 70-7601

Model 70-7600,-7601 disc memory and controller is a peripheral mass storage option for Varian 70 series computers. The system which includes a disc drive unit with two discs, an interface controller capable of operating two drives, a removable disc cartridge, a fixed disc, and all required interconnecting cables, provides storage of up to 2,350,080 16-bit words for each disc drive unit. In addition, the multiple-disc capability of the controller allows an additional disc drive (model 70-7601) to be added which doubles the capacity of the system.

Specifications for the 70-7600,-7601 disc drives and controller are given in table 7-12. Disc memory controller instructions are listed in table 7-13.

Table 7-12. Model 70-7600,-7601 Disc Memory Specifications

Parameter	Description
Controller	
Control	One or two dual-disc units per controller
BIC Control	Data transfers between computer memory and the disc memory system are controlled with a BIC
BTC Control	The disc memory system cannot be used with a PMA block transfer controller
PIM Capability	The disc memory system has an interrupt capability when connected to a PIM
Prerequisites	BIC, PIM for VORTEX systems only, and mounting space in mainframe or expansion chassis
Construction	One wire-wrap circuit board
Input Power	+5V dc, at 2 amperes
Disc Drive Unit	
OEM Model Number	Diablo 43
Disc Type	IBM 5440 (or equivalent) with 24 sectors per track
Storage Capacity	2,350,080 16-bit words
Recording Mode	Double frequency
Data Transfer Rate	92,000 words per second, 1.562 MHz (bit rate)
Rotation Speed	1500 rpm
Access Time	12 milliseconds track to adjacent track 38 milliseconds average 75 milliseconds maximum
Mounting	Mounts on slides in a 19-inch equipment rack designed according to EIA Standard RS-310

Table 7-12. Model 70-7600,-7601 Disc Memory Specifications *(continued)*

Parameter	Description
Interconnection	A 10-foot I/O cable connects to the disc controller. A 6-foot "daisy chain" cable connects to a slave unit. A 5-foot power cable connects to the disc power supply
Input Power	Supplied by disc power supply
Dimensions	19-inches wide, 10.5-inches high, 28.5-inches deep (58.3 by 26.7 by 72.4 cm). Does not include power supply
Weight	Approximately 100 pounds (without power supply)
Operational Environment	10 to 38 degrees C; 20 to 80 percent relative humidity without condensation
Disc Power Supply	
Input Power	115V ac, 5.2 amperes, or 230V ac, 2.8 amperes
Dimensions of Tray	19-inches wide (rack-mounted), 5.25-inches high, 25-inches deep (48.3 by 13.4 by 63.5 cm)
Weight of Tray	Approximately 50 pounds for one power supply Approximately 90 pounds for two power supplies
Operational Environment	Same as disc drive unit

Table 7-13. 70-7600,-7601 Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 016	100016	Select read mode and connect BIC
EXC 0116	100116	Select write mode and connect BIC
EXC 0216	100216	Set controller to seek mode
EXC 0316	100316	Set controller to sector select mode
EXC 0416	100416	Initialize controller
EXC2 016	104016	Select disc drive 0, pack 0 (non-removable)
EXC2 0116	104116	Select disc drive 0, pack 1 removable)
EXC2 0216	104216	Select disc drive 1, pack 0 (non-removable)
EXC2 0316	104316	Select disc drive 1, pack 1 (removable)
	Data Transfer	
OAR 016	103X16	Transfer out (track/sector address)
CIA 016	102X16	Transfer in (status word)
	Program Sense	
SEN 016	101016	Seek completed - disc 0, pack 0
SEN 0116	101116	Seek completed - disc 0, pack 1
SEN 0216	101216	Seek completed - disc 1, pack 0
SEN 0316	101316	Seek completed - disc 1, pack 1
SEN 0416	101416	Controller is busy
SEN 0516	101516	Error*
SEN 0616	101616	Selected disc not ready
SEN 0716	101716	Selected disc write protected

* The status word is provided by the disc controller in response to the program request for status. The meaning of the bits in the status word are as follows:

Bit	Meaning if Bit on
0	Unit 0 Seek Complete
1	Unit 1 Seek Complete
2	Unit 2 Seek Complete
3	Unit 3 Seek Complete
4	Selected Unit Illegal Sector**
5	Selected Unit Illegal Address**
6	Selected Unit Malfunction*
7	Selected Unit Timing Error**
8	Selected Unit Read Parity Error**
9	Selected End of Track Error**
10	Selected Write Protect*
11	Selected Unit - Unit Not Ready**
12	Not Used
13	Not Used
14	Not Used
15	Not Used

* Originate at the disc unit

** Reset by "Initialize"

Models 70-7610 and 70-7611

Model 70-7610,-7611 disc memory and controller is a peripheral mass storage option for Varian 70 series computers. The system which includes a disc drive unit, a removable disc cartridge, disc power supply, a disc controller, and all required interconnecting cables; provides up to 1.169,280 16-bit words for each disc drive unit. In addition, the multiple-disc capability of the controller allows two additional disc drives (model 70-7611) to be added, greatly increasing the storage capacity.

Specifications for the 70-7610,-7611 disc drives and controller are given in table 7-14. Disc memory controller instructions are given in table 7-15.

Table 7-14. Model 70-7610,-7611 Disc Memory Specifications

Parameter	Description
Controller	
Control	Up to three disc drive units per controller
BIC Control	Data transfers between computer memory and the disc memory system are controlled with a BIC
BTC Control	The disc memory system cannot be used with a PMA block transfer controller
PIM Capability	The disc memory system has an interrupt capability when connected to a PIM
Prerequisites	BIC, PIM in VORTEX systems only, and mounting space in mainframe or expansion chassis
Construction	One wire-wrap circuit board
Input Power	+5V dc, at 2 amperes
Disc Drive Unit	
OEM Model Number	Diablo 31
Disc Type	IBM 2315 with 24 sectors per track
Storage Capacity	1,169,280 16-bit words
Recording Mode	Double frequency

Table 7-14. **Model 70-7610,-7611 Disc Memory Specifications** *(continued)*

Parameter	Description
Data Transfer Rate	92,000 words per second, 1.562 MHz (bit rate)
Rotation Speed	1500 rpm
Access Time	15 milliseconds track to adjacent track 70 milliseconds average 135 millisecond maximum
Mounting	Mounts on slides in a 19-inch equipment rack designed according to EIA Standard RS-310
Interconnection	A 10-foot I/O cable connects to the disc controller. A 6-foot "daisy chain" cable connects to a slave unit. A 5-foot power cable connects to the disc power supply
Input Power	Supplied by disc power supply
Dimensions	19-inches wide, 7.25-inches high, 27.25-inches deep (48.3 by 18.4 by 69.2 cm). Does not include power supply
Weight	Approximately 40 pounds (without power supply)
Operational Environment	15 to 32 degrees C; 10 to 90 percent relative humidity without condensation
Disc Power Supply	
Input Power	115V ac, 4 amperes, or 230V ac, 2.2 amperes
Dimensions of Tray	19-inches wide (rack-mounted), 5.25-inches high, 25-inches deep (48.3 by 13.4 by 63.5 cm)
Weight of Tray	Approximately 50 pounds for one power supply Approximately 90 pounds for two power supplies
Operational Environment	Same as disc drive unit

Table 7-15. 70-7610,-7611 Disc Memory Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 016	100016	Select read mode and connect BIC
EXC 0116	100116	Select write mode and connect BIC
EXC 0216	100216	Set controller to seek mode
EXC 0316	100316	Set controller to selector select mode
EXC 0416	100416	Initialize controller
EXC2 016	104016	Select disc drive unit 0
EXC2 0116	104116	Select disc drive unit 1
EXC2 0216	104216	Select disc drive unit 2
	Data Transfer	
OAR 016	103X16	Transfer out (track/sector address)
CIA 016	102X16	Transfer in (status word)
	Program Sense	
SEN 016	101016	Seek completed, disc drive unit 0
SEN 0116	101116	Seek completed, disc drive unit 1
SEN 0216	101216	Seek completed, disc drive unit 2
SEN 0416	101416	Controller is busy
SEN 0516	101516	Error (see status word)
SEN 0616	101616	Selected disc not ready
SEN 0716	101716	Selected disc write protected

Models 70-7700, 70-7701, 77-7702, and 70-7703

Models 70-7700,-7701,-7702, and -7703 for Varian 70 series computers. The system which includes a disc drive unit, a memory system controller, a fixed disc, and all required interconnecting cables; provides storage of up to 491,000 16-bit words. The capacity and number of tracks for each model are listed in table 7-16.

Table 7-16. Capacity and Number of Tracks

Model Number	Word Storage	Number of Tracks
70-7700	61K	16
70-7701	123K	32
70-7702	246K	64
70-7703	491K	128

Specifications for the 70-7700,-7701,-7702, and -7703 fixed-head rotating memory devices and controllers are listed in table 7-17. Rotating memory controller instructions are listed in table 7-18.

Table 7-17. Models 70-7700,-7701,-7702, and -7703 Controller Instructions

Parameter	Description
Controller	
Control	Controls one rotating memory unit
PIM Capability	The rotating memory system has an interrupt capability when connected to a PIM
Construction	One wire-wrap circuit board
Rotating Memory Unit	
OEM Models	General Instruments 500 FR/VDM-16, 500 FR/VDM-32, 500 FR/VDM-64, 500 FR/VDM-128
Capacity:	
Model 70-7700	61,000 16-bit words
Model 70-7701	123,000 16-bit words
Model 70-7702	246,000 16-bit words
Model 70-7703	491,000 16-bit words
Transfer Rate	106,000 words per second
Average Access Time	17 milliseconds
Rotational Speed	1800 rpm ± 10 percent
Disc Characteristics	2 surfaces, one fixed head per track, 16 to 128 tracks, 3840 17-bit words per track, 16 data bits plus parity
Input Power	208 or 230V ac ± 10 percent at 3 amperes
Dimensions	12-1/4-inches high, 18-inches wide, 22-inches deep (31 by 45.7 by 55.9 cm)
Weight	150 pounds
Operational Environment	10 to 40 degrees C; 10 to 90 percent relative humidity without condensation

Table 7-18. Models 70-7700,-7701,-7702, and -7703 Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 014	100014	Select read mode and connect controller to BIC
EXC 0114	100114	Select write mode and connect controller to BIC
	Data Transfer	
OAR 014	103114	Output starting disc address from A register
OBR 014	103214	Output starting disc address from B register
OME 014	103014	Output starting disc address from memory
	Program Sense	
SEN 0114	101114	Disc ready to execute a mode select instruction
SEN 0214	101214	Disc is busy with block transfer operation
SEN 0314	101314	Illegal address selected
SEN 0414	101414	Read parity error detected
SEN 0614	101614	Data transfer timing error

Printer/Plotter

Statos 31 Family

Models 70-6602, 70-6606, and 70-6608

The Statos 31 Printer/Plotters feature high-speed output directly from the data source. They provide quiet, trouble-free operation by the use of electrostatic, non impact, printing. Included in each system are: Statos 31 printer/plotter, controller, cables and connectors, software driver, test and diagnostic routines, and user's manual. A variety of options are available to further enhance the plotting and printing capabilities.

The specifications for models 70-6602, 70-6606, and 70-6608 Statos 31 Printer/Plotters are listed in table 7-19. The Statos 31 Printer/Plotter instructions are listed in table 7-20.

Table 7-19. Models 70-6602, 70-6606, and 70-6608 Statos 31 Printer Plotter Specifications

Parameter	Model 70-6602	Model 70-6606	Model 70-6608
Paper width	14-7/8 inches (38.1 cm)	8-1/2 inches (21.6 cm)	11 inches (27.9 cm)
Paper form	Roll or fan-fold	Roll or fan-fold	Roll or fan-fold
Styli configuration	1408 styli across 14.08 inches (35.75 cm)	640 across 8 inches (20.3 cm)	1056 across 10.56 inches (26.8 cm)
Stylus density	100 styli per inch (0.01 inch spacing) (0.254 mm)	80 styli per inch (0.0125-inch spacing) (0.318 mm)	100 styli per inch (0.01 inch spacing) (0.254 mm)
Step increment	100 steps per inch (0.01 inch size) (0.254 mm)	80 steps per inch (0.0125 inch size) (0.318 mm)	100 steps per inch (0.01 inch size) (0.254 mm)
Maximum asynchronous stepping rate	167 steps per second (equivalent to 770 alphanumeric lines per minute at 8 lines per inch)	167 steps per second (equivalent to 1000 alphanumeric lines per minute)	167 steps per second (equivalent to 770 alphanumeric lines per minute at 8 lines per inch)
Maximum synchronous stepping rate	220 steps per second (equivalent to 1000 alphanumeric lines per minute at 8 lines per inch)	220 steps per second (equivalent to 1320 alphanumeric lines per minute)	220 steps per second (equivalent to 1000 alphanumeric lines per minute at 8 lines per inch)
Maximum paper speed	2.2 inches per second (5.6 cm)	2.75 inches per second (7.0 cm)	2.2 inches per second (5.6 cm)
Minimum paper speed	0.01 inch per second (0.254 mm)	0.0125 inch per second (0.318 mm)	0.01 inch per second (0.254 mm)

Table 7-20. Models 70-6602, 70-6606, and 70-6608 Statos 31 Printer/Plotter Instructions

Interface Control	Function
EXC 0XX	Control select
EXC 01XX	System reset
OAR, OBR, OME	Word transfer
0103	Data select
0105	BIC connect
0307	BIC connect/Data select
0111	Bus reset

Data Destination Control	Function
0340	Raster data
0100	Character generator data (optional)

Printer/ Plotter Control	Function
040	Enable Statos
0241	Disable Statos
0242	Line sync
000	NOP (no operation)
064	Step paper
0265	Slew paper
0263	Form feed
062	Cut (optional)
043	Line sync/step
0244	Autostep select*
045	Autostep reset

*Autostep is a hard-wired sequence for Line Sync and Step Paper to maximize throughput under BIC operation.

Sense Instructions

Mnemonic	Function
SEN 0XX	Input busy
SEN 01XX	Statos not ready
SEN 02XX	Scan complete
SEN 03XX	Paper controller busy
SEN 04XX	Buffer busy (optional with HCG options)
SEN 06XX	Bottom-of-form
SEN 07XX	Roll paper

XX = device code address

Statos 31 Family

Models -6611, -6613, -6615, and -6617

Printer/Plotters

The Statos 33 printer/plotters with the above model numbers have the BI-SCAN writing head. This writing head incorporates a double row of styli for ultra sharp, high contrast hard copy. Table 7-21 lists the specifications for the Statos 33 model 70-6611 through 70-6617 printer/plotter.

Table 7-21. Model 70-6611 through 70- 6617 Statos 33 Printer/Plotter Specifications

Parameter	Model 6611	Model 6613
Paper width	8-1/2 inches (21.6 cm)	11 inches (27.9 cm)
Paper form	Roll or fan-fold	Roll or fan-fold
Styli configuration	640 across 8 inches (20.3 cm)	1056 across 10.56 inches (26.8 cm)
Stylus density	100 styli per inch (0.01 inch spacing) (0.254 mm)	100 styli per inch (0.01 inch spacing) (0.254 mm)
Step increment	80 steps per inch (0.0125 inch size) (0.318 mm)	100 steps per inch (0.01 inch size) (0.254 mm)
Maximum asynchronous stepping rate	100 steps per second (equivalent to 460 alphanumeric lines per minute at 8 lines per inch).	90 steps per second (equivalent to 410 alphanumeric lines per minute at 8 lines per inch)
Maximum paper speed	1.5 inches per second (3.8 cm)	1.5 inches per second (3.8 cm)
Minimum paper speed	0.01 inches per second (0.254 mm)	0.01 inches per second (0.254 mm)
Input power	115V ac, 60 Hz, (Standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)	115V ac, 60 Hz, (Standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)
Power drain	500 watts (peak)	500 watts (peak)
Temperature	+40 to 100 degrees F	+40 to 100 degrees F
Humidity	(5 to 32 degrees C)	(5 to 32 degrees C)
Altitude	15 to 85 percent	15 to 85 percent
Size	To 7,500 feet (2,250 m)	To 7,500 feet (2,250 m)
	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)
Weight	325 pounds (148 kg)	325 pounds (148 kg)
Attitude	± 15 degrees from vertical	± 15 degrees from vertical

Table 7-21. Model 70-6611 through 70- 6617 Statos 33 Printer/Plotter Specifications *(continued)*

Parameter	Model 6615	Model 6617
Paper width	14-7/8 inches (38.1 cm)	22 inches (55.9 cm)
Paper form	Roll or fan-fold	Roll
Styli configuration	1408 styli across 14.08 inches (35.75 cm)	2112 styli across 21.12 inches (53.6 cm)
Stylus density	100 styli per inch (0.01 inch spacing (0.254 mm)	100 styli per inch (0.01 inch spacing) (0.254 mm)
Step increment	100 steps per inch (0.01 inch size) (0.254 mm)	100 steps per inch (0.01 inch size) (0.254 mm)
Maximum asynchronous stepping rate	80 steps per second (equivalent to 370 alphanumeric lines per minute at 8 lines per inch)	60 steps per second (equivalent to 300 alphanumeric lines per minute at 8 lines per inch)
Maximum paper speed	1.5 inches per second (3.8 cm)	1.5 inches per second (3.8 cm)
Minimum paper speed	0.01 inches per second (0.254 mm)	0.01 inches per second (0.254 mm)
Input power	115V ac, 60 Hz, (Standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)	115V ac, 60 Hz, (Standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)
Power drain	500 watts (peak)	500 watts (peak)
Temperature	+40 to 100 degrees F	+40 to 100 degrees F
Humidity	(5 to 32 degrees C)	(5 to 32 degrees C)
Altitude	15 to 85 percent	15 to 85 percent
Size	To 7,500 feet (2,250 cm)	To 7,500 feet (2,250 m)
	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)	38 inches high (96.5 cm) x 31-1/4 inches wide (75.0 cm) x 26-1/4 inches deep (66.7 cm)
Weight	325 pounds (148 kg)	365 pounds (approximate) (166 kg)
Attitude	± 15 degrees from vertical	± 15 degrees from vertical

Models 70-6621, 6623, -6625, and -6627 Printer/Plotters

The Statos 33 Printer/Plotter with the above model numbers have a linear head. This writing head produces clear, easy-to-read output at high speed. Table 7-22 lists the specifications for the Statos 33 model 70-6620 through 70-6627 printer/plotters.

Table 7-22. Model 70-6621 through 70- 6627 Statos 33 Printer/Plotter Specifications

Parameter	Model 70-6621	Model 70-6623
Paper width	8-1/2 inches (21.6 cm)	11 inches (27.9 cm)
Paper form	Roll or fan-fold	Roll or fan-fold
Styli configuration	640 across 8 inches (20.3 cm)	1056 across 10.56 inches (26.8 cm)
Stylus density	100 styli per inch (0.01 inch spacing) (0.254 mm)	100 styli per inch (0.01 inch spacing) (0.254 mm)
Step increment	80 steps per inch (0.0125 inch size) (0.318 mm)	100 steps per inch (0.01 inch size) (0.254 mm)
Maximum asynchronous stepping rate	150 steps per second (equivalent to 690 alphanumeric lines per minute at 8 lines per inch)	150 steps per second (equivalent to 690 alphanumeric lines per minute at 8 lines per inch)
Maximum paper speed	1.5 inches per second (3.8 cm	1.5 inches per second (3.8 cm)
Minimum paper speed	0.01 inches per second (0.254 mm)	0.01 inches per second (0.254 mm)
Input power	115V ac, 60 Hz, (standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)	115V ac, 60 Hz, (standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)
Power drain	500 watts (peak)	500 watts (peak)
Temperature	+40 to 100 degrees F	+40 to 100 degrees F
Humidity	(5 to 32 degrees C)	(5 to 32 degrees C)
Altitude	15 to 85 percent	15 to 85 percent
Size	To 7,500 feet (2,250 m)	To 7,500 feet (2,250 m)
	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)
Weight	325 pounds (148 kg)	325 pounds (148 kg)
Attitude	± 15 degrees from vertical	± 15 degrees from vertical

Table 7-22. Model 70-6621 through 70-6627 Statos 33 Printer/Plotter Specifications *(continued)*

Parameter	Model 70-6625	Model 70-6627
Paper width	14-7/8 inches (38.1 cm)	22 inches (55.9 cm)
Paper form	Roll or fan-fold	Roll
Styli configuration	1408 styli across 14.08 inches (35.75 cm)	2112 styli across 21.12 inches (53.6 cm)
Stylus density	100 styli per inch (0.01 inch spacing) (0.254 mm)	100 styli per inch (0.01 inch spacing) (0.254 mm)
Step increment	100 steps per inch (0.01 inch size) (0.254 mm)	100 steps per inch (0.01 inch size) (0.254 mm)
Maximum asynchronous stepping rate	150 steps per second (equivalent to 690 alphanumeric lines per minute at 8 lines per inch)	120 steps per second (equivalent to 550 alphanumeric lines per minute at 8 lines per inch)
Maximum paper speed	1.5 inches per second (3.8 cm)	1.5 inches per second (3.8 cm)
Minimum paper speed	0.01 inches per second (0.254 mm)	0.01 inches per second (0.254 mm)
Output power	115V ac, 60 Hz, (standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)	115V ac, 60 Hz, (standard) 230V ac, 50 Hz, (optional) 100V ac, 50 Hz, (optional)
Power drain	500 watts (peak)	500 watts (peak)
Temperature	+40 to 100 degrees F	+40 to 100 degrees F
Humidity	(5 to 32 degrees C)	(5 to 32 degrees C)
Altitude	15 to 85 percent	15 to 85 percent
Size	To 7,500 feet (2,250 m)	To 7,500 feet (2,250 m)
	38 inches high (96.5 cm) x 24-1/2 inches wide (62.3 cm) x 24-1/4 inches deep (61.5 cm)	38 inches high (96.5 cm) x 32-1/4 inches wide (75.0 cm) x 26-1/4 inches deep (66.7 cm)
Weight	325 pounds (148 kg)	365 pounds (approximate) (166 kg)
Attitude	± 15 degrees from vertical	± 15 degrees from vertical

Line Printer Equipment
Models 70-6720, 70-6721

The line printer system (model 70-6720) comprises a 300 line-per-minute printer and a peripheral controller.

This operational, self contained system offers medium speed with high printing quality for a wide range of on-line computer output applications. It has a forms length selector switch to select 11 different prewired form lengths.

The line printer system (model 70-6721) is the same as model 70-7620 with a tape controlled vertical format unit (TCVFU). The tape is 12 channels wide and allows for a variety of form lengths and allows rapid paper slewing within individual forms.

The 300 line-per-minute printer system specifications are given in table 7-23, and table 7-24 lists the line printer controller instructions.

Table 7-23. Models 70-6720 and 70-6721 Line Printer Specifications

Parameter	Description
Controller	
Organization	Contains timing and control logic, select/ deselect logic, word buffer, and drivers and receivers
Control Capability	One model 70-6720 line printer, six bit words
I/O Capability	Five external control, three transfer, and three program sense instructions
Standard Device Address	035
Size	One wire-wrap circuit board
Input Power	+5V dc at 1.5 amps
Line Printer	
Character Data	Format: standard ASCII Characters per line: 136 Horizontal spacing; ten characters per inch Vertical spacing: six or eight lines per inch
Dynamic Characterstics	Printing speed: 300-lines per minute Drum rotation: 1,200 rpm Primary power: 115, 220, 240V ac-50 or 60 Hz Current requirement: 525 watts
Paper	Type: standard fanfold, edge-punch, single copy minimum 15-pound bond, multicopy (up to six parts) of 12-pound bond with single shot carbon Size: 4-to-16.75 inches (8.1 to 42.5 cm) wide with 11 inches (27.9 cm) between folds
Ribbon	Type: vertically fed roll Size: 15 inches (38.1 cm) wide and 20-yards (18 m) long
Dimensions	45-inches (114.3 cm) high, 33 inches (83.8 cm) wide, and 22 inches (55.9 cm) deep
Weight	Approximately 340 pounds (154.2 kg)
Operational Environment	10 to 40 degrees C; 30 to 90 percent relative humidity without condensation

Table 7-24. Model 70-6720 and 70-6721 Line Printer Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 035	100035	Write connect (BIC)
EXC 135	100135	Write connect
EXC 0435	100435	Initialize controller
EXC2 035	104035	8 bit ASCII mode
EXC2 135	104135	Format command
	Data Transfer	
OAR 035	103135	Transfer A register contents to printer buffer
OBR 035	103235	Transfer B register contents to printer buffer
OME 035	103035	Transfer memory contents to printer buffer
	Program Sense	
SEN 0035	101035	Printer ready
SEN 0135	101135	Character buffer ready
SEN 0635	101635	Printer alert

Models 70-6722, 70-6723

The line printer system (model 70-6722 or 70-6723) comprises a 600 line-per-minute printer and a peripheral controller.

The model 70-6722 printer has an 11-position form-length selector switch which allows the operator to conventiently handle a variety of commonly used form lengths and to advance the paper the appropriate number of lines to top-of-form, under push-button or program control. The selector switch positions correspond to form lengths of: 3, 3-1/2, 4, 5-1/2, 6, 7, 8, 8-1/2, 11, 12, and 14 inches. (The standard top-of-form spacing is set for 11-inches at 6 or 8 lines-per-inch.)

The model 70-6723 printer has a 12-channel tape-controlled vertical format unit. The vertical format unit, consisting of a tape-reader and associated electronics, enables handling of a variety of form lengths and allows rapid paper slewing within individual forms. Upon initialization the paper-tape is read into the printer memory thus eliminating continuous paper-tape motion. The paper-tape reader uses 12-channel IBM carriage tape or equivalent.

The 600 line-per-minute printer system specifications are given in table 7-25, and table 7-26 lists the line printer controller instructions.

Table 7-25. Models 70-6722 and 70-6723 Line Printer Specifications

Parameter	Description
Printer Controller	
Control.	Either line printer system can be operated either under programmed I/O control or under control of a BIC (buffer interlace controller).
BIC control	Data transfers between computer memory and the line printer can be implemented with a BIC.
BTC control	The line printer system cannot be used with the block transfer controller (BTC).
PIM capability	The line printer system has an interrupt capability when connected to a PIM (priority interrupt module).
Prerequisites	Mounting space in mainframe or expansion chassis.
Dimensions	Contained on one 7.75 by 12 inch (19.7 by 30.3 cm) circuit board.
Interconnection	Requires one slot of an I/O expansion chassis: Connects to the computer and BIC via a 122-terminal connector. One 44-terminal connector attaches to the line printer through a 20-foot cable.
Logic levels	Jumper selectable input/output.
Negative true (standard)	Logic 1 = 0 to 0.5V dc. Logic 0 = +2.4 to +5.0V dc.
Positive true	Logic 1 = +2.4 to +5.0V dc. Logic 0 = 0 to 0.5V dc.
Input power	+5V dc, at 1 ampere.
Operational environment	0 to 50 degrees C (32 to 122 degrees F), 0 to 90 percent relative humidity without condensation.

Table 7-25. Models 70-6722 and 70-6723 Line Printer Specifications *(continued)*

Parameter	Description
Line Printer	
Printing speed	600 lines-per-minute
Character format	64 Gothic print
Characters/line	136 characters
Drum rotation	800 rpm
Line advance time	25 milliseconds (maximum)
Paper slew speed	25 inches per second (minimum)
Buffer	136 characters
Primary power	115, 220, 240V ac, 50 or 60 Hz 680 watts (maximum current)
Dimensions	33 inches wide, 45 inches high, 25 inches deep (83.8 by 114.3 by 63.5 cm)
Weight	370 pounds
Operational environment	10 to 37.8 degrees C, 80 to 90 percent relative humidity without condensation.

Table 7-26. Models 70-6722 and 70-6723 Line Printer Controller Instructions

Mnemonic	Octal Code	Function
External Control		
EXC 01	100135	Write connection (BIC)
EXC 04	100435	Initialize controller
Sense		
SEN 00	101035	Printer ready
SEN 01	101135	Buffer ready
SEN 06	101635	Printer error
Transfer		
OAR	103135	Output from A register
OBR	103235	Output from B register
OME	103035	Output from memory

Oscilloscope Display Terminal

Model 70-6400

The oscilloscope display (model 70-6400) provides a visual display of the computer output. The display module permits visual display with high resolution and flutter-free performance, and provides a stable trace of alphanumeric and graphic information without periodic computer update.

The oscilloscope controller converts digital data to analog values that drive the horizontal and vertical deflection plates of the oscilloscope. The oscilloscope unit is designed for X-Y presentation. One of the digital-to-analog (D/A) converters in the controller drives the X axis; the other, the Y axis. The Z channel input turns the CRT beam on and off.

The oscilloscope display system is directly under program control. The display is programmed by outputting data to one of the two D/A converters. The mode of operation and screen erasing can be selected by the program.

The oscilloscope display system specifications are given in table 7-27 and table 7-28 lists the oscilloscope controller instructions.

Table 7-27. Oscilloscope Display System Specifications

Parameter	Description
Controller	
Organization	Contains decoding logic, two D/A converters, Z axis controller, and voltage regulator
Control Capability	One model 70-6400 oscilloscope unit
I/O Capability	Eight external control, three transfer instructions, and one sense instruction
Size	One wire-wrap circuit board
Oscilloscope	
OEM Model Number	Tektronix Model 611, Mod. 162C
Display Area	20.48 centimeters in the horizontal (X-Axis) and 16.00 centimeters in the vertical (Y-Axis) with 1024 by 800 addressable grid points
Display Linearity	The voltage required to produce a 2 centimeter deflection at any point on the CRT will not vary more than 10 percent
Viewing Time	At least 15 minutes without loss of resolution
Settling Time	3.5 microseconds/centimeter +5 microseconds
Input Power	+15V dc ± 0.1 percent at 90 mA -15V dc ± 3 percent at 90 mA +5V dc ± 1 percent at 850 mA
Resolution	0.02 cm in both X and Y axes
Erase Time	0.5 seconds
Dot Writing Time	5.0 microseconds
Mounting	Table top (optional rack mount)
Dimensions:	
Height	11-7/8 inches (30.1 cm)
Width	11-5/8 inches (29.5 cm)
Depth	22-3/8 inches (56.8 cm)
Weight	51 pounds (23.1 kg)

Table 7-28. Oscilloscope Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 025x	10025x	Pulse Z-axis (write dot)
EXC 035x	10035x	Erase Screen
EXC 045x	10045x	Select NON-STORE mode
EXC 055x	10055x	Select STORE mode
EXC 065x	10065x	Select WRITE THRU mode
EXC 075x	10075x	Select NON-WRITE THRU mode
EXC2 05x	10405x	Select X DAC
EXC2 015x	10415x	Select Y DAC
	Data Transfer	
OAR 05x	10315x	Output A register to selected DAC
OBR 05x	10325x	Output B register to selected DAC
OME 0x	10305x	Output from memory to selected DAC
	Program Sense	
SEN 05x	10105x	Sense ERASE INTERVAL true

Where x = last digit of the device address

Keyboard Display Terminal

The keyboard/display terminal (model 70-6401) features a Teletype compatible keyboard and an 80-character 24-line display. The terminal provides an efficient means of communication between the operator and computer system. The terminal is a desk-top unit that contains its own power supply and two kinds of asynchronous serial interfaces consisting of a standard EIA RS232C and a 20 or 60 mA Teletype-style current loop.

The specifications for the keyboard/display terminal are given in table 7-29 and 7-30 lists the keyboard/display terminal instructions.

Table 7-29. Model 70-6401 Keyboard/ Display Specifications

VDM model number	70-6401
OEM model number	Infoton Incorporated, Vistar/GT.
Interfaces	EIA RS232C, and 20 or 60 mA current loop.
Receiving rate	Eleven switch-selectable rates from 110 to 9600 baud. When operating with VORTEX, the maximum rate is 2400 baud.
Stop bits	A two-position switch is used to determine whether transmitted characters contain 10 bits (one Stop bit) or 11 bits (two Stop bits). When operating with VORTEX, the switch is set to the 10-bit position.
Parity	A three-position switch is used to select Odd, Even, or Mark parity. When operating with VORTEX, the switch is set to the Mark position.
Keyboard	Compatible with KSR-33 and -35 Teletypes.
Lines per display	24
Characters per line	80
Character set	64-character ASCII, upper case.
Character format	5 by 7 dot matrix.
Character size	0.08 by 0.19 inches (0.20 by 0.48 cm)
Viewing area	9 inches wide and 7 inches high.
Display color	Displayed characters are white (P4 phosphor) on a dark background.
Readability	Screen easily read without disruptive reflections in a 100 foot-candle illumination.
Mounting	Desk-top.
Interconnection	Connects to computer via a 20-foot (6.1 m) I/O cable.

Table 7-29. Model 70-6401 Keyboard/ Display Specifications
(continued)

Input power	115V ac ± 10 percent at 1 ampere, 230V ac ± at 0.5 ampere, 50 or 60 Hz.
Dimensions	19 inches wide, 13 inches high, 23 inches deep (48.3 by 33.0 by 58.4 cm).
Weight	Approximately 35 pounds (15.9 kg).
Operational environment	0 to 50 degrees C (32 to 122 degrees F), 10 to 95 percent relative humidity without condensation.

Table 7-30. Model 70-6401 Keyboard/ Display Instructions

Mnemonic	Octal Code	Function
	External Control	
EXC 0	1000xx*	Connects controller to BIC (for output-data transfers).
EXC 01	1001xx	Connects controller to BIC (for output-data transfers).
EXC 02	1002xx	Connects controller to BIC (for input-data transfers).
EXC 04	1004xx	Initializes the controller.
EXC 05	1005xx	Connects controller to BIC (for input-data transfers).
	Sense	
SEN 0	1010xx	Senses for a frame error or break.
SEN 01	1011xx	Sense if the controller is ready for output transfers (computer to terminal).

Table 7-30. Model 70-6401 Keyboard/ Display Instructions *(continued)*

Mnemonic	Octal Code	Function
SEN 02	1012xx	Senses if the controller is ready for input transfers (terminal to computer).
SEN 03	1013xx	Not used.
SEN 04	1014xx	Senses for input parity error.
SEN 05	1015xx	Not used.
SEN 07	1017xx	Input overflow error.
	Transfer	
IME xx	1020xx	Transfers data from the controller to main memory.
INA xx	1021xx	Transfers data from the controller to the A register.
INB xx	1022xx	Transfers data from the controller to the B register.
CIA xx	1025xx	Transfers data from the controller to the cleared A register.
CIB xx	1026xx	Transfers data from the controller to the cleared B register.
OME xx	1030xx	Transfers address from main memory to the controller.
OAR xx	1031xx	Transfers data from the A register to the controller.
OBR xx	1032xx	Transfers data from the B register to the controller.

*xx = device address, normally in the range 01 through 07.

Punched Card Equipment

Card Reader

The card reader system (model 70-6200) reads data from 80-column punched cards and transfers the data to the computer. The system consists of a card reader and a controller.

The card reader controller provides a nonbuffered interface between the reader and the computer. It also provides the timing and control logic to effect the transfers.

The controller can transfer data to the computer under direct program control or under the supervision of the BIC.

The controller specifications are given in table 7-31. and table 7-32 lists its instructions.

Table 7-31. Card Reader Controller Specifications

Parameter	Description
Organization	Contains input receivers and output drivers for I/O bus and BIC interfacing, and a sequence controller to transfer data from the reader to the computer
Control Capability	One punched card reader unit
I/O Capability	Three external control, five transfer, and five program sense instructions
Reader Characteristics	Operating speed: 300 cards per minute Character length: 12 bits Card type: standard 80-column read on a per-column basis (end feed) Transfer rate: 1,050 characters per second
Standard Device Address	030
Size	One wire-wrap circuit board
Input Power	+5V dc at 0.5 amperes

Table 7-32: Card Reader Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 030	100030	Initialize reader
EXC 0230	100230	Read one card
EXC 0330	100330	Feed cards continuously
	Data Transfer	
INA 030	102130	Transfer character to A register
INB 030	102230	Transfer character to B register
IME 030	102030	Transfer character to memory
CIA 030	102530	Transfer character to cleared A register
CIB 030	102630	Transfer character to cleared B register
	Program Sense	
SEN 030	101030	Card in read station
SEN 0130	101130	Character ready
SEN 0230	101230	Reader error
SEN 0330	101330	Hopper empty
SEN 0630	101630	Reader ready

Card Punch

The card punch system (model 70-6201) comprises a punch and a controller. The controller controls data transfers from the computer to the card punch. The system can be operated under CPU control and, optionally, under BIC control.

The data buffer register in the controller stores the 12-bit data words from the CPU. The control section synchronizes the control punch operation.

Under program control, the controller senses the punch (ready or not busy) and transfers data to it. Under BIC control, the controller requests data and the BIC controls the transfer from the specified memory addresses. Transfer of the data continues until terminated by the BIC.

The controller specifications are given in table 7-33, and table 7-34 lists its instructions.

Table 7-33. Card Punch Controller Specifications

Parameter	Description
Organization	Contains line drivers and receivers for I/O and punch cabling, punch output data buffer, and control logic
Control Capability	One card punch unit
I/O Capability	Two external control, three transfer, and one program sense instructions
Punch Characteristics	Character length: 12 bits Card type: standard 80-column punched on a per-column basis Transfer rate: 35 cards per minute
Standard Device Address	031
Size	One wire-wrap circuit board
Input Power	+5V dc at 485 milliamperes

Table 7-34. Card Punch Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0031	100031	Initialize
EXC 0131	100131	Connect punch to BIC
	Data Transfer	
OAR 031	103131	Transfer initial buffer address to BIC from A register
OBR 031	103231	Transfer initial buffer address to BIC from B register
OME 031	103031	Transfer initial buffer address to BIC from memory
	Program Sense	
SEN 031	101031	Card punch busy

Digital Controllers

Buffered I/O Controller Model 70-8301

The buffered I/O controller (model 70-8301) provides a self-contained programmable hardware interface for general-purpose data processing.

The input and output buffer registers provide parallel-word data communications between the computer I/O bus and an external device. In addition to data-handling, the output buffer register can be programmed to output discrete control signals to an external device.

The buffered I/O controller uses a user-supplied cable, up to 20-feet (6 m) long, for communication with external devices.

The buffered I/O controller specifications are given in table 7-35, and table 7-36 lists the controller instructions.

Table 7-35. Buffered I/O Controller Specifications

Parameter	Description
Organization	Contains operation and function decoder, input and output buffer registers, eight sense input gates and eight variable-pulsewidth control gates, and interface drivers and receivers
Control Capability	Provides buffered data transmission to and from external devices and the computer
I/O Capability	One external control, eight transfer, and one program sense instructions
Standard Device Address	060 through 067
Current Load	Sensing line input: nominally 7-milliampere source at 0 volt Buffered output register: nominally 36-milliampere source at 0 volt
Current Drive	Up to 65-milliampere sink at 0 volt
Size	One printed circuit board
Input Power	+5V dc at 1.0 ampere

Table 7-36. Buffered I/O Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0x6z	100x6z	Output control pulse on line selected by x from controller addressed by z
	Data Transfer	
OAR 06z	10316z	Load output buffer of controller addressed by z from A register
OBR 06z	10326z	Load output buffer of controller addressed by z from B register
OME 06z	10306z	Load output buffer of controller addressed by z from memory
INA 06z	10216z	Read input buffer of controller addressed by z into A register
INB 06z	10226z	Read input buffer of controller addressed by z into B register
IME 06z	10206z	Read input buffer of controller addressed by z into memory
CIA 06z	10256z	Read input buffer of controller addressed by z into cleared A register
CIB 06z	10266z	Read input buffer of controller addressed by z into cleared B register
	Program Sense	
SEN 0x6z	101x6z	Test state of line selected by x from controller addressed by z

where x = discrete control/sense line (00-07), and z = last digit of the controller device address.

Relay I/O Module

Models 70-8500, 70-8501, 70-8502

The relay I/O module (model 70-8500, 70-8501, or 70-8502) provides a general-purpose, relay-buffered data link between special external devices and the computer. This I/O interface option has the capability of 16 relay-buffered inputs (70-8500), 16 mercury-wetted relay contact outputs (70-8501), or combined input/output (70-8502).

In the relay-buffered input configuration (70-8500), input relay contacts are activated by voltages from the user's equipment through the 12V dc input relay coil. Series resistors for coil input voltages greater than 12V can be installed. An energized input relay coil closes a contact that is gated into a 16-bit flip-flop register that can be accessed by the computer with one of five transfer instructions. The register can be cleared with external control instructions or by system reset. The flip-flops remain set after an initial relay input until they are cleared.

The 70-8501 relay output module allows isolated parallel transfer of a 16-bit word from the computer via mercury-wetted relay contacts to the user's equipment. The 16-bit word is clocked into a flip-flop register by any of three transfer instructions. The register drives 16 discrete circuits that, in turn, drive the 12V relay coils closing the contacts.

The relays can be cleared by an external control instruction, transferring all-zeros data out, or by system reset. The relay contacts remain closed until the flip-flops are cleared.

The specifications of the relay I/O module are given in table 7-37, and table 7-38 lists its instructions.

Table 7-37. Relay I/O Module Specifications

Parameter	Description
Relay Type	Output: mercury-wetted reed Input: dry reed
Contact Rating	Output: 50 volt-amperes resistive; 3 amperes or 400 volts (maximum) Input: 12 volt-amperes resistive; 0.5 ampere or 200 volts (maximum)
Contact Responses	Output: 0.05 ohm (average) Input: 0.2 ohm (average)
Capacitance	Output: 1 picofarad plus 0.01 microfarad external Input: less than 1 picofarad
Operating Time	Output: 2 milliseconds (average) Input: 1 millisecond (average including bounce)
Release Time	Output: 2 milliseconds (average) Input: 1 millisecond (average)
Coil Power Consumption	Output: 500 milliwatts (average) Input: 100 milliwatts (average)
Drive Requirements	Output: three TTL load (through an amplifier) Input: 10 volts at 6.5 milliamperes (minimum)
Rated Load Life	Output: 25 million operations Input: 100 million operations
Standard Device Address	074 through 077
Size	One wire-wrap circuit board

Table 7-38. Relay I/O Module Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 007x	10007x	Clear all outputs
EXC 017x	10017x	Clear all inputs
	Data Transfer	
OAR 07x	10317x	Transfer A register contents to buffered relay output contacts
OBR 07x	10327x	Transfer B register contents to buffered relay output contacts
OME 07x	10307x	Transfer memory contents to buffered relay output contacts
INA 07x	10217x	Transfer relay-buffered data to A register
INB 07x	10227x	Transfer relay-buffered data to B register
IME 07x	10207x	Transfer relay-buffered data to memory
CIA 07x	10257x	Transfer relay-buffered data to cleared A register
CIB 07x	10267x	Transfer relay-buffered data to cleared B register
	Program Sense	
SEN 07x	10107x	Contact closed

where x = last digit of the device address.

Analog/Digital Conversion Equipment

Analog Input Module (Models 70-8010, 70-8011)

The Models 70-8010 and -8011 analog input modules (AIM) convert multiplexed analog input signals to digital values for use in the computer. The basic AIM comprises an analog-to-digital converter (ADC), a 16-channel (single-ended or differential) multiplexer, and control logic. The Model 70-8200 A/D power supply is a prerequisite.

The multiplexer accepts, samples, and holds high-level analog signals for conversion by the ADC. The basic multiplexer services 16 differential, or single-ended, channels. This basic configuration can be expanded to a maximum of 256 channels in 16-channel increments. The multiplexer selects channels for input to the ADC either sequentially or on a random basis.

The ADC, using successive approximations, converts analog signals passed from the multiplexer to equivalent 13-bit digital values at an effective throughput of 50,000 conversions per second. ADC conversions can be initiated by the computer, an internal programmable timer, or an external pulse.

The control logic implements the transfer of the converted data to the computer under program control or under the direction of the optional PIM and BIC.

The specifications of the AIM are given in table 7-39, and table 7-40 lists the AIM instructions.

Table 7-39. Analog Input Module Specifications

Parameter	Description
Conversion	50,000 samples per second (maximum)
Resolution	13 bits
Conversion Accuracy	± 0.012 percent, ± 1/2 LSB
Conversion Time	13 microseconds (maximum)
Full-Scale Range	± 10 volts
Temperature Coefficient	± 50 microvolts per degree C (maximum)
Warm-Up Time	Essentially zero
Multiplexer Accuracy	± 0.01 percent
Source Impedance	1 kilohm
Voltage Range	± 10V dc at 100 milliamperes
Output Impedance	20 ohms
Prerequisite	One model 70-8200 A/D power supply
Standard Device Address	054 through 057
Size	Two etched circuit boards
Input Power	+ 5V dc ± 5 percent at 2 amperes ± 15V dc ± 3 percent at 165 milliamperes + 20V dc ± 5 percent at 15 milliamperes − 22V dc ± 5 percent at 5 milliamperes

Table 7-40. Analog Input Module Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0160	100160	Start conversion cycle
EXC 0270	100270	Select sequential scan of channels
EXC 0170	100170	Select random scan of channels
EXC 070	100070	Set utility flip-flop output to ground
EXC 0370	100370	Reset utility flip-flop output to +5V dc
	Data Transfer	
OAR 060	103160	Transfer number of microseconds in timed interval
OAR 070	103170	Transfer channel number
CIA 060	103560	Transfer data to cleared A register
	Program Sense	
SEN 060	100060	ADC ready
SEN 0160	100160	Timer flag
SEN 0260	100260	Utility input
SEN 070	100070	End of sequential scan

Analog Output Module (Models 70-8210, 70-8221)

The models 70-8210 through 70-8221 analog output modules (AOM) convert digital values to their equivalent voltage output signals. The basic AOM comprises one or two digital-to-analog converters (DAC) with 10-, 12-, and 14-bit resolution and control and addressing logic for up to eight analog output channels. The model 70-8200 A/D power supply is a prerequisite.

The AOM system can be expanded to eight DACs per device address to a maximum of 64. All DACs are initialized to zero-volt output by a SYSTEM RESET. They are selected for output under program control, and the computer outputs data to the DAC without reselection for each word until another DAC is selected or the system is reinitialized.

The control and addressing logic implements the transfer of the converted data under program control or under the direction of the optional BIC.

The specifications of the AOM are given in table 7-41, and table 7-42 lists the AOM instructions.

Table 7-41. Analog Output Module Specifications

Parameter	Description
Resolution	10-, 12-, or 14-bits
Conversion Accuracy	10-bits: ± 0.05 percent 12-bits: ± 0.012 percent 14-bits: ± 0.003 percent
Voltage Range	10-bits: ± 10V dc at ± 5 milliamperes 12-bits: ± 10V dc at ± 10 milliamperes 14-bits: ± 10V dc at ± 10 milliamperes
Temperature Coefficient	± 0.1 LSB per degree C
Warm-Up Time	Essentially zero
Slew Rate	0.5V per microsecond, and 6V per microsecond
Settling Time	20 microseconds to stated accuracy
Adjustments	10-bits: full scale and zero 12-bits: full scale, zero, and MSB 14-bits: full scale, zero, and three MSB
Standard Device Address	050 through 053
Size	One etched circuit board
Input Power	+ 5V dc ± 5 percent at 500 milliamperes + 15V dc ± 0.1 percent at 50 milliamperes - 15V dc ± 0.1 percent at 50 milliamperes

Table 7-42. Analog Output Module Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0x5y	100x5y	Select channel x
	Data Transfer	
OAR 05y	103x5y	Transfer A register contents to selected channel
	Program Sense	
SEN 0x5y	101x5y	Status of line x

where x = channel (00 through 07), and y = last digit of the device address.

Data Communications Equipment

The term *data communications* implies the transmission of digital information between two distant points. As presently used, the term also implies that two or more different transmission techniques are involved and that the data, therefore, exist in different forms during its transmission. The essential role of the equipment and interfaces described below is to act as translators and transmitters. A data communications interface must be able to handle data rates that can range from the hunt-and-peck keyboard typing of a novice to the thousands of bits per second being transferred over high-speed telephone lines. It must be equally adaptable to the varying codes generated by Teletype, paper tape, magnetic tape, and punched card sources, all without losing a single bit of information in the process.

Three modes of communication are available. Depending on the application, they can apply to all types of communications services.

A *full-duplex* communications line can simultaneously carry information in both directions. A *half-duplex* line can carry information in both directions, but not simultaneously. A *simplex* line is designed to carry data in only one direction; it is either transmitting or receiving.

In addition, there are two methods of transmitting data on a communications line. *Synchronous transmission* provides the highest data transfer rate for a given line capacity, but this is balanced by the increased costs and complexity of the transmitting and receiving equipment. *Asynchronous transmission* is slower, but makes use of simpler equipment. In synchronous transmissions, large blocks of characters are transmitted as a continuous series of bits; in asynchronous transmissions, each character is transmitted as a unit with distinguishing start and stop signals identifying the beginning and end of the character.

Another important phase of data communications is the ability to *multiplex* several communications lines. Multiplexing is the process of transferring data from several storage devices operating at relatively low transfer rates to one storage device (e.g., a time-shared computer) operating at a high transfer rate. It is, therefore, the simultaneous transmission of a number of different messages over a single circuit.

Dataset Controllers (Models 70-540X, 70-550X)

The dataset controllers (models 70-540x and 70-550x) interface with the computer and modems that are compatible with standard datasets. These controllers can operate in half- or full-duplex mode. They detect and establish input sychronization and switch to word mode for data transfers.

Data are transferred in one of three ways:

a. The controller can be connected to the BIC for operations requiring minimum program intervention. BIC can be connected for half-duplex operation to input or output data. In full-duplex operation, I/O functions can be connected to the BIC and data flow can be controlled by the stored program.

b. Full- or half-duplex operation can be completely controlled by the program.

c. Inputting data to the PIM provides interrupt-controlled transfers.

The controller functions are:

a. Receiving data from the modem and transferring it to the computer.

b. Transmitting data received from the computer to the modem.

c. Simultaneously receiving/transmitting data in both directions.

Models 70-5401, 70-5402

The models 70-5401 and 70-5402 dataset controller specifications are given in table 7-43 and table 7-44 lists their instructions.

Table 7-43. Models 70-5401 and 70-5402 Dataset Controller Specifications

Parameter	Description
Organization	Contains control logic, timing circuits, read and write buffers, and I/O drivers and receivers
Control Capability	
70-5401	One Bell 103 or 202, or equivalent, dataset
70-5402	One or two Bell 103 or 202, or equivalent, dataset
I/O Capability	One external control, eight data transfer, and six program sense instructions
Operational Modes	Full- and half-duplex, asynchronous, automatic answer
Transfer Rate	Up to 9600 baud, hardware selectable
Standard Device Address	070 through 073
Size	
70-5401	One etched circuit board
70-5402	Two etched circuit boards
Interconnection	Interfaces with dataset via a 10-foot (3 m) cable; interfaces with computer and BIC via the backplane wiring
Input Power	+5V dc at 1.2 ampere

Table 7-44. Models 70-5401 and 70-5402 Dataset Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0471	100471	Reset request to send
	Data Transfer	
OAR 071	103171	Output A register to controller
OBR 071	103271	Output B register to controller
OME 071	103071	Output memory to controller
INA 071	102171	Input status and data to A register
INB 071	102271	Input status and data to B register
IME 071	102071	Input status and data to memory
CIA 071	102571	Input status and data to cleared A register
CIB 071	102671	Input status and data to cleared B register
	Program Sense	
SEN 071	101071	Sense status change
SEN 0171	101171	Sense output ready
SEN 0271	101271	Sense input ready
SEN 0371	101371	Sense carrier on
SEN 0471	101471	Sense clear to send
SEN 0771	101771	Sense data set ready

Models 70-5501, 70-5502, 70-5503, 70-5504

Specifications for the models 70-5501, 70-5502, 70-5503, and 70-5504 controllers are given in table 7-45 and instructions for models 70-5501 and 70-5502 are listed in table 7-46. Instructions for models 70-5503 and 70-5504 are listed in table 7-47.

Table 7-45. Models 70-5501, -5502, -5503, and -5504 Dataset Controller Specifications

Parameter	Description
Organization	Contains timing circuits, read and write buffers, modem control register, control logic, and I/O drivers and receivers
Control Capability	
70-5501	One Bell 201 or 208, or equivalent, dataset
70-5502	One or two Bell 201 or 208, or equivalent, datasets
70-5503	One Bell 201 or 208, or equivalent, dataset
70-5504	One Bell 201 or 208, or equivalent, dataset
I/O Capability	
70-5501,-5502	Six external control, eight data transfer, and four program sense instructions
70-5503,-5504	Eight external control, eight data transfer, and eight program sense instructions
Operational Modes	Full- or half-duplex, synchronous, automatic answer
Transfer Rate	Up to 50,000 baud
Standard Device Address	070 through 073
Size	One wire-wrap circuit board
Interconnection	Interfaces with dataset via a 20-foot (6 m) cable; interfaces with computer and BIC via the backplane wiring
Input Power	+5V dc at 0.7 ampere

Table 7-46. Models 70-5501 and 70-5502 Dataset Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 071	100071	Go to search
EXC 0171	100171	Connect write buffer to BIC
EXC 0271	100271	Connect read buffer to BIC
EXC 0471	100471	Turn on request to send
EXC 0571	100571	Turn off request to send
EXC 0671	100671	Go to character format
	Data Transfers	
IME 071	102071	Transfer read buffer to 8 LSB of memory
INA 0171	102171	Transfer read buffer to 8 LSB of A register
INB 0271	102271	Transfer read buffer to 8 LSB of B register
CIA 0571	102571	Transfer read buffer to 8 LSB of A register cleared
CIB 0671	102671	Transfer read buffer to 8 LSB of B register cleared
OME 071	103071	Transfer memory 8 LSB to write buffer
OAR 0171	103171	Transfer A register 8 LSB to write buffer
OBR 0271	103271	Transfer B register 8 LSB to write buffer
	Program Sense	
SEN 0171	101171	Write buffer empty
SEN 0271	101271	Read buffer full
SEN 0371	101371	Carrier on
SEN 0471	101471	Clear to send

Table 7-47. Models 70-5503 and 70-5504 Dataset Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 0xx	1000xx	Reset request to send
EXC 01xx	1001xx	Initialize input
EXC 02xx	1002xx	Initialize output
EXC 03xx	1003xx	Enable request to send
EXC 04xx	1004xx	Reset error storage
EXC 05xx	1005xx	Enable BIC to input
EXC 06xx	1006xx	Enable BIC to output
EXC 07xx	1007xx	Enable transparent mode
	Data Transfer	
OME 0xx	1030xx	Output memory to M.C. output buffer
OAR 01xx	1031xx	Output A register to M.C. output buffer
OBR 02xx	1032xx	Output B register to M.C. output buffer
IME 0xx	1020xx	Input M.C. input buffer to memory
INA 01xx	1021xx	Input M.C. input buffer to A register
INB 02xx	1022xx	Input M.C. input buffer to B register
CIA 05xx	1025xx	Input M.C. input buffer to cleared A register
CIB 06xx	1026xx	Input M.C. input buffer to cleared B register
	Program Sense	
SEN 0xx	1010xx	Channel disabled
SEN 01xx	1011xx	Input buffer ready
SEN 02xx	1012xx	Output buffer ready
SEN 03xx	1013xx	Carrier on
SEN 04xx	1014xx	Error (any error)
SEN 05xx	1015xx	Parity error
SEN 06xx	1016xx	Overflow error/underflow error
SEN 07xx	1017xx	Control word

Note: xx = 70 for first modem controller
xx = 77 for eighth modem controller

Binary Synchronous Communications

Controllers (Models 70-5515, 70-5516)

The Binary Synchronous Communications Controllers (models 70-5515 and 70-5516) are designed to interface the Varian 70 series computers to Bell Telephone 201 and 208 type modems (or equivalent). A wide band option (70-5801) is also available for 1 or 2 lines in any combination.

The controllers act as a modem-to-computer interface, providing all necessary timing, decoding, mode selection, and character assembly and disassembly.

The binary synchronous communications controller specifications are given in table 7-48 and the instructions are listed in table 7-49.

Table 7-48. Binary Synchronous Communications Controller Specifications

Basic Clock Frequency	9.8304 MHz
Computer interface;	
Device Code	070 (standard)
Interrupts	Six interrupts generated by the Controller (no PIM)
DMA (operation)	Accesses memory using table stored in memory for DMA control (no BIC)
Instructions	Six external-control; two transfer
Priority assignment	Interrupt and DMA priority determined by position on the I/O priority chain, communication-line priority by line address
BSC Modem Controller package	Models 70-5506 and 70-5516 Four printed-circuit boards.
Modem Interface	EIA RS232C Interface (Optional Coax for Wideband)
Modes of Operation	a. BSC EBCDIC b. BSC USASCII nontransparent c. BSC USASCII with transparency d. Transmit test mode e. Receive test mode
Options	
303 Interface	Bell 303 series modem interface (requires 70-5801)
Data Set Cables	RS232 Type-70-5903 300 Series Type-70-5904
Power Consumption	
Single line	+12V dc at 50 mA -12V dc at 50 mA +5V dc at -6.0 amps
Dual line	+5V dc at 8.5 amps +12V dc at -100 mA -12V dc at 100 mA
Prerequisites	Installs in an I/O communication chassis 70-5913 or 70-5914. Modem interfaces via a 44 pin edge connector (mating connector supplied).

Table 7-49. Binary Synchronous Communications Controller Instructions

Octal	Mnemonic	Description
100070	EXC 070	Programmed system reset that clears the BSC Controller
100170	EXC 0170	Aborts current sequence and returns controller to scanning mode, but does not disable interrupts.
100270	EXC 0270	Enables the six BSC Control interrupts.
100470	EXC 0470	Disables the six BSC Control interrupts.
100570	EXC 0570	Permits computer to request use of the controller bus for setup; the controller generates a control interrupt upon completion of current operation.
100670	EXC 0670	Permits computer to request use of the controller bus for reading status; the controller generates a control interrupt upon completion of current operation.
100244	EXC 0244	BSC Controller also responds to this general system interrupt-enabling instruction.
100444	EXC 0444	BSC Controller also responds to this general system interrupt-disabling instruction.
102070	IME 070	Transfers a 16-bit character from the BSC Controller to memory. The six least-significant bits come from the scan counter and the remainder from the interface buffer.

Table 7-49. Binary Synchronous Communications Controller Instructions *(continued)*

Octal	Mnemonic	Description
102170	INA 070	Transfers a 16-bit character from the BSC Controller to the A register. The six least-significant bits come from the scan counter and the remainder from the interface buffer.
102270	INB 070	Transfers a 16-bit character from the BSC Controller to the B register. The six least-significant bits come from the scan counter and the remainder from the interface buffer.
102570	CIA 070	Clears A register, then transfer a 16-bit character from the BSC Controller, where the six least-significant bits come from the scan counter and the remainder from the interface buffer.
102670	CIB 070	Clears B register, then transfer a 16-bit character from the BSC Controller, where the six least-significant bits come from the scan counter and the remainder from the interface buffer.
103170	OAR 070	Transfers a 16-bit character to the BSC Controller from the A register. The six least-significant bits go to the scan counter and the remainder to the interface buffer.
103270	OBR 070	Transfers a 16-bit character to the BSC Controller from the B register. The six least-significant bits go to the scan counter and the remainder to the interface buffer.
103370	OME 070	Transfers a 16-bit character to the BSC Controller from memory. The six least-significant bits go to the scan counter and the remainder to the interface buffer.

Automatic Call Unit Controller

The automatic call unit controller (model 70-5701) interfaces between the computer and a Western Electric 801 Automatic Call Unit (ACU) to permit the computer to initiate the automatic dialing of any telephone number in a communications network. When such a call is acknowledged, the controller switches the line to a dataset for fully automatic transmission of data. The ACU thus performs all the functions of an attendant in originating a data call and transmission.

Telephone numbers to be called are included in the program stored in the computer and transferred to the ACU via the controller in four-bit, parallel configurations.

Operation of the ACU controller is under program control or under the direction of the optional BIC.

The ACU controller specifications are given in table 7-50, and table 7-51 lists its instructions.

Table 7-50. ACU Controller Specifications

Parameter	Description
Organization	Contains a four-bit output buffer, instruction decoder, and I/O drivers and receivers
Control Capability	One Western Electric 801 ACU
I/O Capability	Four external control, three transfer, and five program sense instructions
Data Format	Four-bit-parallel; output meets RS232 specifications
Standard Device Address	070 through 073
Size	One etched circuit board
Interconnection	Interfaces with ACU via a 25-foot (7.62 m) cable; interfaces with computer and BIC via a 122-terminal connector
Input Power	+5V dc at 180 milliamperes −12V dc at 9 milliamperes +12V dc at 48 milliamperes

Table 7-51. ACU Controller Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 007x	10007x	Initialize
EXC 017x	10017x	Enable call request
EXC 027x	10027x	Disable call request
EXC 047x	10047x	Reset digit present (DPR) flip-flog
	Data Transfer	
OAR 07x	10317x	Transfer A register contents to controller output buffer
OBR 07x	10327x	Transfer B register contents to controller output buffer
OME 07x	10307x	Transfer memory contents to controller output buffer
	Program Sense	
SEN 07x	10107x	Dial line busy
SEN 017x	10117x	Power indication
SEN 027x	10127x	Abandon call retry
SEN 037x	10137x	Call-origin status
SEN 047x	10147x	Next digit present

where x = last digit of the device address.

Data Communications Multiplexor and Line Adapter

The Varian model 70-52xx data communications multiplexor (DCM) and its associated line adapters (LAD), (model 70-53xx) provide a communications interface for Varian 70 series processors with 620-type I/O. Each DCM can serve as an interface for up to 64 terminals or communication lines, in any combination of groups of four, operating concurrently in either asynchronous or sychronous transmission modes. Multiple DCMs in a system can accommodate virtually any number of terminal devices.

The DCM provides the communication link between the LADs and the computer via programmed I/O and DMA. The DCM logic handles interrupts and trap requests (DMA) internally, and therefore requires neither an external priority-interrupt module (PIM) nor a buffer interlace controller (BIC). For DMA operation, the DCM uses a control table stored in the computer memory. This table contains control characters, output and input block-lengths, output and input buffer-locations, and control information for each line. The DCM accesses the table through the computer's DMA with individual line addresses as pointers.

The individual lines are set up under program control to begin a transfer-in, transfer-out, or both (full-duplex). Once a line is set up, the DCM inputs or outputs data to or from the line on a demand basis using the DMA port of the computer. The program is interrupted only for a line error, control-character detection, or upon completion of the transfer. The data are automatically packed or unpacked by the DCM hardware. The character-assembly, disassembly, parity-generation, parity-checking, modem-control, and buffering tasks are handled at the LAD level.

The DCM is available in six models, according to capacity and system configuration:

70-5201, 16 line
70-5202, 32 line
70-5203, 64 line
70-5211, 16 line for memory map systems only
70-5212, 32 line for memory map systems only
70-5213, 64 line for memory map systems only

The LAD is available in five models:

70-5301, Asynchronous modem
70-5302, RS232 interface (direct connect)
70-5303, Current-loop interface (direct connect)
70-5304, Relay interface (direct connect)
70-5305, Synchronous modem

Specifications for the DCMs are listed in table 7-52 and table 7-53 lists the specifications for the LADs. The instructions for the DCM are listed in table 7-54.

Table 7-52. Specifications for the DCM

Parameter	Description
Basic Clock Frequency	9.8304 MHz
Line-Scanning Rate	Up to 614.4 kHz
Computer Interface:	
Device Code	070 (standard)
Interrupts	Six interrupts generated by the DCM (no PIM)
DMA (operation)	Accesses memory using table stored in memory for DMA control (no BIC)
Instructions	Six external control, two transfer
Priority Assignment	Interrupt and DMA priority determined by position on the I/O priority chain, communication-line priority by line address
Multiplexor Bus	Provides up to six asynchronous data rates
MU Package	Two printed-circuit boards
Power Consumption (MU)	+5V dc at 3.5A
Interconnection	Special I/O backplane that fits a 620 or V70 series I/O expansion chassis

Table 7-53. Specifications for the LADs

Parameter	Description
Model 70-5301	
Number of Lines/Board	Four
Transmission Type	Serial asynchronous (normal or mirror-image)
Modem Interface	Interface circuits conform to EIA RS232C (maximum cable length is 15 meters or 50-feet)
Character Length	5, 6, 7, or 8 (hardware selectable)
Number of Stop Bits	1 or 2 (hardware selectable)
Parity Generation and Checking	Odd, even, or none (hardware selectable)
Buffering	Each line character buffered on input and output
Maximum Bit Rate	9,600 bits per second
Bit Rate Selection	Each line is assigned two hardware selectable speeds; these two speeds are software selectable
Error Reporting	Parity errors, line breaks (or framing errors), and overflow errors
Package	One PC board with approximately 75 ICs
Power Requirements	+5V dc at 1.2A, +12V dc at 100 mA, −12V dc at 150 mA
Operational Environment	Temperature 0 to 50 degrees C; humidity 0 to 90 percent (without condensation)
Interconnection	Plugs into DCM backplane, interface with up to four modems via two 44-pin connectors

Table 7-53. Specifications for the LADs *(continued)*

Parameter	Description
Models 70-5302,-5303, and -5304	
Number of Lines/Board	Four
Transmission Type	Serial asynchronous
Line Terminal Interface	EIA RS232C, current-loop (70-5303) or relay (70-5304)
Character Length	5, 6, 7, or 8 bits (hardware selectable)
Number of Stop Bits	1 or 2 (hardware selectable)
Parity Generation and Checking	Odd, even, or none (hardware selectable)
Buffering	Each line character buffered on input and output
Maximum Bit Rate (Models 70-5302,-5303)	
Cable Length:	
Up to 1,000 feet (300 meters)	9,600 bps
1,000 to 2,000 feet (300 to 600 meters)	4,800 bps
2,000 to 3,500 feet 600 meters to 1 km.)	1,800 bps
3,500 to 5,000 feet (1 to 1.5 km.)	900 bps
5,000 to 10,000 feet (1.5 to 3 km.)	300 bps
Maximum Bit Rate (Model 70-5304)	45 - 300 bps
Error Reporting	Parity errors, line breaks (or framing error), and overflow errors
Package	One PC board with approximately 75 ICs

Table 7-53. Specifications for the LADs *(continued)*

Parameter	Description
Power Requirements	
Model 70-5302	+5V dc at 1.2A, +12V dc at 100 mA, −12V dc at 100 mA
Model 70-5303	+5V dc at 1.2A, −12V dc at 50 mA (exclusive of user-supplied loop-current power source)
Model 70-5304	+5V dc at 1.2A, −12V dc at 50 mA (exclusive of user-supplied loop-current power source)
Operational Environment	Temperature 0 to 50 degrees C; humidity 0 to 90 percent (without condensation)
Interconnection	Plugs into DCM backplane, interface with up to four terminals via two 44-pin connectors
Model 70-5305	
Number of Lines/Board	Four
Transmission Type	Serial synchronous
Modem Interface	Interface circuits conform to EIA RS232C (maximum cable length 15 meters or 50-feet)
Character Length	5, 6, 7, or 8 (hardware selectable)
Parity Generation and Checking	Odd, even, or none (hardware selectable)
Buffering	Each line character buffered on input and output, and there are buffers for an input and an output synchronization character
Maximum Bit Rate	20,000 bits per second

Table 7-53. Specifications for the LADs *(continued)*

Parameter	Description
Bit Rate Selection	Transmission and receipt clocks provided by the modem
Error Reporting	Parity errors, overflow errors, and overrun errors
Package	One PC board with approximately 75 ICs
Power Requirements	+5V dc at 1.5A, +12V dc at 100 mA, −12V dc at 150 mA
Operational Environment	Temperature 0 to 50 degrees C; humidity 0 to 90 percent (without condensation)
Interconnection	Plugs into DCM backplane, interfaces with up to four modems via two 44-pin connectors

Table 7-54. DCM Instructions

Mnemonic	Octal Code	Description
	External Control	
EXC 070	100070	Initialize
EXC 0170	100170	Clear control logic
EXC 0270	100270	Enable DCM interrupts
EXC 0470	100470	Disable DCM interrupts
EXC 0570	100570	Request control (write)
EXC 0670	100670	Request control (read)
EXC 0244	100244	Enable interrupts (common)
EXC 0444	100444	Disable interrupts (common)
	Data Transfer	
IME 070	102070	Input data from DCM to memory
INA 070	102170	Input data from DCM to A register
INB 070	102270	Input data from DCM to B register
CIA 070	102570	Clear A register and input data from DCM
CIB 070	102670	Clear B register and input data from DCM
OAR 070	103170	Output data from A register to DCM
OBR 070	103270	Output data from B register to DCM
OME 070	103370	Output data from memory to DCM

Binary Synchronous Communications Multiplexor (Model 70-5712)

The binary synchronous communications multiplexor (model 70-5712) and an associated line adapter (LAD, model 70-5306) provide a binary synchronous communications interface for the Varian 73 system computer. Each binary synchronous communications multiplexor (BSCM) can serve as an interface for up to eight LAD's.

Operations of the BSCM, once initiated, continues independent of the main program by virtue of the table driven Direct Memory Access (DMA) facility. Interruptions of the program only occur at the completion of message transfers or when line or terminal attention is required. Throughout the operation, internal BSCM control is relegated to a self-contained microprogram which directs all multiplexor, Line Adapter (LAD), and DMA related sequence.

Throughput for a BSCM can range up to 50,000 bytes per second. Programming overhead associated with this exceedingly high capability is minimized due to message oriented interrupts, DMA table driven control and hardware implementation of BSC block checking, and control sequence in the LAD.

Specifications for the BSCM are listed in table 7-55 and table 7-56 lists the specifications for the LAD. Instructions for the BSCM are listed in table 7-57.

Table 7-55. Binary Synchronous Communications Specifications Multiplexor

Basic Clock Frequency	9.8304 MHz
Line-Scanning Rate	614.4 kHz (1.63 usec)
Computer interface:	
Device Code	070 (standard)
Interrupts	Six interrupts generated by the BSCM (no PIM)
DMA (operation)	Accesses memory using table stored in memory for DMA control (no BIC)
Instructions	Six external-control; two transfer
Priority assignment	Interrupt and DMA priority determined by position on the I/O priority chain, communication-line priority by line address
BSC Multiplexor package	Two printed-circuit boards
Power consumption multiplexor	+5V dc at 3.5A
Interconnection	Installs in communications chassis 70-5910 or 70-5911

Table 7-56. Specifications for LAD (Model 70-5306)

Number of lines/ board	one
Transmission Type	Serial Synchronous
Modem Interface	EIA RS232C Interface (optional coax for wide band option)
Modes of Operation	a. BSC EBCDIC (transparent, non-transparent) b. BSC USASCII non-transparent c. BSC USASCII with transparency d. Transmit test mode e. Receive test mode
Maximum Bit Rate	20,000 bits per second (RS232C) 50,000 bits per second with optional modem interface
Error Reporting	Format errors, BCC errors overrun, underrun, receive time out
Options	
303 Interface	Bell 303 series modem interface (70-5801)
Data Set	RS232C type– 70-5903
Cables	300 series type– 70-5904
Power Consumption	+5V at 2.5 amps +12V at 50 mA –12V at 50 mA
Environment	Temperature 0 to 50 degrees C (32 to 122 degrees F), 0 to 90 percent relative humidity without condensation
Interconnection	Plugs into BSCM backplane. Interfaces to modem via a 44 pin edge connector (Mating connector supplied)
Prerequisites	Multiplexor 70-5712, Communications chassis 70-5910 or 70-5911

Table 7-57. Binary Synchronous Communications Multiplexor Instructions

Octal	Mnemonic	Description
100170	EXC 0170	Aborts current sequence and returns multiplexor to scanning mode, but does not disable interrupts
100270	EXC 0270	Enables the six BSCM interrupts
100470	EXC 0470	Disables the six BSCM interrupts
100570	EXC 0570	Permits computer to request use of multiplexor bus for LAD setup; the multiplexor generates a control interrupt upon completion of current operation
100670	EXC 0670	Permits computer to request use of multiplexor bus for reading LAD status; the multiplexor generates a control interrupt upon completion of current operation
100244	EXC 0244	BSCM also responds to this general system interrupt-enabling instruction
100444	EXC 0444	BSCM also responds to this general system interrupt-disabling instruction
102070	IME 070	Transfers a 16-bit character from the BSCM where the six least-significant bits come from the scan counter and the remainder from the interface buffer
102170	INA 070	Transfers a 16-bit character from the BSCM to the A register. The six least-significant bits come from the scan counter and the remainder from the interface buffer
102270	INB 070	Transfers a 16-bit character from the BSCM to the B register. The six least-significant bits come from the scan counter and the remainder from the interface buffer

Table 7-57. Binary Synchronous Communications Multiplexor Instructions
(continued)

Octal	Mnemonic	Description
102570	CIA 070	Clears A register, then transfers a 16-bit character from the BSCM where the six least-significant bits come from the scan counter and the remainder from the interface buffer
102670	CIB 070	Clears B register, then transfers a 16-bit character from the BSCM where the six least-signficant bits come from the scan counter and the remainder from the interface buffer
103170	OAR 070	Transfers a 16-bit character from the A register to the BSCM. The six least-significant bits go to the scan counter and the remainder to the interface buffer
103270	OBR 070	Transfers a 16-bit character from the B register to the BSCM. The six least-significant bits go to the scan counter and the remainder to the interface buffer
103370	OME 070	Transfers a 16-bit character from the memory to the BSCM. The six least-significant bits go to the scan counter and the remainder to the interface buffer

Universal Controller

The universal asychronous serial controller (model 70-560x) designed for interfacing any model 70 series system to peripheral devices utilizing any asynchronous serial interface. This asynchronous controller should not be confused with a modem controller as it does not include modem control logic. The controller is intended for "direct connect" (no modem) interfacing of peripheral devices equipped with asynchronous serial interfaces. There are three versions of model 70-560x.

Model Number	Interface Type
70-5601	RS232C
70-5602	20 or 60 mA discrete current loop
70-5603	20 or 60 mA relay contact

Functionally, all three versions are the same. They differ only in their electrical interface characteristics; hence, selection of the proper model will depend upon the electrical interface requirement of the peripheral device to be connected.

Typical applications may be as follows:

- In-house Teletypes equipped with RS232C data set coupler (no modem required) - model 70-5601.
- In-house or remote Teletypes with 20 or 60 mA relay or discrete current loop interface - models 70-5602,-5603.
- CRT displays.
- Computer to computer links.
- Certain serial printers, cassettes, and other serially interfaced devices.
- 83 B 1 equipment.

All versions of the controller are serial, direct connect, character buffered units capable of half- or full-duplex operation. Mqdem control logic is not included. The existing TTY command set is employed.

Operation can be either under program control or in the interrupt mode using the priority interrupt module (PIM) option. Also, automatic block transfers are possible when the operation is in conjunction with the buffer interface controller (BIC) option. Table 7-58 depicts specific characteristics of each version.

All models are tested in a back-to-back manner, i.e., CPU output data is "wrapped-around", input back to the CPU and a comparison made for proper transmission and reception. Test programs for checkout of specific peripherals and connection to the peripheral device are the user's responsibility, except in the cases where Varian Data Machines also provides the peripheral device. When Varian Data Machines supplies the controller and peripheral device, a test routine is provided to exercise the functions of the peripheral device connected to the controller.

The standard 20-foot cable kit furnished with each controller consists of two twisted pairs (24 gauge) in an overall jacket.

A controller mating connector is supplied in the kit. Cables longer than 20 feet are optionally available and must be specified when ordering. The kit also contains material for a test connector.

The controller is packaged on a printed circuit board, has rear edge connectors and occupies one I/O slot in the main-frame or I/O expansion chassis.

The operating distance on model 70-5601 is guaranteed to 50 feet; in most cases, operation up to 100 feet is feasible. Users requiring operating distances beyond 100 feet should use model 70-5602. Maximum operating distance for models 70-5602 and 70-5603 depends upon the serial bit rate. Operation at Teletype rate, for example, can be up to 10,000 feet. The following is a guideline on maximum distance versus bps rate under normal conditions.

Distance up to 1,000 feet
10,000 bps maximum
Distance up to 2,000 feet
4,800 bps maximum
Distance up to 3,500 feet
1,800 bps maximum
Distance up to 5,000 feet
900 bps maximum
Distance up to 10,000 feet
300 bps maximum

Standard baud rates (within the above limits are): 45, 75, 110, 150, 300, 600, 1200, 1800, 2000, 2400, 4800, 9600.

The user should specify the serial rate, character size, stop bits and cable length. When variable data is not specified, the controller will be equipped as follows:

- 70-5601,-5602; 1200 bps; 8 levels; one stop bit; 20-foot open-ended cable.
- 70-5603, 110 bps; 8 level; two stop bits; 20 mA; 20-foot open-ended cable.

When either model 70-5602 or 70-5603 is used, the peripheral device must have a current loop interface. This can be of the relay type or a discrete circuit similar to the circuit on the controller. The user is responsible for providing the "line battery" source for the current loops. The line battery source typically can be obtained from the peripheral device or a small separate power supply. Any voltage from 12V dc to 50V dc can be used.

Table 7-58 list specific characteristics for each version of the controller.

Table 7-58. Universal Controller Specifications

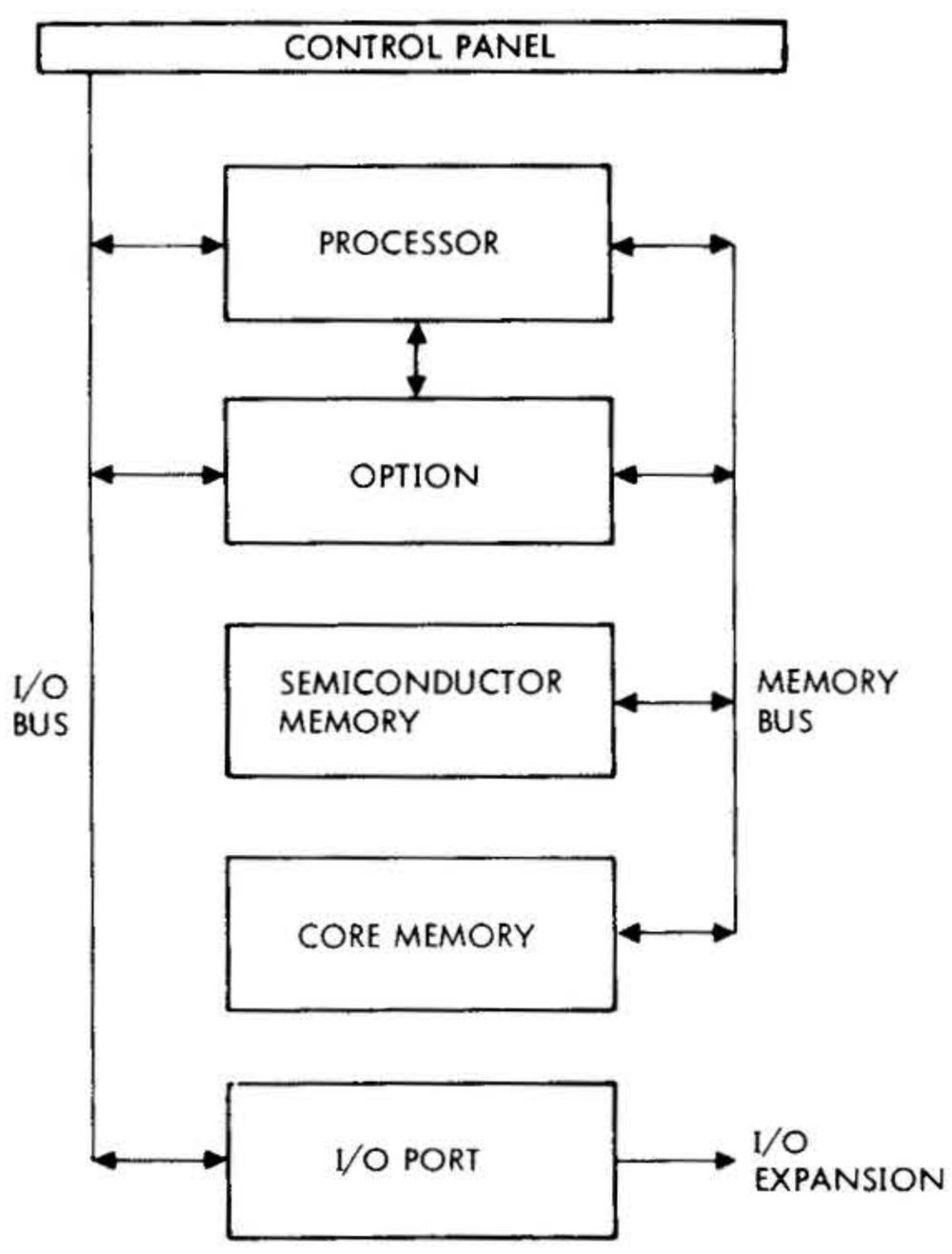
Parameter	Description
Model Number 70-5601	
Serial Rate, bps (Bits per second)	45 to 9,600 bps depending upon cable length
Character Size	5, 6, 7, or 8 level (bits). Parity bit (if any is added as one of the bits - i.e., 7 bits and parity = 8 bits; or 8 bits and no parity = 8 bits
Stop Bit	Standard length = 20 feet, optionally up to 100 feet
Model Number 70-5602	
Serial Rate, bps (Bits per second)	Same as 70-5601
Character Size	Same as 70-5601
Stop Bit	Same as 70-5601
Cable and Distance	Standard length = 20 feet, optionally up to one mile
Miscellaneous	User must provide "line battery" for loop current
Model Number 70-5603	
Serial Rate, bps (Bits per second)	45 to 300 bps. Typically would be 75 or 110 bps. Relays will not operate reliably at higher than 300 bps
Character Size	Same as 70-5601
Stop Bit	Same as 70-5601
Cable and Distance	Same as 70-5602
Miscellaneous	Same as 70-5602

SECTION 8 - SYSTEM CONFIGURATIONS

The following configurations have been selected to illustrate the flexibility built-in to the packaging scheme and to show chassis and cabling layouts of several classes of systems.

Conventional System Configuration

A conventional system is shown by a block diagram (figure 8-1) and a typical system interconnection (figure 8-2). This system utilizes a standard processor, option board, semiconductor and core memories, and an I/O expansion chassis connection.



VTII-1477A

Figure 8-1. Conventional System Block Diagram

Dual Port System

A dual port system is shown by a block diagram (figure 8-3) and a system interconnection (figure 8-4). Dual ports A and B permit system configurations with simultaneous I/O transfers and processor to memory reference operations without interference when common memory modules are not simultaneously accessed on both buses.

The PMA port shown in the block diagram is always on the port B memory bus.

Writable Control Store System

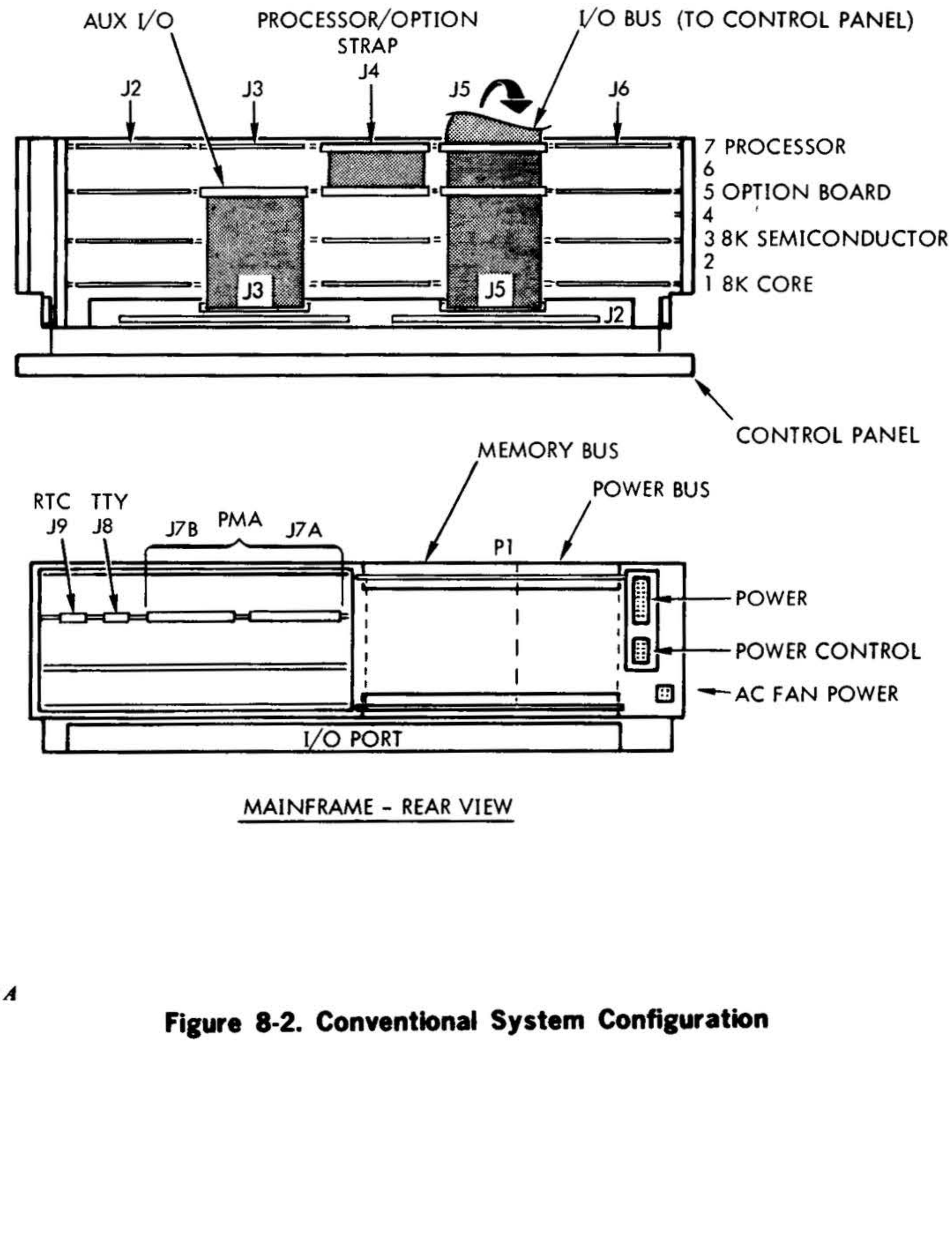
A writable control store (WCS) system is shown by a block diagram (figure 8-5) and a system interconnection (figure 8-6). The WCS inhibits the processor's internal control store and permits application of externally generated test stimuli during troubleshooting procedures. This enables high resolution static or dynamic fault isolation. The WCS is capable of dynamically changing the emulated instruction set and improving performance for specialized functions. Other advantages are:

a. Dynamic alteration of control store contents.

b. Systems can be reconfigured through replacement of faulty hardware functions by firmware routines.

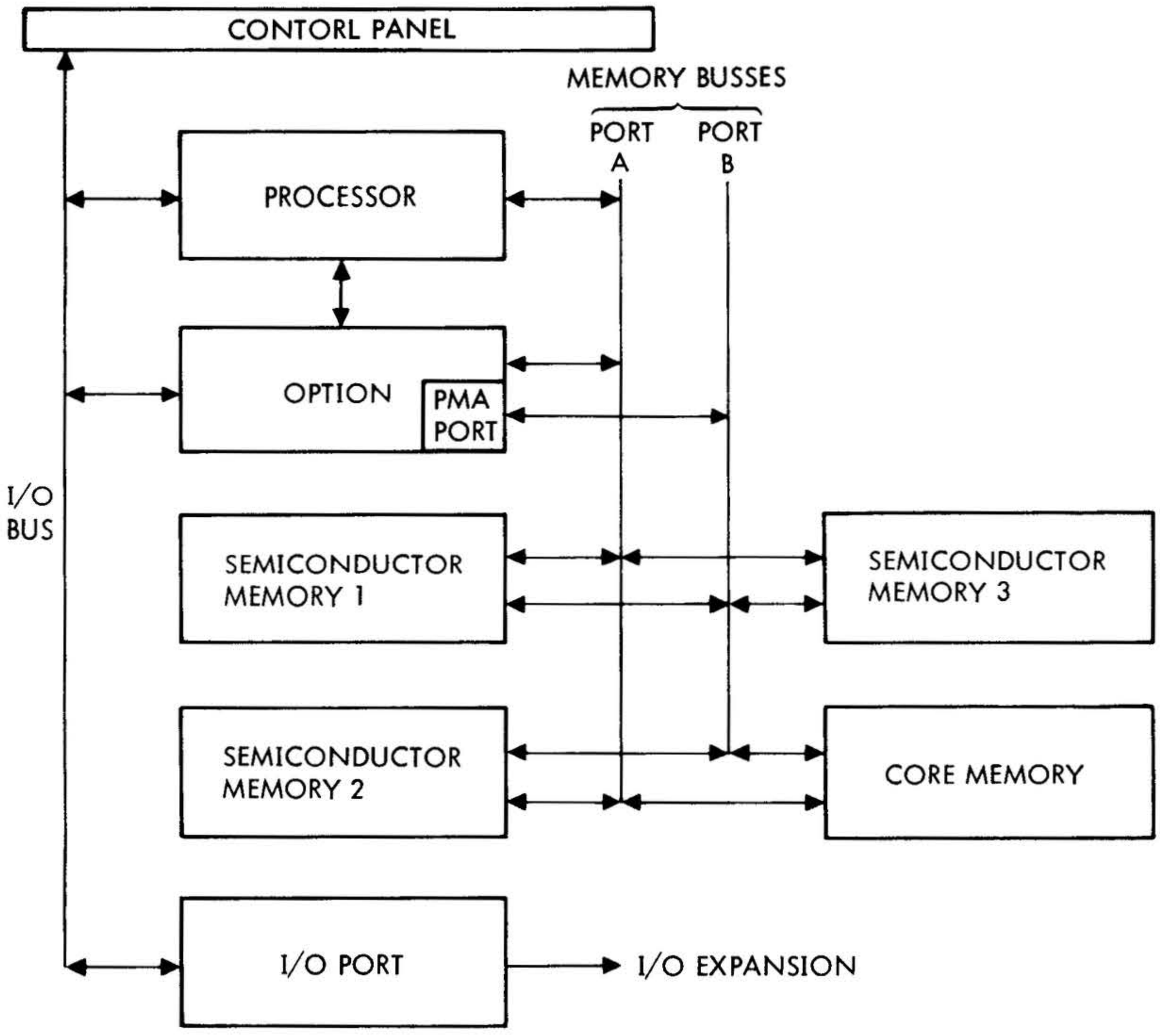
c. Easy incorporation of extensions to existing instruction sets.

The WCS is capable of loading from the memory bus to permit dynamic swapping of control store contents. Control can be by means of privileged I/O instructions.



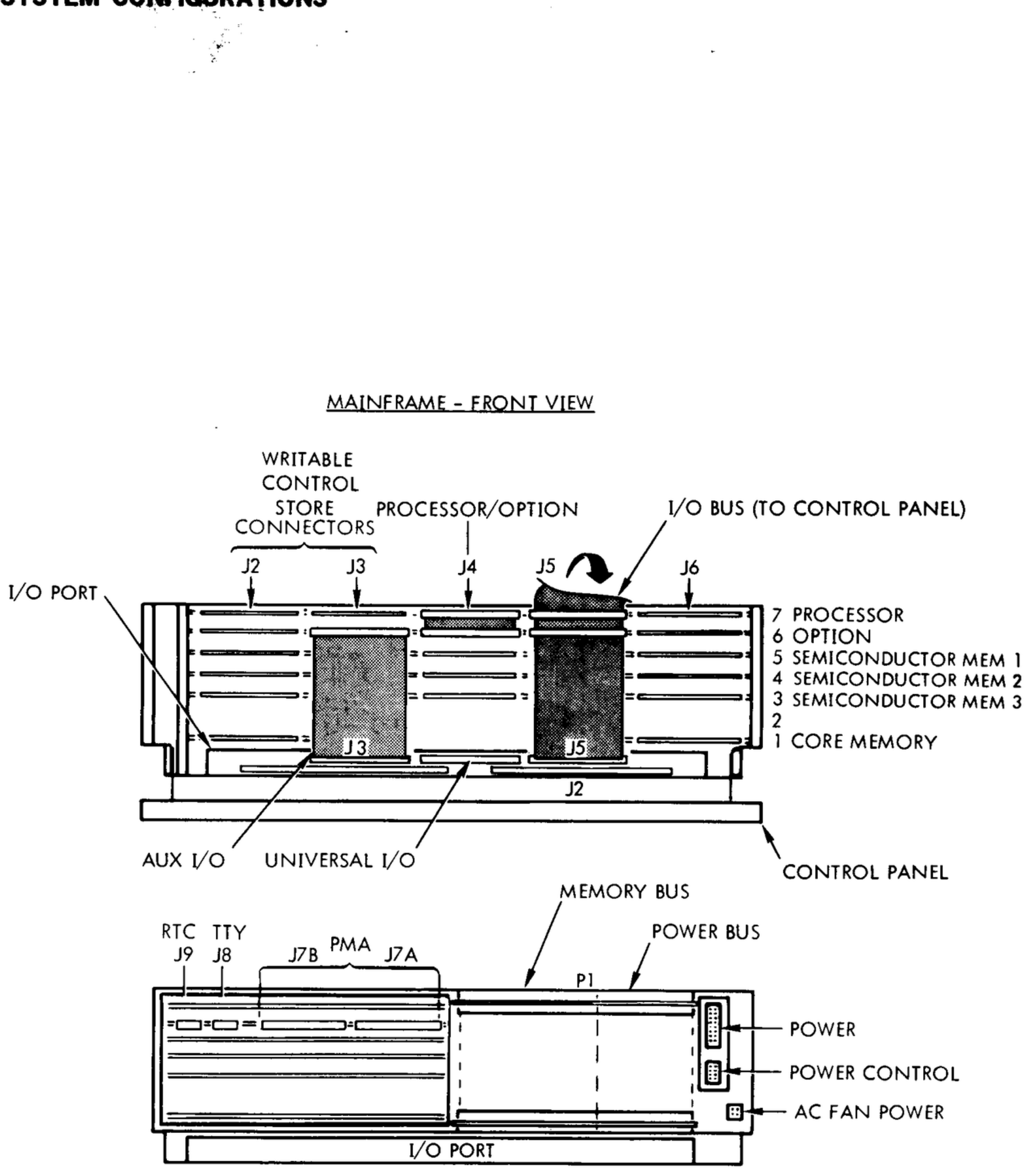
VTI1-1492 A

Figure 8-2. Conventional System Configuration



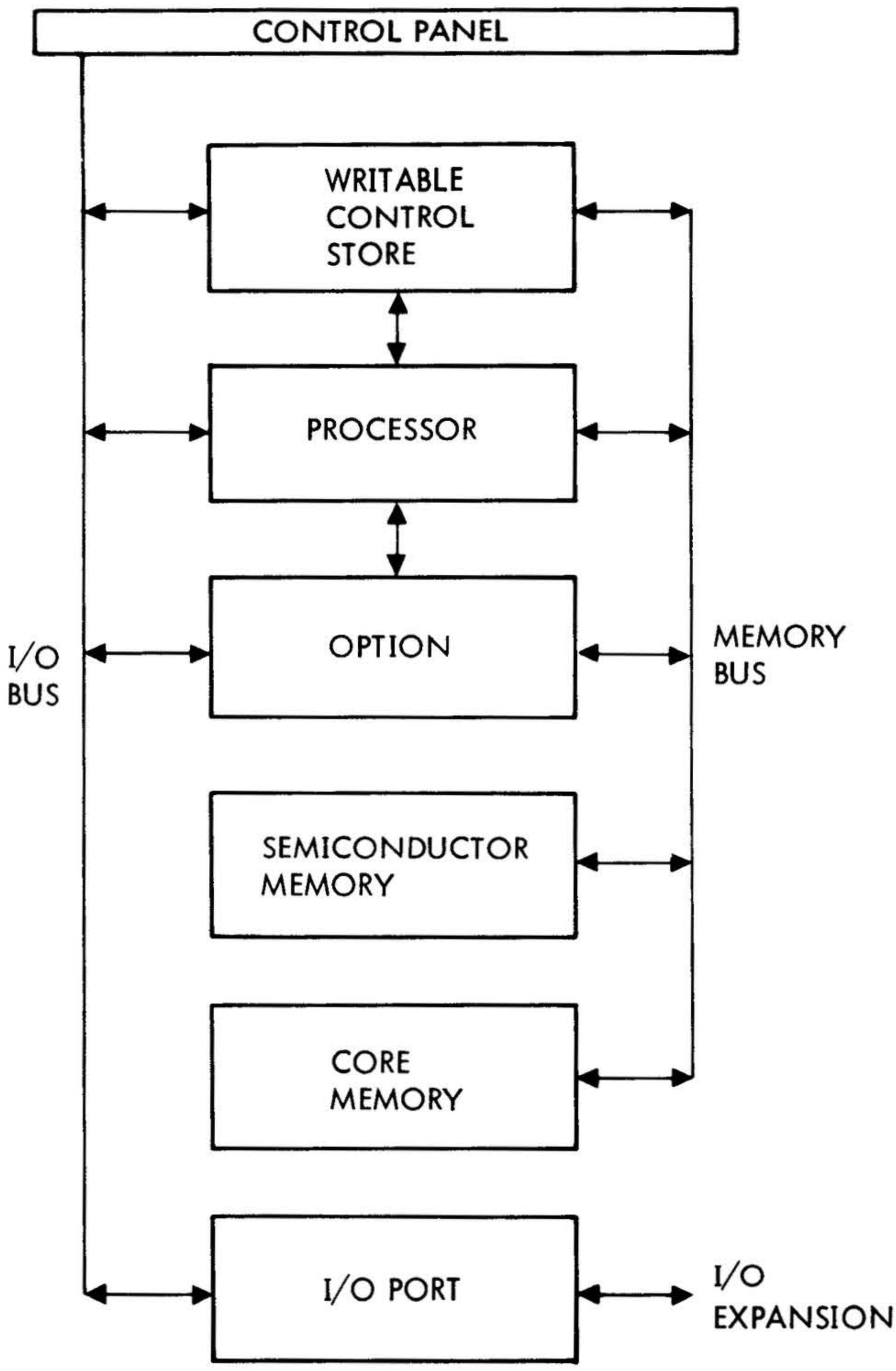
VTI1-1476 A

Figure 8-3. Dual Port System Block Diagram



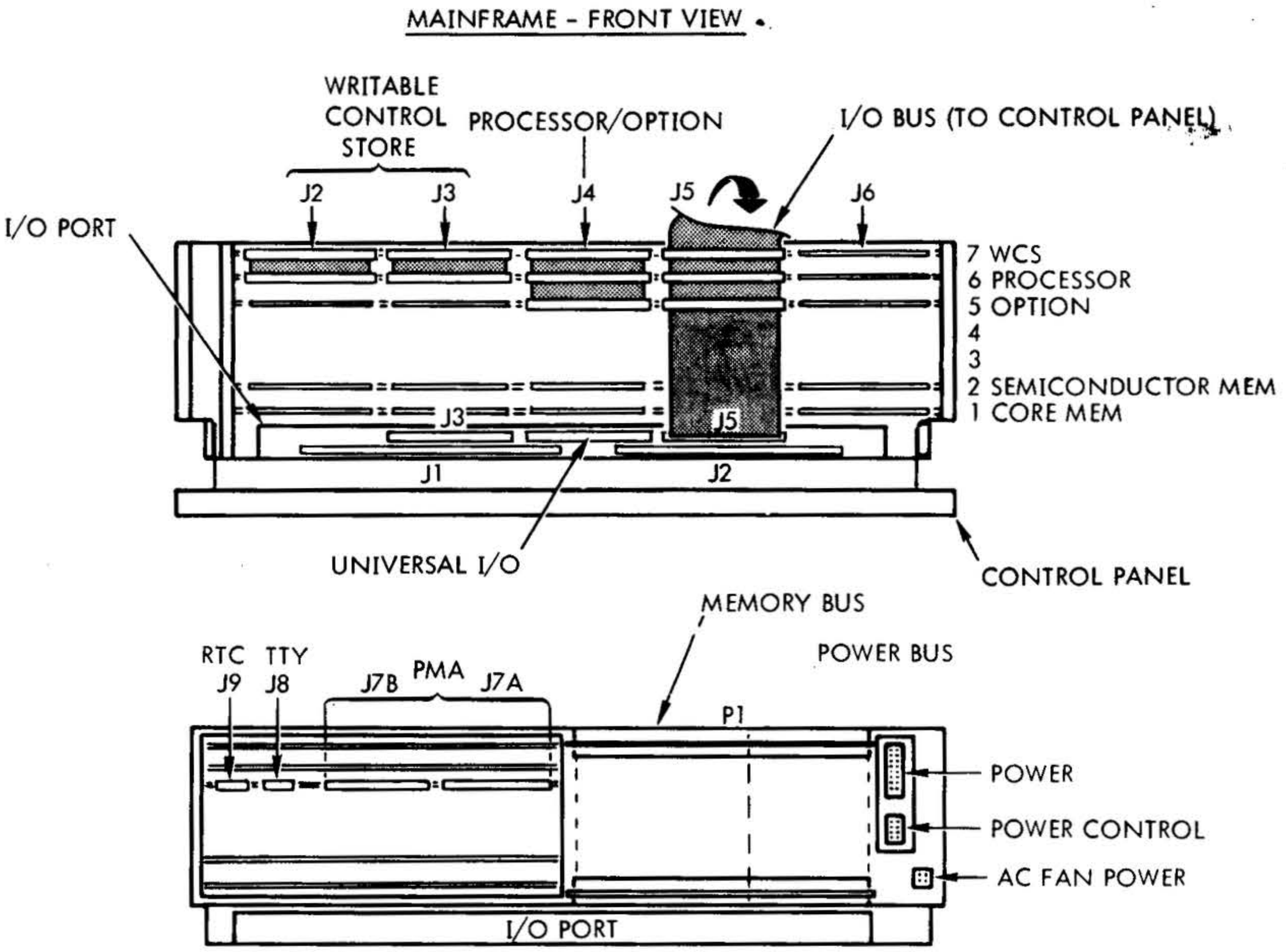
VTI1-1493A

Figure 8-4. Dual Port System Configuration



VTII-1475 A

Figure 8-5. Writable Control Store System Block Diagram



VTI1-1494

Figure 8-6. Writable Control Store System Configuration

SECTION 9 - INSTALLATION

The following sections explain and illustrate the unpacking, inspecting, installing, interconnecting, power up, and testing of a Varian 73 system computer.

Unpacking

A Varian 73 computer is shipped in a single container with component boards mounted in respective slots of the mainframe chassis and held in place with a foam pad. The interconnecting cables are packaged separately for protection. Resilient packing pads are placed between each component board to minimize flexing and vibration. The computer control panel, hinged to the front of the mainframe is protected from damage by cushioning material attached to the front surface with adhesive tape.

To unpack, remove the Varian 73 computer from the shipping container, remove the packing material, check the shipping list to ensure all the equipment has been received, and proceed with the inspection.

Inspection

After unpacking, inspect the mainframe chassis, component boards, connectors, and interconnection cabling for any shipping damage. Make sure component boards slide in and out of tray slots without jamming or binding, all plugs and connectors mate easily, the front panel is not marred or scratched, and that all connector pins are clean. A cursory visual examination of each board should also be made. If damage exists:

a. Notify the transportation company.

b. Notify Varian Data Machines.

c. Save all packing material.

Installation

The 7-inch mainframe chassis slots are numbers 1 to 7 from bottom to top. The console is hinged to the bottom front of the mainframe and swings down, parallel to the floor, for troubleshooting purposes.

The next step is to mount the interconnecting cables as described in the following section.

Interconnection

Table 9-1 lists the interconnecting cable description, connector number, and number of pins in each connector.

Power Up

After making sure all mainframe and peripheral device cables, connectors, system and power supply circuit breakers etc., are correctly mounted, plugged in and turned on (including peripheral devices powered on), turn power key on the control panel to the on position (reference section 1 for power distribution explanations and

Table 9-1. Typical System Cables and Connectors

Description	Connector	Pins
I/O bus connections: I/O Data Standard I/O control (620)	J5	50
Expanded control store	J2, J3	50 each
Auxiliary I/O connector: Priority lines High Speed DMA Control	J3	50
Universal I/O connector: Additional priority lines BIC control signals I/O controller board interconnection (M.T., P.T., Disc, Drum, C.R., etc.)	J4	50
I/O Port:		
Printed-circuit mating connector	J2	122
Printed-circuit edge connector	J3, J4, J5	50 each
Power and memory bus etched card connects all seven slots (17 for 14 inch chassis)	P1	132
Device connectors to I/O controllers	J2	44
Priority Memory Access	J7A, J7B	40 each
Teletype controller	J8	
Real-Time Clock	J9	
Power supply control	J2	8
Power supply	J3	16
Ac fans in	J1	3

illustrations). In the on position, there is ac power to the power supply and both the system and console are fully operational with the STEP indicator illuminated, sense switches off, data display register cleared, display select on REG., and register select cleared.

If manually loading the bootstrap program, the console is ready as it is already in the STEP mode. For an automatic bootstrap load, press the STEP/RUN button to illuminate RUN. For the bootstrap program loading sequence, reference the step-by-step procedures in section 16.

Typical System Integrity Check

Once the program is loaded and running, check to see that the correct information is being output by the peripheral devices, the options are performing correctly, maximum read/write memory functions are working, and that the program runs completely through without error. If any fault conditions exist, troubleshoot the system with the supplied debugging and test routines along with information supplied in the maintenance manuals.

SECTION 10 - OPERATION

Control Panel Switches And Indicators

The Varian 73 control panel (figure 10-1) contains all the switches and indicators needed for computer operation. Except for the POWER switch, which is key operated, all control panel switches are pushbutton type. The functions of the switches and indicators are described in the following paragraphs.

POWER Switch

The POWER switch is a key-operated, four-position switch that controls the ac line voltage to the computer power supply.

In the OFF position, the ac line voltage is removed from the input of the power supply.

In the HOLD position, the ac line voltage is applied to the power supply and all dc voltages are disabled except those required to maintain data in the semiconductor memory. In the HOLD condition, neither the computer nor the control panel are operational.

The CONSOLE DISABLE position is jumper selectable to operate in two modes:

a. All control-panel pushbutton switches are disabled.

b. Only the STEP/RUN and RESET switches are disabled.

The jumper is factory installed on the control-panel circuit board. With the POWER switch in the CONSOLE DISABLE position, the ac line voltage is applied to the power supply, the computer is operational, and the control-panel indicator lights are functional. The key can be removed from the POWER switch in any of the four positions.

To turn off the computer from the CONSOLE DISABLE condition, turn the POWER switch ON, place the computer in the step mode (using STEP/RUN switch), and then turn the POWER switch to either the HOLD position (to maintain data in semiconductor memory) or the OFF position.

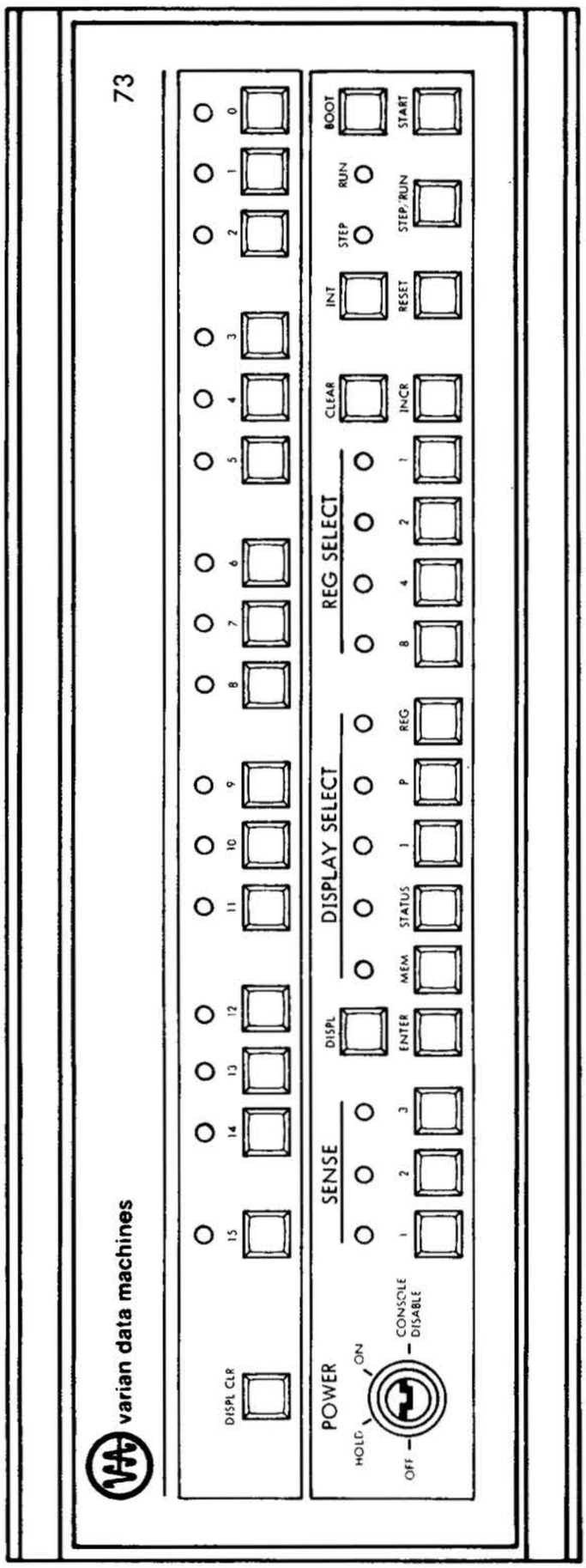
Note:

Before turning on power on systems with semiconductor memory, allow at least 30 seconds of power off time to ensure the refresher logic is operable.

STEP/RUN Switch and STEP and RUN Indicators

The STEP/RUN switch is an alternate-action switch that switches the computer alternately to the step and run modes. In the step mode, the STEP indicator lights; in the run mode, the RUN indicator blinks on and off until the START switch is pressed at which time the RUN indicator is on continuously.

When the computer is in the step mode, pressing the STEP/RUN switch places the computer in the run mode. The STEP indicator goes out and the RUN indicator blinks on and off. When in the run mode,



VTII-1474

Figure 10-1. Varian 73 Control Panel

the computer is ready to be started (by pressing the START switch).

When the computer is in the run mode and has been started, pressing the STEP/RUN switch halts the computer after the current instruction has been executed and the next sequential instruction fetched and loaded into the I register; the RUN indicator goes out and the STEP indicator lights. In addition, a halt instruction (after the computer has been started) halts the computer and causes the RUN indicator to blink.

START Switch

When the computer is in the run mode but has not been started, pressing the START switch starts the program at the location specified by the contents of the program counter. The RUN indicator stops blinking and comes on continuously.

When the computer is in the step mode, pressing the START switch executes the instruction in the instruction register. Then it fetches the next instruction from the memory address specified by the contents of the program counter and loads it in the instruction register. The STEP indicator remains on.

BOOT Switch

The BOOT switch allows the bootstrap program to be loaded into the computer memory automatically. The bootstrap program enables the loading of the binary load/dump program into memory. When the BOOT switch is pressed, the RUN indicator lights. Refer to the program execution portion of this section for bootstrap program loading procedures.

Register-Entry Switches and Register-Display Indicators

The top row of control-panel lights comprise the 16 register-display indicators. They display the contents of the display register. This register, located on the control panel circuit board, can be loaded from the register-entry switches, located on the control panel just below the 16 indicators. In addition, 16-bit data words can be loaded into the display register under control of the DISPLAY SELECT and REG SELECT switches allowing the contents of the various register and memory locations to be visually examined.

The contents of the display register can be cleared (set to zero) by pressing the DISPL CLR switch. This turns off all sixteen display indicators. Any of the sixteen bits can be set by pressing the corresponding register-entry switch. With a bit set, the corresponding display indicator lights. Pressing a register entry switch for a bit already set, has no effect. Bits can only be reset to zero by pressing the DISPL CLR switch. For negative data, the sign bit (bit 15) is set (one).

DISPL and ENTER Switches

The DISPL switch is used with the MEM switch for displaying memory data on the register-display indicators.

The ENTER switch is used with the MEM switch to load data into memory from the register-entry switches.

The procedures for displaying memory data and entering data into memory are described in the manual operations part of this section.

OPERATION

DISPLAY SELECT Switches and Indicators

The five DISPLAY SELECT switches are used to select one of several registers for displaying its contents on the register display indicators and altering them from the register-entry switches. Pressing any DISPLAY SELECT switch cancels any previous selection, turns off the indicator for the previous selections, and lights the indicator for the new selection. The functions for each selector switch are described in the following paragraphs.

The MEM switch selects the memory for data entry or display. For entering data into memory and displaying the contents of memory refer to the manual operations portion of this section.

The STATUS switch displays the status of various signals from the processor. To display the status of these processor signals, perform the following:

a. Turn the POWER switch ON.

b. Place the computer in the step mode.

c. Press STATUS.

The register display indicators now indicate the following:

Bit 15, Key register bit 15 (DCK15+)
Bit 14, Key register bit 14 (DCK14+)
Bit 13, Key register bit 13 (DCK13+)
Bit 12, Key register bit 12 (DCK12+)
Bit 11, Arithmetic and logic unit carry (DCNDC+)
Bit 10, Arithmetic and logic unit sign (DSGN+)
Bit 9, Arithmetic and logic unit output equals all ones (DEQ+)
Bit 8, Arithmetic and logic unit overflow (DOVF+)
Bit 7, Shift counter output bit 4 (DSC04+)
Bit 6, Shift counter output bit 3 (DSC03+)
Bit 5, Shift counter output bit 2 (DSC02+)
Bit 4, Shift counter output bit 1 (DSC01+)
Bit 3, Shift counter output bit 0 (DSC00+)
Bit 2, Arithmetic and logic unit output zero (DCNOZ+)
Bit 1, Supervisor mode (CESK+)
Bit 0, Not used

The I switch selects the instruction (I) register for data display or entry. Pressing the I switch with the RUN indicator off or blinking (step mode or halted) displays the contents of the instruction register on the register display indicators. Changing the contents of the display register, by pressing the DISPLAY CLR switch and the register entry switches, automatically changes the contents of the instruction register. The instruction register contains the instruction to be executed next.

The P switch selects the program (P) counter for data display or entry. Pressing the P switch with the RUN indicator off or blinking (step mode or halted) displays the contents of the program counter on the register display indicators. Changing the contents of the display register, by pressing the DISPLAY CLR and register entry switches, automatically changes the contents of the program counter. The program counter contains the address of the next instruction to be fetched.

The REG switch enables one of the registers designated by the REG SELECT

switches to be selected for data display or entry.

REG SELECT Switches and Indicators

When the REG switch is pressed, any desired general-purpose register (including A, B, or X) can be selected for displaying its contents on the register display indicators or altering its contents from the register entry switches. The register selection is accomplished by entering a binary code using the four REG SELECT switches designated 8, 4, 2, 1. A one bit is produced by pressing the appropriate REG SELECT switch; a zero bit is produced by not pressing the switch. A one bit causes the corresponding indicator to light. The binary codes for specific registers are listed in table 10-1. When the binary code has been entered, the register display indicators automatically display the contents of the selected register. Changing the contents of the display register, using the DISPLAY CLR and register entry switches, automatically changes the contents of the selected register.

The binary code for a selected register can be cleared (set to zero) by pressing the CLEAR switch. Each time the INCR switch is pressed, the binary code for a selected register is incremented by one allowing the subsequent register to be selected.

INT Switch

The INT switch is used to interrupt the computer and is functional only in the run mode (RUN indicator on). Pressing the INT switch, interrupts to memory address zero.

Table 10-1. Binary Codes for Register Selection

REG SELECT Switches (8 4 2 1)	Selected
0 0 0 0	A
0 0 0 1	B
0 0 1 0	X
0 0 1 1*	
0 1 0 0*	
0 1 0 1*	
0 1 1 0*	
0 1 1 1*	
1 0 0 0*	
1 0 0 1*	
1 0 1 0*	
1 0 1 1*	
1 1 0 0*	
1 1 0 1*	
1 1 1 0*	
1 1 1 1*	

* These codes select registers that are used for WCS microprogramming. with two exceptions the contents of those registers can be displayed and altered using the control panel; however, alteration from the control panel should be done only for maintenance purposes or special applications. The register selected with the binary code equal to four always contains the contents of the instruction register. The registers selected with binary codes equal to three and five always contain all zeros and all ones, respectively; the contents of these two registers can not be altered from the control panel.

RESET Switch

Pressing the RESET switch:

a. Halts the computer

b. Stops I/O operation

c. Initializes both the computer and its peripheral devices

d. Leaves the computer in the step mode

e. Turns the RUN indicator and the STEP indicator on, if the computer was in the run mode

f. Resets the overflow indicator (bit-8 register display indicator, with STATUS switch pressed)

SENSE Switches and Indicators

The three SENSE switches permit the execution of predetermined program branching by the operator. When the program contains jump, jump-and-mark, or execution instructions that depend upon the setting of the SENSE switches, the jumps and executions occur only if the switch conditions are met.

Pressing a SENSE switch sets it and causes its associated indicator to light. Pressing the same switch again resets it, causing its indicator to go out.

EXAMPLE

A program can be written so that the operator can obtain a partial total of a column of figures being added by use of the JSS1 (jump if SENSE switch 1 is set) instruction. The program writes individual entries as long as SENSE switch 1 is not set. When the operator wants a partial total, he sets the switch. The program then jumps to an instruction sequence that prints the desired information.

Manual Operations

Using the control panel switches, data or instructions can be manually transferred to or from memory or a selected register, and stored programs can be manually executed.

Displaying Register Contents

To display the contents of the instruction register:

a. Place the computer in step mode.

b. Press I.

To display the contents of the program counter:

a. Place the computer in step mode.

b. Press P.

To display the contents of the A, B, or X registers:

a. Place the computer in step mode.

b. Press REG.

c. Using the four REG SELECT switches, enter the appropriate binary code (0000 for A register, 0001 for B register, and 0010 for X register).

Entering Data Into a Register

To enter data or instructions into a register:

a. Display the contents of the selected register as described in the preceding paragraphs.

b. Using the DISPLAY CLR and register-entry switches, enter the desired data or instruction into the selected register.

Displaying Memory Contents

To display the contents of a memory address:

a. Place the computer in step mode.

b. Press P.

c. Using the DISPL CLR and register-entry switches, enter the desired memory address into the program counter.

d. Press MEM.

e. Press DISPL. The contents of the selected memory address are now displayed on the register-display indicators. The program counter is automatically incremented.

f. Repeated actuation of the DISPL switch displays the contents of consecutive memory addresses.

Entering Data Into Memory

To enter data into memory:

a. Place the computer in the step mode.

b. Press P.

c. Using the DISPL CLR and register-entry switches, enter the desired memory address into the program counter.

d. Press MEM.

e. Using the DISPL CLR and register-entry switches, enter the desired data into the register.

f. Press ENTER to load the desired data into the previously addressed memory location. The program counter is automatically incremented.

g. Repeat steps e and f to enter data into consecutive memory addresses.

Executing a Stored Program

To execute a stored program manually:

a. Place the computer in step mode.

b. Press P.

c. Using the DISPL CLR and register-entry switches, enter the address of the first program instruction into the program counter.

d. Press I.

e. Press DISPL CLR to clear the instruction register.

f. Press START. This loads the instruction specified by the program counter into the instruction register.

g. Press START again. This executes the instruction and loads the next program instruction into the instruction register.

h. Repeat step g once for each instruction in the program.

Overflow Indication

To observe the overflow indication:

a. Place the computer in step mode.

b. Press STATUS.

c. Observe the bit-8 register display indicator. If the indicator is on, an overflow condition exists; if it is off, overflow does not exist.

Program Execution

To make a cold start (i.e., when a new system is being initialized or the contents of memory are unknown), the following operations are required:

a. Turn power on.

b. Load the bootstrap program.

c. Load the binary load/dump program.

d. Load the object program.

Descriptions of power turn on and bootstrap program loading (automatic and manual) are provided in this section. Section 16 provides descriptions for loading the binary load/dump and object programs.

Power On

Turn on computer power by placing the POWER switch to ON. When power is initially applied the following conditions will occur:

a. Step mode (STEP indicator on).

b. Sense switches not set (SENSE indicators off).

c. Register cleared (register display indicators off).

d. P switch on (P indicator on).

e. REG SELECT switches off (REG SELECT indicators off).

When power is removed and reapplied without actuation of the POWER switch (by loss and recovery of the ac line voltage), the same conditions apply, except the computer will be in the run mode (RUN indicator on) instead of the step mode.

Loading the Bootstrap Program

The bootstrap program permits the loading of the binary load/dump program into memory. Various input devices such as Teletype paper tape reader, high-speed paper tape reader, or disc memory unit can be used to load the binary load/dump program. The computer is wired at the factory to allow it to operate with a specific input device. Before the bootstrap program is loaded, the binary load/dump tape should be inserted into the Teletype paper tape reader (standard configuration) with the first binary frame at the reading station.

Addresses and instruction codes (octal) for the automatic bootstrap programs are listed in tables 10-2 and 10-3. When loading a bootstrap program manually, refer to table 10-4.

To load the automatic bootstrap program:

a. With the POWER switch in the ON position, place the computer in the run

mode by pressing the STEP/RUN switch (RUN indicator blinking)

b. Press BOOT (RUN indicator is now on). This transfers the bootstrap program from the processor's control-store to computer memory. The binary load/dump program can now be loaded into memory automatically

To load the bootstrap program manually:

a. With the POWER switch in the ON position, place the computer in step mode (STEP indicator on)

b. Press P

c. Using the DISPL CLR and register entry switches, enter the starting memory address (007756) of the bootstrap program in the program counter

d. Press MEM

e. Using the DISPL CLR and register entry switches, enter the appropriate code of the next instruction in the display register (table 10-4)

f. Press ENTER to load instruction code into the memory address specified by the program counter. The program counter is incremented automatically.

g. Repeat steps e and f for each of the remaining bootstrap instructions

Table 10-2. Automatic Bootstrap Programs for High-Speed and Teletype Readers

Address	Instruction Code	Symbolic Coding		
000200	102637* (102601)	READ	CIB	RDR
000201	004011		ASLB	NBIT
000202	004041		LRLB	1
000203	004446		LLRL	6
000204	001020		JBZ	SEL
000205	000214	(Memory address)		
000206	055000		STA	0,1
000207	001010		JAZ	LHLT + 1
000200	007000	(Memory address)		
000211	0005144		IXR	
000212	0005101	ENTR	INCR	1
000213	100537* (102601)		SEL	RDON
000214	101537* (101201)	SEL	EXC	IBFR,READ
000215	000200	(Memory address)		
000216	001000		JMP	*2
000217	000214	(Memory address)		

* When using the Teletype reader, replace this code with the one in parentheses.

Table 10-3. Automatic Bootstrap Program for Disc Memory

Address	Instruction Code
001130	100416
001131	104016
001132	100216
001133	005001
001134	103116
001135	101016
001136	001141
001137	001000
001140	001135
001141	102516
001142	151167
001143	100021
001144	001130
001145	100021
001146	100316
001147	005102
001150	103216
001151	103120
001152	006010
001153	001130
001154	103121
001155	100020
001156	100016
001157	101416
001160	001157
001161	102516
001162	151167
001163	001016
001164	001130
001165	001000
001166	000600
001167	007760

Table 10-4. Manual Bootstrap Program Instructions

Address	High-Speed Reader Code	Teletype Reader Code	Symbolic Coding
007756	102637	102601	READ CIB RDR
007757	004011	004011	ASLB NBIT -7
007760	004041	004041	LRLB 1
007761	004446	004446	LLRL 6
007762	001020	001020	JBZ SEL
007763	007772	007772	(Memory address)
007764	055000	055000	STA 0,1
007765	001010	001010	JAZ LHLT + 1
007766	007000†	007000†	(Memory address)
007767	005144	005144	IXR
007770	005101	005101	ENTR INCR 1
007771	100537	102601	SEL RDON
007772	101537	101201	SEL EXC IBFR,READ
007773	007756	007756	(Memory address)
007774	001000	001000	JMP *-2
007775	007772	007772	(Memory address)

NOTE

The bootstrap loader routine is always loaded into the highest address of the first 4K memory increment, regardless of available memory. BLD II relocation and adaptation to the specified input device are described in section 16.

† Replace this code with 007600 if the test executive of MAINTAIN III (refer to document number 98 A 9952 07x) is to be loaded and executed.

SECTION 11 - MAINTENANCE

Integrated-circuit (IC) design reduces the occurrence of malfunctions in Varian 73 systems. Convenient packaging methods make all system elements accessible for troubleshooting and maintenance in the rare case when the system malfunctions.

Complete troubleshooting and maintenance information for the Varian 73 systems is presented in the Varian 70 series maintenance manuals, in the MAINTAIN III Manual (document number 98 A 9952 07x), in applicable computer and I/O option and peripheral controller manuals, and in manufacturer's instruction manuals with peripheral devices.

This section of the Handbook outlines, in general terms, routine maintenance and troubleshooting concepts.

Routine Maintenance

The PF/R (section 2) is an attractive addition to the Varian 73 system. It provides an orderly shutdown in case of power failure or turn-off, and automatically restarts the interrupted program when power is restored.

To prevent accidental setting of control panel switches during computer operation, turn the POWER switch to the console DISABLE position.

Cooling fans for the memory and power supplies, are permanently lubricated and require no routine attention.

To ensure effective maintenance of the Varian 73 system:

a. Study the documentation furnished with the equipment.

b. Assess system performance with adequate test equipment.

c. Use the available maintenance aids and test programs.

d. Apply orderly and logical troubleshooting techniques.

Test Equipment

Table 11-1 lists the test equipment recommended for computer maintenance. These items also satisfy maintenance requirements of the peripheral controllers.

Test Programs

The MAINTAIN III test program system briefly outlined in section 27, and described in detail in the MAINTAIN III Manual (document number 98 A 9952 07x), verifies correct system operation. This system tests all phases of system operation, including memory, machine instructions, computer and I/O options, and peripherals and their controllers.

MAINTAIN III aids in preventive maintenance by determining whether the system is actually malfunctioning, and, in most cases, helps to isolate the error if one exists. When the system is definitely malfunctioning and the exact nature of the trouble is not known, test programs that exercise the suspected area of the fault can be loaded and executed independent of other MAINTAIN III programs.

Table 11-1. Recommended Test Equipment

Item	Description
Oscilloscope	Tektronix, type 547 (or equivalent) with dual-trace plug-in unit
Multimeter	Simpson 260 (or equivalent)
Extender Board	Varian extender board type DM312 (part number 44P0540)
Board Puller	Titchener 1731 (or equivalent)
Wire-wrap Gun	Gardener-Denver 14R2 (or equivalent)
Soldering Iron	15-watt pencil type
Wire Stripper	Thermo-strip type
Assorted Hand Tools	Screwdrivers, spin-tight wrenches, long-nosed pliers, etc.

Circuit Board Accessibility

A circuit board in the mainframe can be made accessible for troubleshooting as follows:

a. If rack mounted, extend the mainframe forward on its slides.

b. Remove the mainframe top cover.

c. Insert the board suspected of the malfunction into the top mainframe card slot (the processor and option boards should normally remain in adjacent card slots).

The suspected circuit board is now accessible for troubleshooting through the top of the mainframe.

A circuit board in the I/O expansion chassis can be installed on a DM312 extender board (table 11-1). This configuration extends the circuit board outside the rear of the I/O expansion chassis providing access for troubleshooting.

Troubleshooting

Although many troubleshooting techniques can be applied, there is no substitute for an ordered, logical analysis of the problem and isolation of the cause of a failure to the level of the faulty component. Such an analysis can be based only on thorough knowledge of the equipment design.

One of the most valuable troubleshooting aids available to the technician is the control panel. Using the switches and

indicators, the operator can execute simple procedures manually to check out computer operation.

In general, the characteristics of a failure can indicate the faulty section of the system; for example:

a. A CPU failure usually produces errors in a large percentage of the MAINTAIN III test programs. If the failure is catastrophic and all instructions fail, the cause is usually in the timing, decoding, and control sections. Incorrect arithmetic operations or incorrect incrementation of the P register indicates that the arithmetic/logic and register sections are at fault.

b. Memory failures are indicated by repeated, unprogrammed halts in the execution of a program, or by completely random instruction sequencing. Malfunctions of a single memory bit or word are rare and usually require special test procedures (refer to the appropriate Varian maintenance manual).

c. Failures in I/O operations are associated only with the logic of the failing device. Such failures can be easily diagnosed if malfunctions occur only during the execution of I/O routines.

General troubleshooting steps are summarized below.

Define the problem thoroughly. For example, if the ADD instruction (section 14) does not produce correct results, check that the fault is a function of the sign (plus or minus), of the carries, or of some other element. Verify that the operation registers are being loaded properly. Use the displays, connector pins, and IC terminals for gathering the necessary data.

Look for obvious solutions. Make sure that a malfunction has actually occurred. Relate problems to recent events, such as cleaning and servicing. Look for improperly set controls or test equipment and accidental disconnections of plugs, etc. Consider miscellaneous temporary failure, such as mechanical jamming of peripheral equipment.

Isolate the fault to a functional area, such as memory control, arithmetic/logic, operation register, I/O, peripheral controller, or peripheral device. This is generally a straightforward process of eliminating areas that are operating properly.

Analyze the faulty area. Use the logic and timing diagrams, and observe waveforms to isolate the problem to an individual replaceable element. Make sure that the computer is in step mode (STEP indicator on) before removing input power.

Correct the fault by replacing the faulty circuit card or component. Before restoring power, take any necessary measures to prevent recurrence of the failure.

Restore the system to normal operation. Verify proper operation by running the test programs.

SECTION 12 - DATA AND INSTRUCTION FORMATS

There are two basic word formats used in the Varian 73: data and instruction.

The instruction word format is further divided into four types: single-word addressing, single-word non-addressing, double-word addressing and double-word non-addressing.

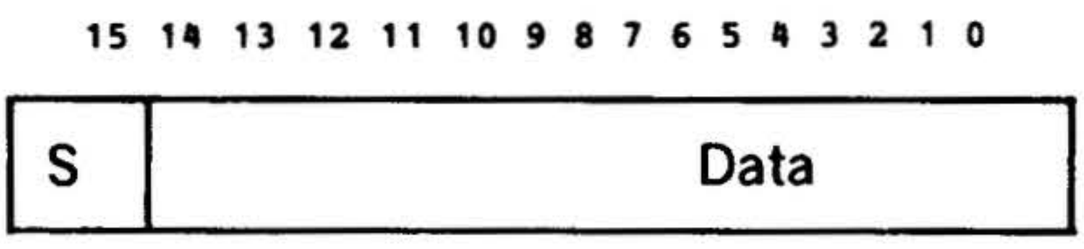
The floating point option data and instruction formats, are described at the end of this section.

Data Word Formats

Data words may contain operands, operand addresses or indirect addresses, depending upon the instruction or addressing mode in process.

Data Words

The data word format is:

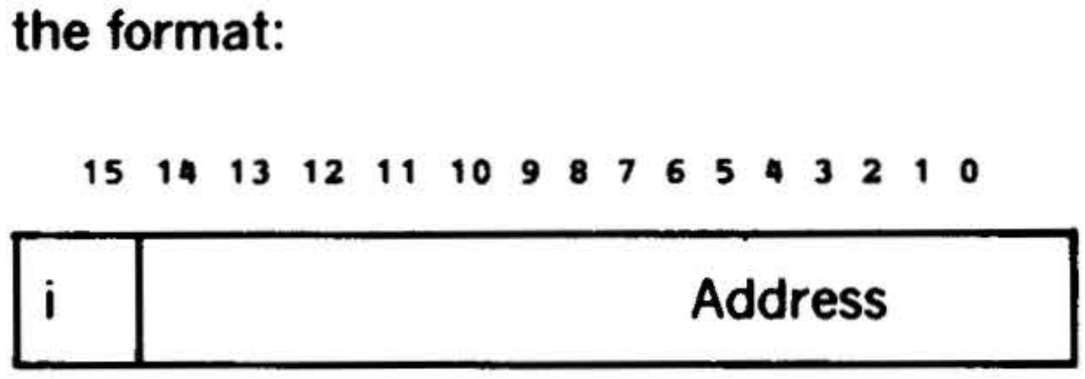


The most significant bit (bit 15) is the sign bit. It is one for negative numbers and zero for positive numbers. The other 15 bits (0-14) contain the data itself.

Negative numbers are represented in two's complement form. Zero is considered positive.

Direct/Indirect Addresses

When the data word is direct/indirect address, rather than an operand, it has the format:



i = 0, word contains operand location, i = 1, word contains indirect address word location

A direct/indirect-address word is accessed by an instruction that is in the direct/indirect address modes (see section 13).

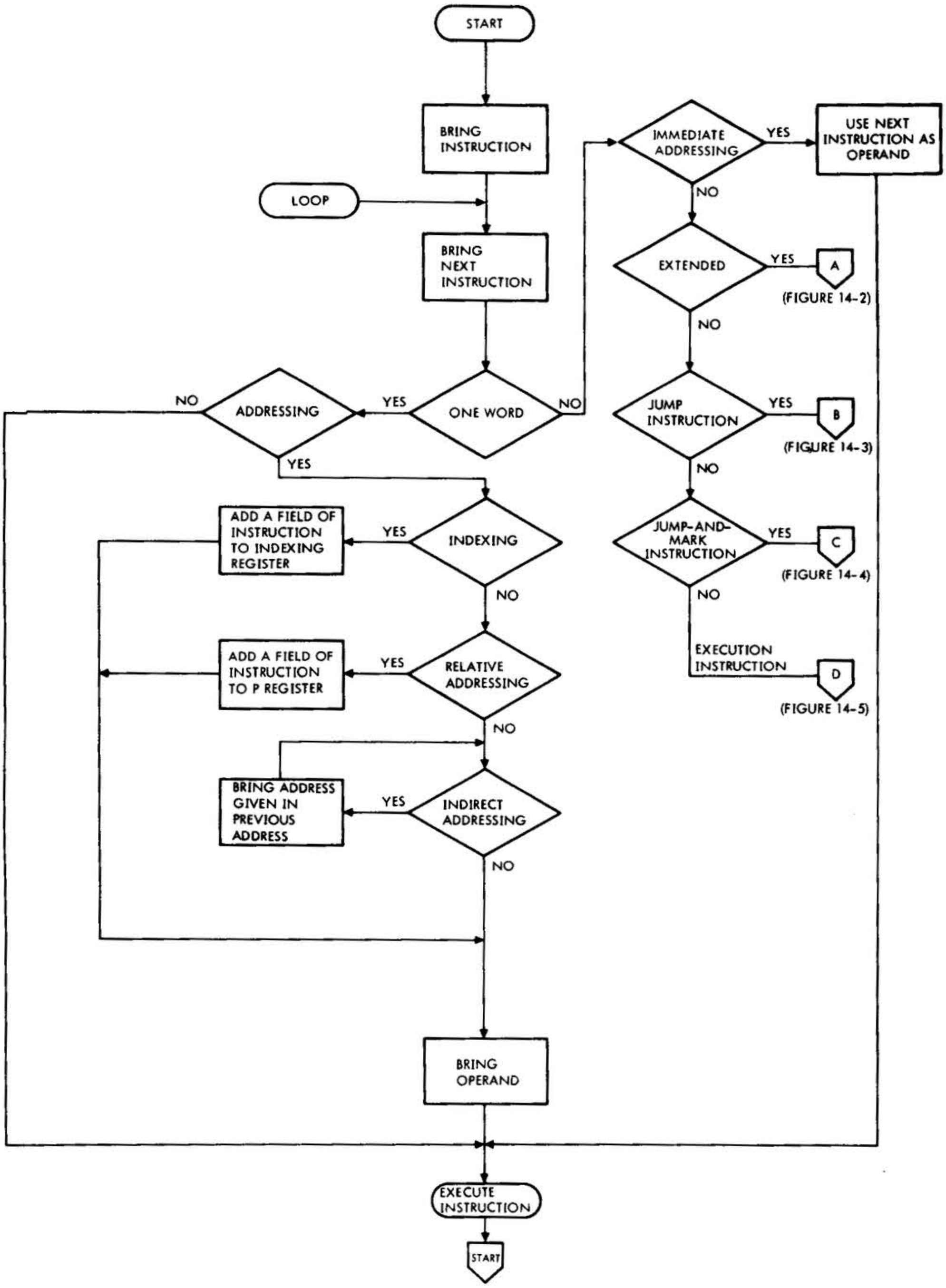
Bit 15 contains the i bit, which designates whether the memory location that is addressed by the direct/indirect-address word contains the operand (i = 0), or contains the location of yet another direct/indirect address word (i = 1).

Direct/indirect addressing may be extended to many levels. Each level of direct/indirect addressing adds approximately one cycle to the basic execution time of an instruction.

Instruction Word Formats

Instruction words may be either addressing or non-addressing, single-word or double-word.

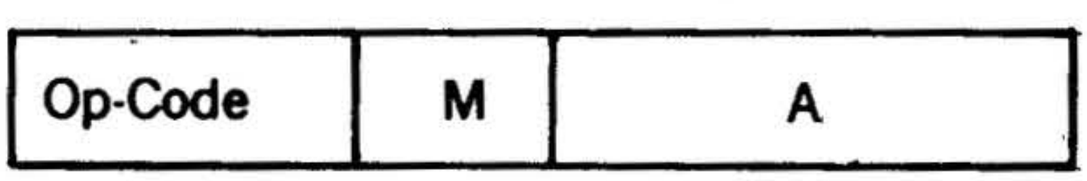
Figure 12-1 is a simplified flowchart of instruction processing operations. Addressing mode information is given in section 13.



VTI3-02/78

Figure 12-1. Instruction Processing, Simplified Flow

The basic instruction word format:



The format shown is applicable to all instruction words. For double-word instructions, the format shown applies to the first instruction word.

The instruction word is divided into three fields; op-code field, M field and A field. The function of the three fields vary according to the type of instruction, but may generally be defined as follows:

Op-code field	bits 12-15	Designates type of instruction (e.g., single-word addressing, I/O instructions or other).
M field	bits 9-11	Designates address mode or mode of operation
A field	bits 0-8	Contains a variety of information depending upon type of instruction

Single-Word Addressing Instructions

Instruction groups applicable to this type of instruction are the direct version of:

Load/Store
Arithmetic
Logical

These instruction groups are designated by octal number 01 through 07, and 11 through 17 in the op-code field. The M field contains one of the following addressing modes:

Direct	binary 0 X X
Relative mode	binary 1 0 0
Index (X)	binary 1 0 1
Index (B)	binary 1 1 0
Indirect	binary 1 1 1

For direct addressing, bits 9 and 10 of the M field are combined with the A field to form a direct address to any of the first 2,048 locations.

Single-Word Non-Addressing Instructions

Instruction groups applicable to this type of instruction are:

Shift
Control
Register Change
Input/Output

The op-code field contains octal 00 except for the last type, Input/output, which is designated by octal 10. The M field designates the mode of operation, and the A field specifies the action to be performed by the computer such as:

a. Number of shifts

b. Kind of register change as well as source and destination registers

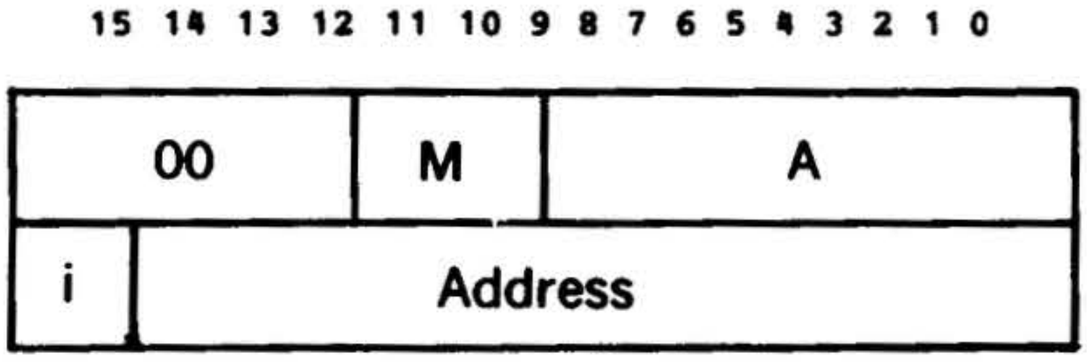
c. Input/output

Double-Word Addressing Instructions

Instruction groups applicable to this type of instruction are:

Jump
Jump and Mark
Execute
Extended Address Version of:
Load/Store
Arithmetic
Logic

The format for both the first and second words of two-word addressing instructions other than the optional extended-addressing instructions is:

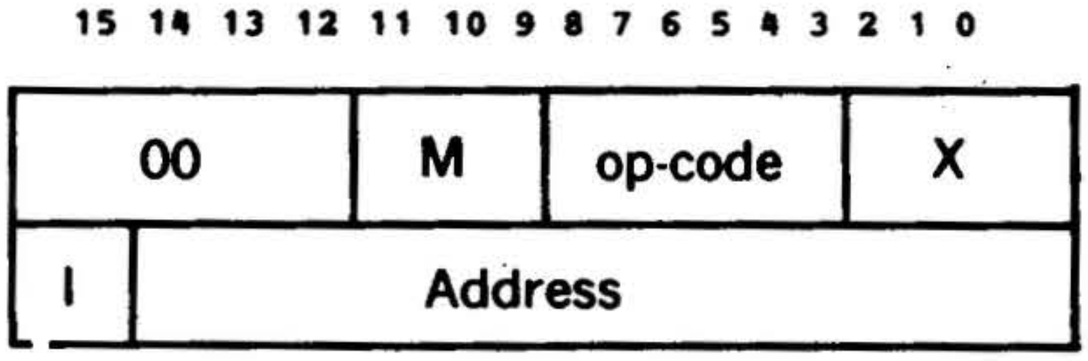


i = 0, word contains an address, i = 1, word contains an indirect address

The op-code field contains octal 00; the M field an octal 1, 2, 3, or 6, designating the mode of instruction to be performed; and the A field defines the logical states which condition the execution of the instruction.

The second word contains address of either an instruction or operand, or the location of the instruction to be executed if the condition is met. Indirect addressing is permitted.

For the extended address instructions (see Section 15), the A field is further divided into two sub-fields. Bits 0-2 form the X field, and are coded to indicate the address mode. Bits 3-8 contain any single-word operation instruction which, in a single word instruction, ordinarily appear in the op-code field.



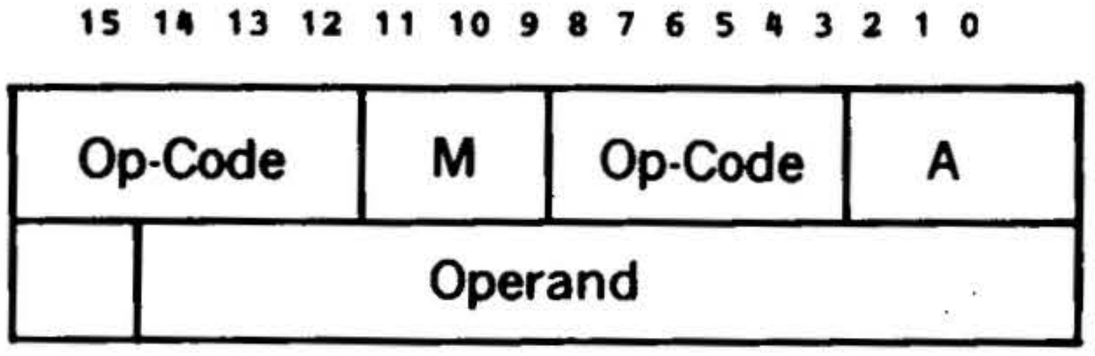
1 = 0, word contains an address, 1 = 1, word contains an indirect address

Double-Word Non-Address Instructions

Instruction groups applicable to this type of instruction are the immediate-addressing version of the following:

Load/Store
Arithmetic
Logical

The format for these instructions is:



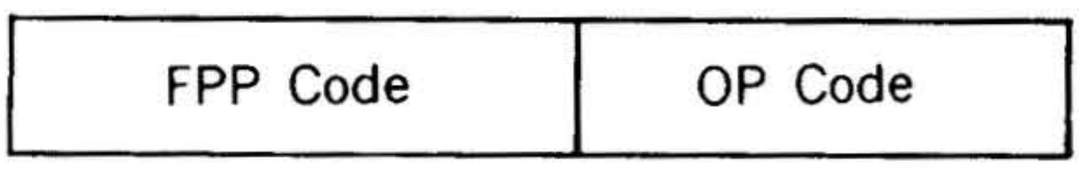
The op-code field contains octal 00 and the M field contains an octal 6. The A field contains the operation mode (octal 0) in bits 0-2, and bits 3-8 contain the equivalent of a single word operation instruction.

Since addressing is not permitted, the second word always contains an operand.

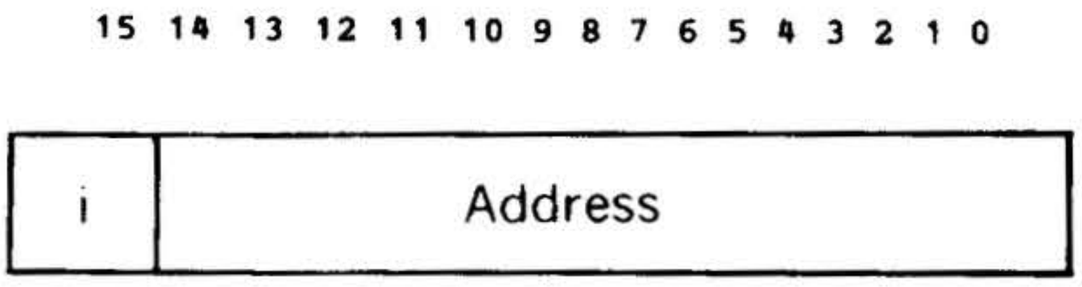
Figure 12-2 is a flowchart for extended addressing. Figures 12-3 and 12-4 are flowcharts for Jump and Jump-and-mark instructions, respectively. Figure 12-5 is a flowchart for execute instructions.

Floating Point Instruction Format

All instructions recognized by the floating point processor option have the following format:



The memory word fetched after the instruction is always an operand address with the following format:



i = 0, word contains address of the operand.

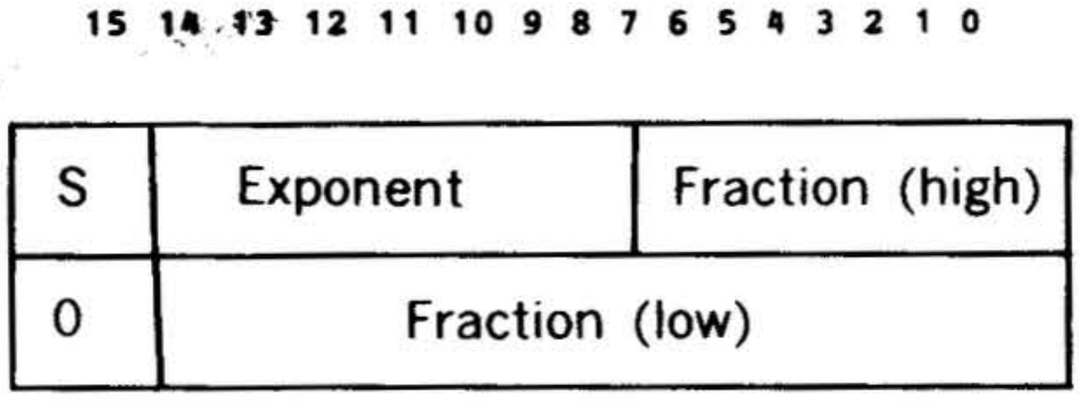
i = 1, word contains indirect address word location.

The address points to the first data word in a sequence of 2, if the instruction is a single precision floating point instruction. If the instruction is double precision, the address points to a sequence of 4 words.

Floating Point Data Word Formats

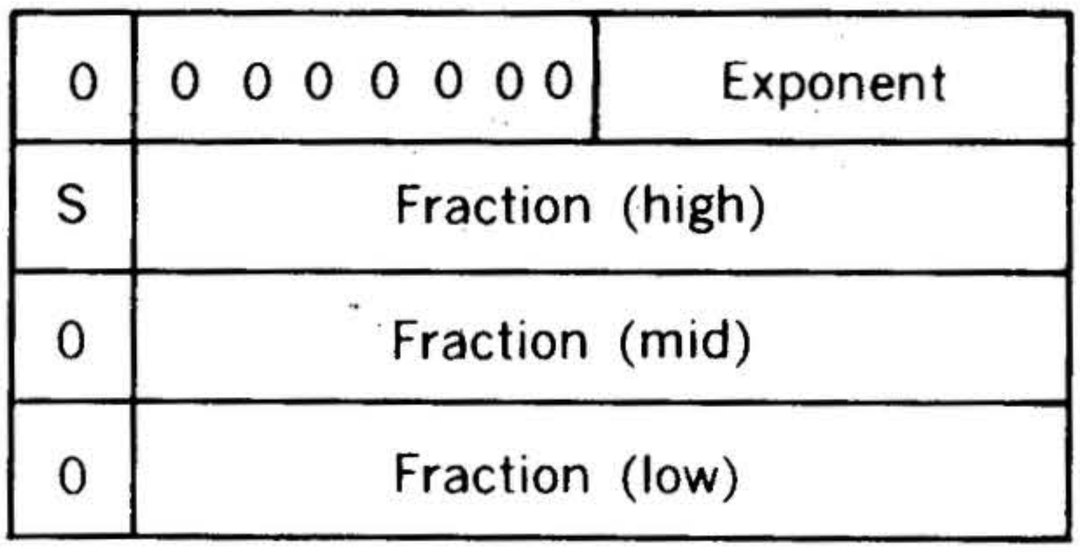
Floating point data is stored in memory in single and double precision word formats.

Single Precision



Double Precision

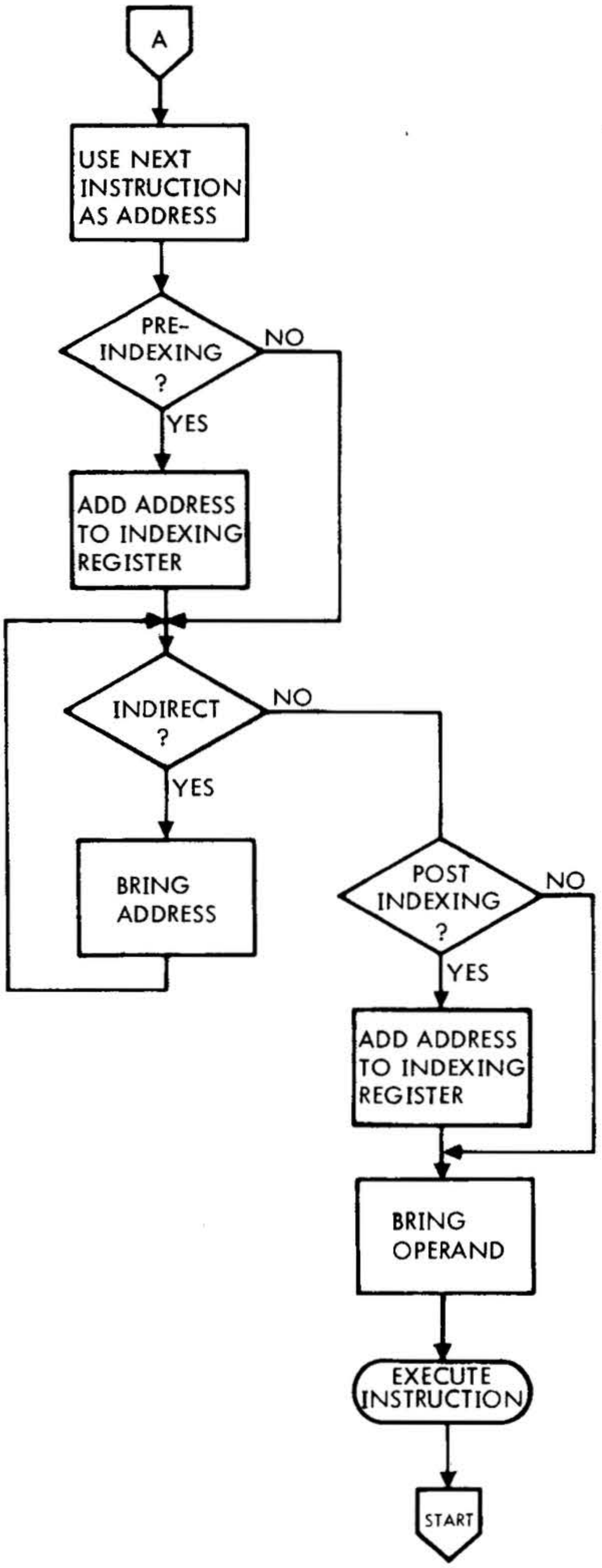
The double precision data word format is:



The exponent is represented in an excess 128 format so that the smallest exponent representable contains all 0's. An exponent field containing 128 $(200)_8$, corresponds to an exponent value of 0. The largest exponent representable contains all 1's.

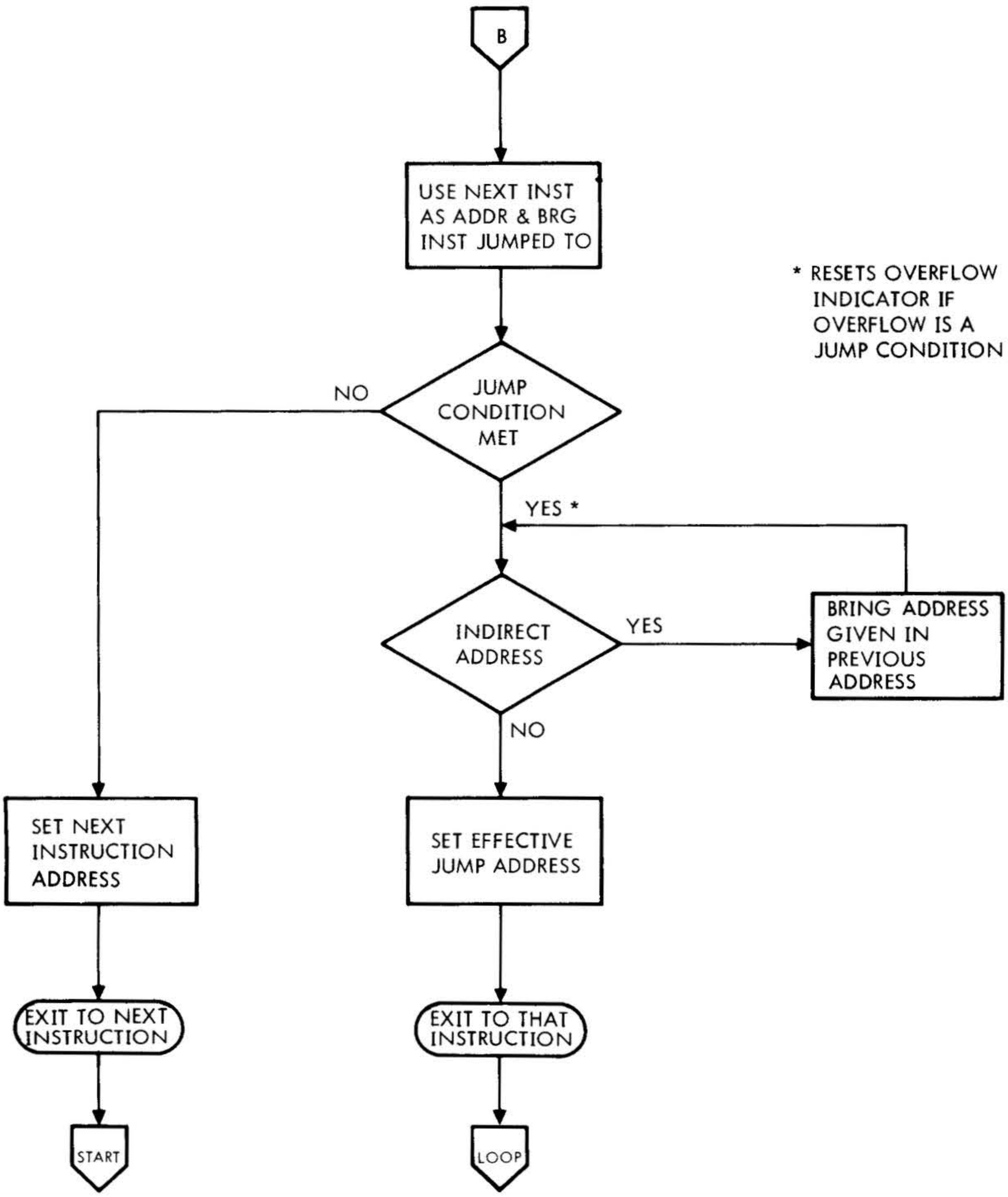
The fraction is expressed in a modified sign-magnitude format. Rather than inverting the sign bit for negative numbers, the complete word in which the sign appears is inverted. In single precision, this inverts the exponent, the sign, and the high 7 bits of the fraction. In double precision, the sign and the high 15 bits of the fraction are inverted.

The number 0 is represented by all 0's. All other numbers are normalized.



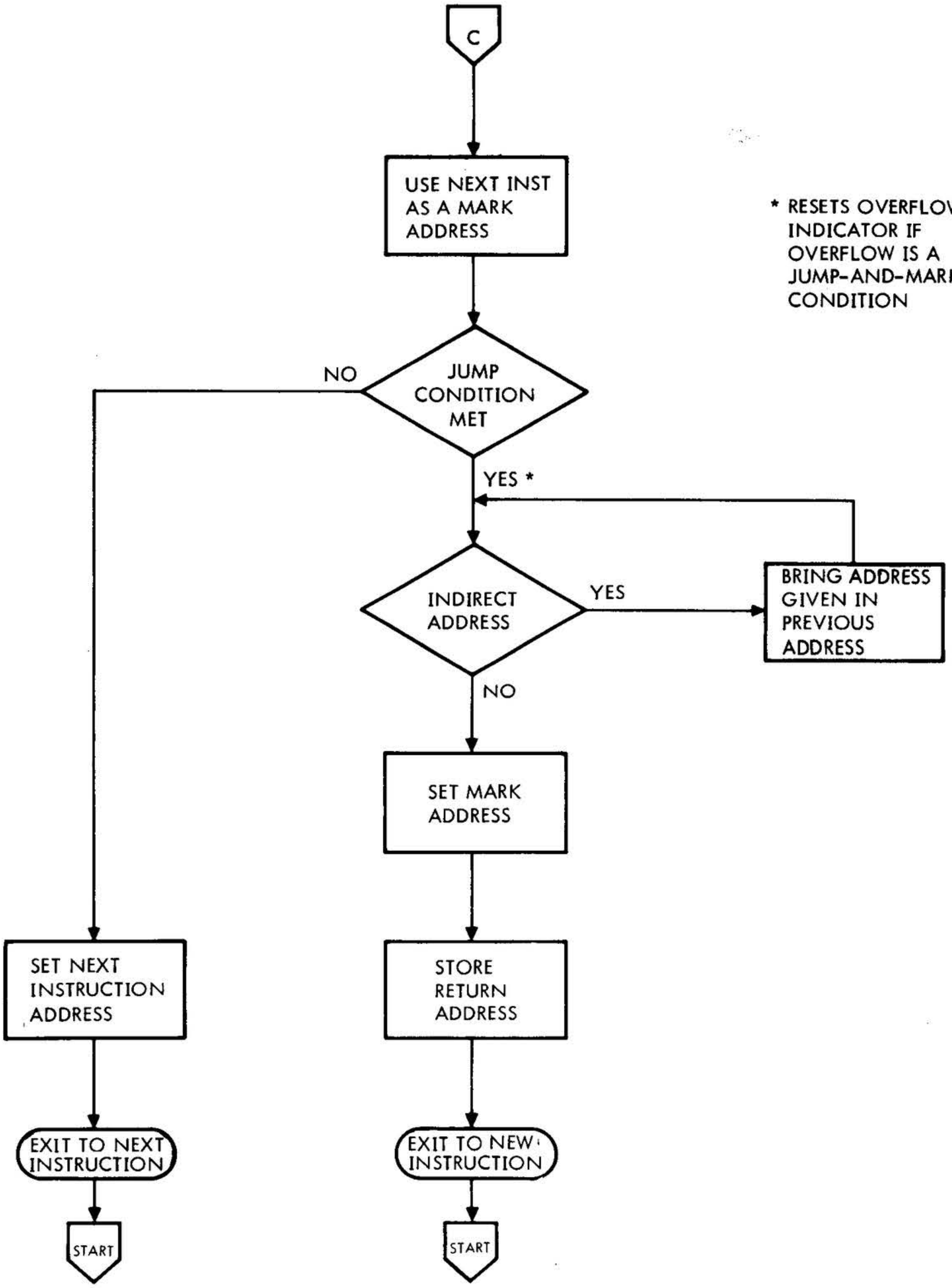
VTI1-1503

Figure 12-2. Extended Instruction, General Flow



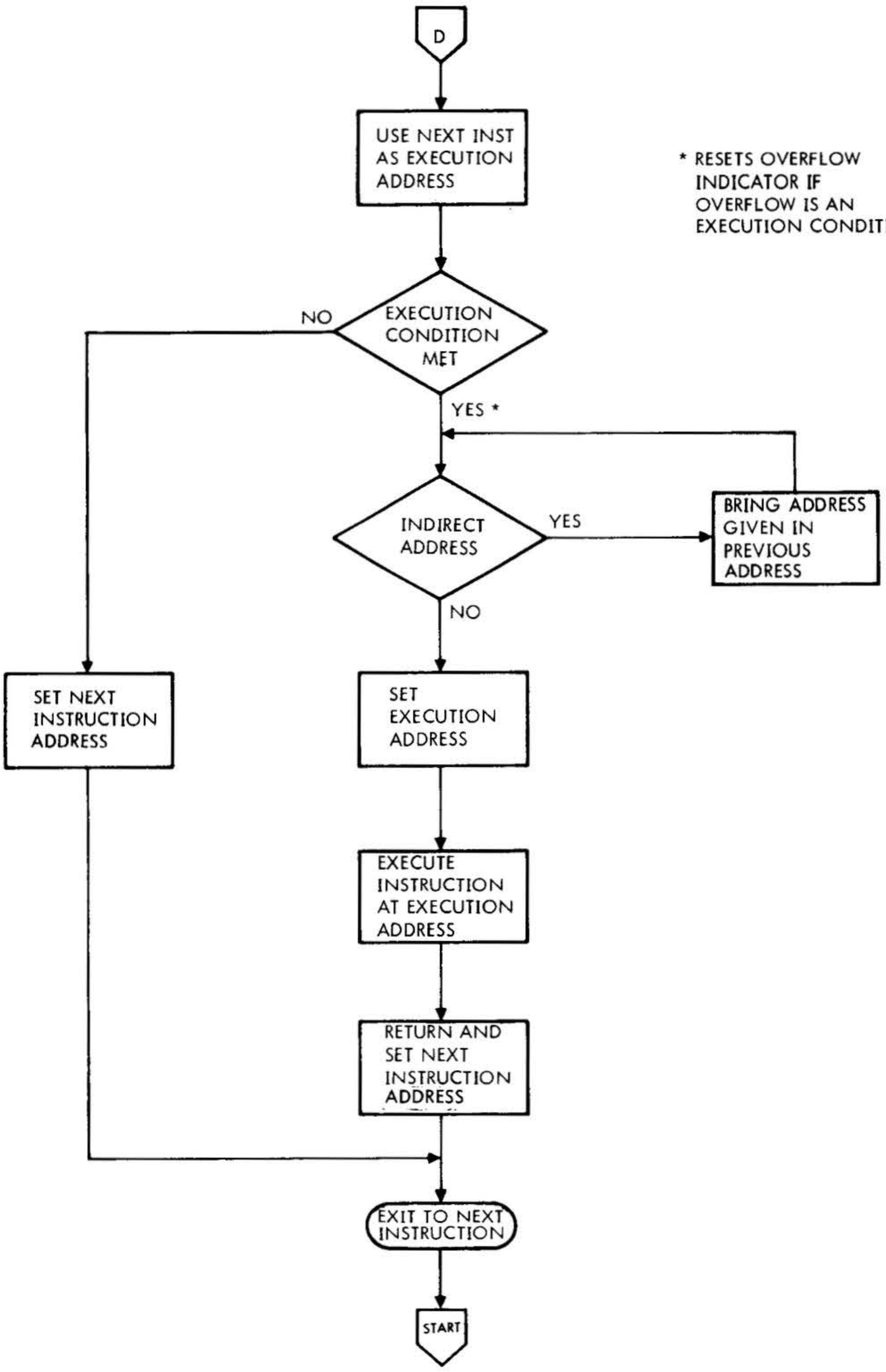
VTI1-1497

Figure 12-3. Jump Instruction, General Flow



VTII-1498

Figure 12-4. Jump and Mark Instruction, General Flow



VTI1-1499

Figure 12-5. Execute Instruction, General Flow

SECTION 13 - ADDRESSING MODES

The Varian 73 features a number of addressing modes that can be used by the programmer to increase the efficiency of his program.

The addressing mode is a function of both the type of instruction and the coding within the instruction.

Three types of instructions are involved: single-word addressing, extended, and immediate.

Table 13-1 summarizes the relationship between the three instruction types and the eight addressing modes.

Immediate instructions are limited to the immediate-addressing mode. Single-word addressing instructions may be used in all of the addressing modes except immediate. Extended instructions can take advantage of all eight addressing modes.

Address-mode codes for the single-word instruction and extended instructions are given in tables 13-2 and 13-3. For additional details on the format of these instructions, see section 14.

Immediate Addressing

Immediate Addressing with Immediate Instructions

Immediate instructions are nominally classified (Section 14) as double-word nonaddressing instructions since the second word of the instruction is the operand itself. No further addressing of memory is required.

Table 13-1. Relationship Between Instruction Type and Addressing Mode

Addressing Mode	Instruction Type		
	Single-Word	Extended	Immediate
Immediate		X	X
Direct	X	X	
Indirect	X	X	
Preindexed with X Register	X	X	
Postindexed with X Register		X	
Preindexed with B Register	X	X	
Postindexed with B Register		X	
Relative to P Register (PRE)	X	X	
Relative to P Register (POST)		X	

"Immediate addressing" is included in this discussion, however, because it is one of the options available to the programmer for accessing operands stored in memory.

The address of the operand is, in this case, the memory location containing the second word of the instruction. The processor addresses this location when it fetches the immediate instruction for execution.

Since there is no separate addressing phase, no modification of the address is possible. Indexed, relative, and indirect addressing do not apply to immediate instructions.

Immediate Addressing with Extended Instructions

Extended-address instructions may be used in the immediate mode by inserting an octal 0, 1, 2, or 3 in the X field of the first word of the instruction.

With this code in effect, the CPU processes the second word of the instruction as an operand rather than as an address.

Direct Addressing

In direct addressing, the address of the operand is contained within the instruction itself.

Direct Addressing with Single-Word Instructions

Single-word addressing instructions operate in the direct mode whenever the most significant bit of the M field (table 13-2) is a 0.

The remaining two bits of the M field are combined with the nine bits in the A field to form an 11-bit effective address. The address directs the processor to an operand stored in the first 2,048 words (0000 thru 2,047) of memory.

Direct Addressing with Extended Instructions

Extended-addressing instructions operate in the direct mode whenever an octal 7 is inserted in the X field and the most significant bit of the second word is 0.

The remaining 15 bits of the second word form the effective address of the operand. Any location in a full 32K of memory can be directly addressed.

Indirect Addressing

In indirect addressing, the address of the operand is stored in memory at a location specified by the instruction.

Indirect Addressing with Single-Word Instructions

Single-word addressing instructions operate in the indirect mode whenever an octal 7 is inserted in the M field.

The 9 bits of the A field direct the processor to an address location in the first 512 words of memory. The word stored in that location is the address of the operand.

Indirect Addressing with Extended Instructions

Extended-addressing instructions operate in the indirect mode whenever the most significant bit of the second word is 1 and the X field is 7.

The remaining 15 bits of the second word direct the processor to any address location in a full 32K of memory. The word stored in that location is the address of the operand or another address.

Multi-Level Indirect Addressing

The word stored in the memory location specified by any indirect-addressing instruction may itself be an indirect address. The CPU is directed to this second memory location to determine the address of the operand.

Indirect addressing is limited to five levels for one-word instructions, and to four levels with two-word instructions.

Indexed with X Register

The effective address may be modified by adding it to the contents of the X register.

X-Register Indexing with Single-Word Instructions

Single-word addressing instructions may be indexed with the X register by inserting an octal 5 in the M field.

The contents of the A field are added to the contents of the X register to form the effective address of the operand.

X-Register Preindexing and Postindexing with Extended Instructions

Extended-addressing instructions may be indexed with the X register by inserting an octal 5 in the X field of the first word of the instruction. (See figure 13-1 for effect of pre- and postindexing.)

If bit 7 of the first word is zero, preindexing is specified. The contents of the second word of the instruction are added to the contents of the X register, and the result is used as intermediate address for subsequent indirect-addressing steps.

If bit 7 of the first word is one, postindexing is specified. All indirect-addressing steps are performed first, and the final address is then added to the contents of the X register to form the effective address.

Indexed with B Register

The effective address may be modified by adding it to the contents of the B register.

B-Register Indexing with Single-Word Instructions

Single-word addressing instructions may be indexed with the B register by inserting an octal 6 in the M field.

The contents of the A field are added to the contents of the B register to form the effective address of the operand.

B-Register Preindexing and Postindexing with Extended Instructions

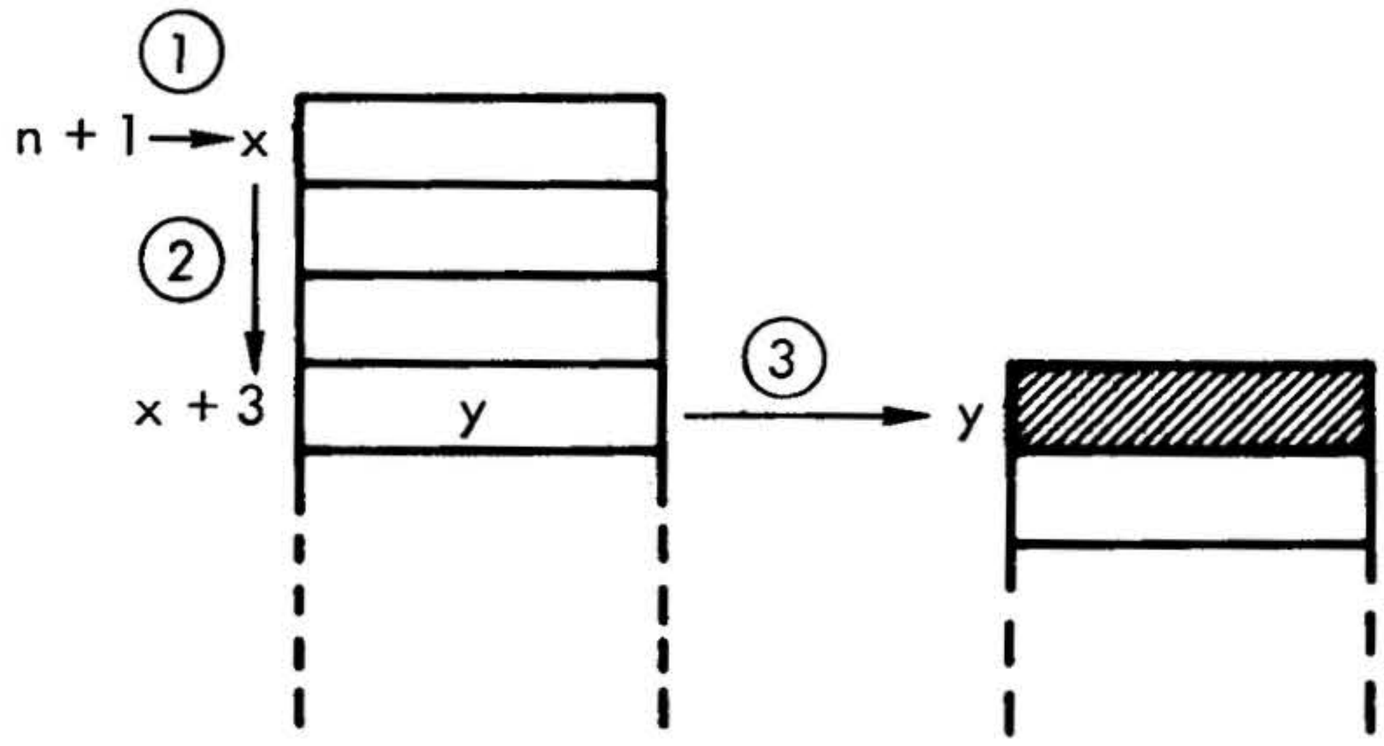
Extended-addressing instructions may be indexed with the B register by inserting an octal 6 in the X field of the first word of the instruction.

If bit 7 of the first word is zero, preindexing is specified. The contents of the second word of the instruction are added to the contents of the B register, and the result is used as intermediate address for subsequent indirect-addressing steps.

PREINDEXING

Let (n + 1) = x
(B) = 3

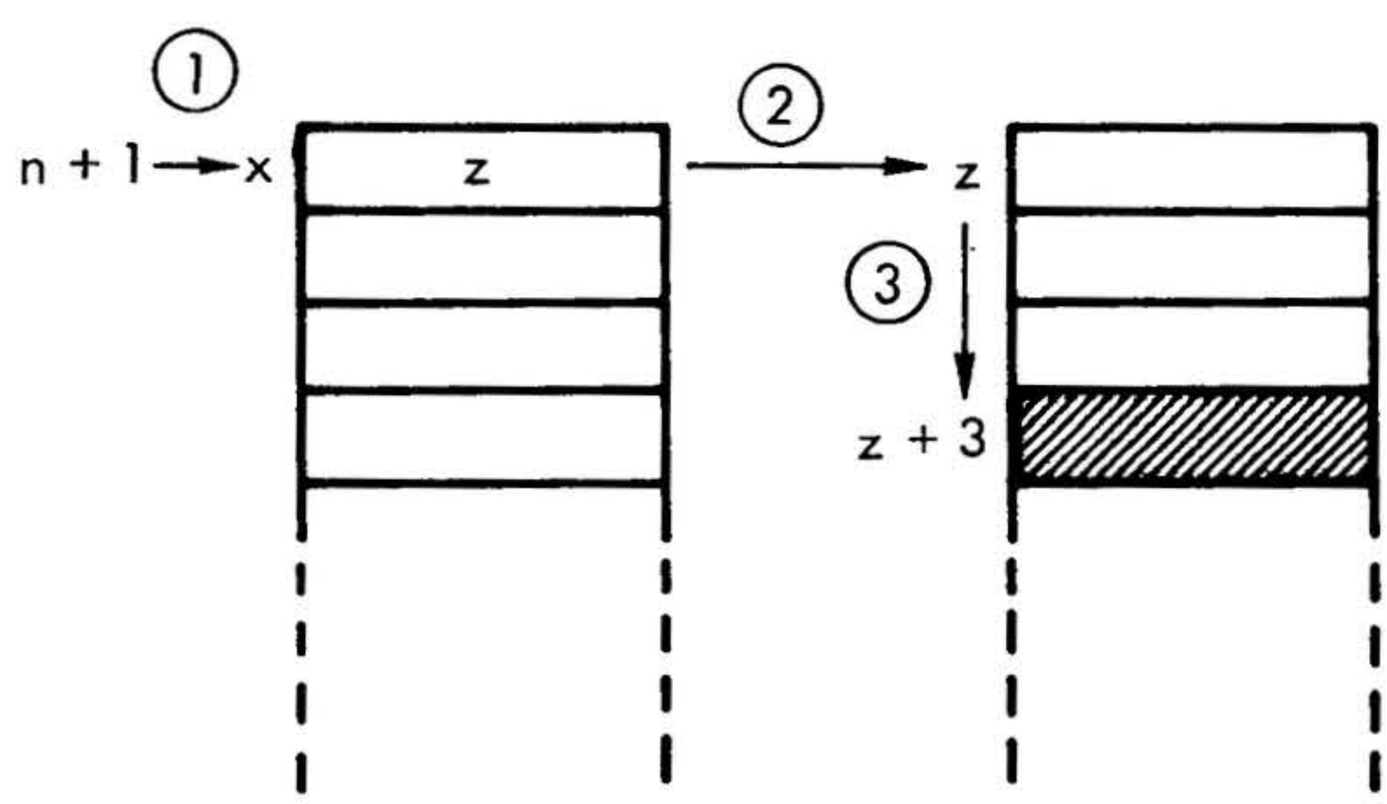
(n + 1) + (B) = x + 3
(x + 3) = y



POSTINDEXING

Let (n + 1) = x
(B) = 3

(x) = z
z + (B) = z + 3



Circled numbers indicate the sequence of events.
Shaded area is the effective memory address.

VTI1-0682

Figure 13-1. Preindexing and Postindexing

If bit 7 of the first word is one, postindexing is specified. All indirect-addressing steps are performed first, and the final address is then added to the contents of the B register to form the effective address.

Relative Addressing

In relative addressing, the address contained within an instruction is modified by adding it to the contents of the P register.

Relative Addressing with Single-Word Instructions

Single-word addressing instructions operate in the relative mode whenever an octal 4 is inserted in the M field.

The contents of the A field are added to the contents of the P register to form the effective address of the operand. This permits addressing locations up to 512 words in advance of the current program locations.

NOTE: P register is +1 from the instruction.

Relative Addressing with Extended Instructions

Extended-addressing instructions operate in the relative mode whenever an octal 4 is inserted in the X field of the first word of the instruction.

The contents of the second word of the instruction are added to the contents of the P register to form the effective address of the operand.

Table 13-2. Address Coding for Single-Word Instructions

M Field (octal)	Addressing Mode	Operation
0-3	Direct	Two least significant bits of M field are combined with A field to form effective address.
4	Relative	A field is added to contents of P register plus one to form effective address.
5	Indexed with X register	A field is added to contents of X register to form effective address.
6	Indexed with B register	A field is added to contents of B register to form effective address.
7	Indirect	A field specifies location in which address of operand is stored.

Table 13-3. Address Coding for Extended-Addressing Instructions

X Field (octal)	Addressing Mode	Operation
0-3	Immediate	Second word of instruction contains the operand.
4	Relative	Second word of instruction is added to contents of P register to form effective address.
5	Indexed with X register	Second word of instruction is added to contents of X register to form effective address.
6	Indexed with B register	Second word of instruction is added to contents of B register to form effective address.
7	Direct	If the most significant bit of the second word is 0, the remaining bits are the operand address.
7	Indirect	If the most significant bit of the second word is 1, the remaining bits identify the location of the operand address.

THIS

PAGE

WAS

LEFT

BLANK

INTENTIONALLY

THIS

PAGE

WAS

LEFT

BLANK

INTENTIONALLY

SECTION 14 - INSTRUCTION SET

This dictionary of Varian 73 computer instructions can be divided into the following functional groups:

- Load/store instructions
- Arithmetic instructions
- Logic instructions
- Shift/rotation instructions
- Register transfer/modification
- Jump instructions
- Jump-and-mark instructions
- Execution instructions
- Control instructions
- I/O instructions
- Floating point processor instructions

Table 14-1 summarizes the length and addressing mode information applicable to these groups of instructions.

Table 14-1. Instruction Groups

Group	Length	Addressing
Load/store	One word	Normal
	Two words	Extended
	Two words	Immediate
Arithmetic	One word	Normal
	Two words	Extended
	Two words	Immediate
Logic	One word	Normal
	Two words	Extended
	Two words	Immediate
Shift/rotation	One word	None
Register transfer/ modification	One word	None
Jump	Two words	Extended
Jump and mark	Two words	Extended
Execution	Two words	Extended
Control	One word	None
Input/output	One word	None
	Two words	Extended (Direct)
Floating point processor	Two words	Normal

Appendix A is a list of the Varian 73 system instructions, arranged alphabetically by mnemonic and indexed to the page of this handbook where the instruction is explained.

The physical implementation of the Varian 73 instruction set has resulted in the most efficient use of machine execution time. During the execution of one instruction the next instruction is being fetched. This techneque is commonly referred to as "pipelining". Instruction execution time becomes a function of the instruction sequence and memory cycle time. This relationship is described for each instruction type in Appendix H.

Load/Store Instructions

This group comprises the instructions for loading registers from memory or for storing the contents of registers in memory. Subgroups of these instructions permit such loading or storing in normal, extended, or immediate addressing modes.

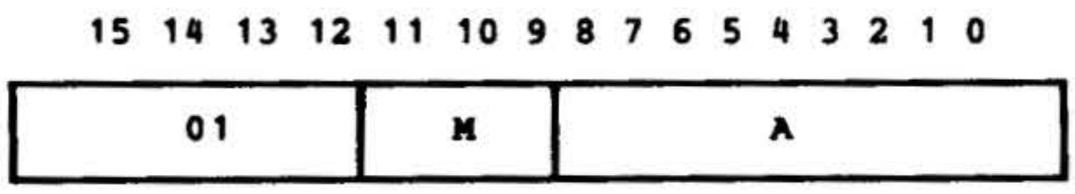
Normal Load/Store Instructions

Mnemonic	Instruction
LDA	Load A register
LDB	Load B register
LDX	Load X register
STA	Store A register
STB	Store B register
STX	Store X register

These instructions have the following one-word addressing format:

The operation code varies for each instruction. The A field contains an address in the addressing mode specified by the M field.

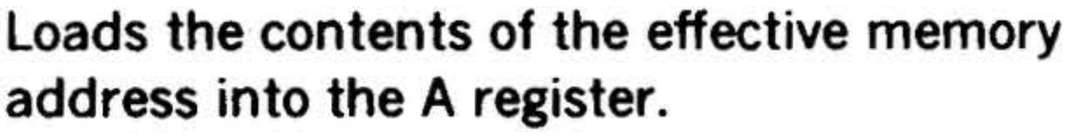
LDA **Load A Register**



Loads the contents of the effective memory address into the A register.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

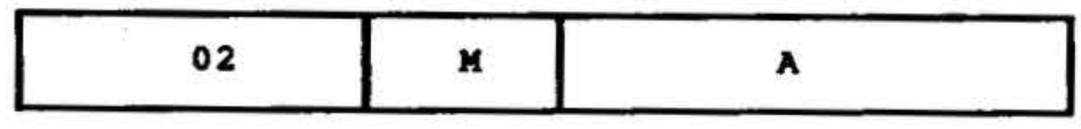
LDB **Load B Register**



Loads the contents of the effective memory address into the B register.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	B

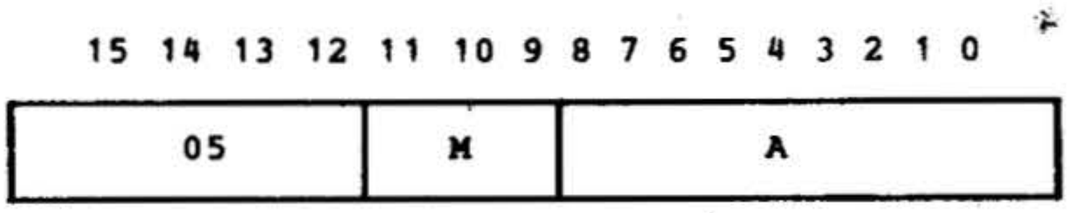
LDX **Load X Register**



Loads the contents of the effective memory address into the X register.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	X

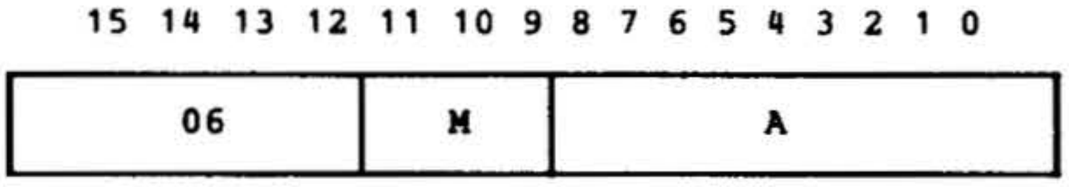
STA Store A Register



Stores the contents of the A register in the effective memory address.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

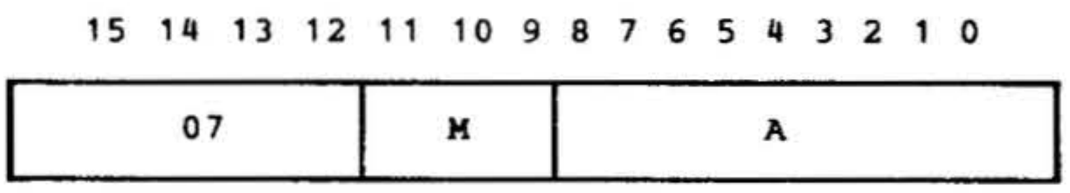
STB Store B Register



Stores the contents of the B register in the effective memory address.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

STX Store X Register



Stores the contents of the X register in the effective memory address.

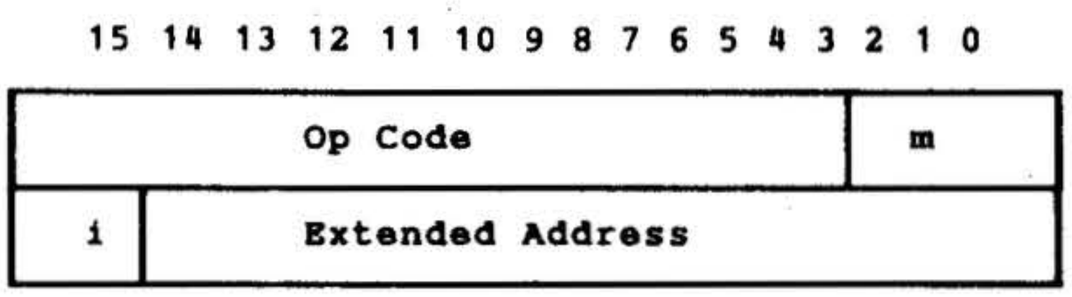
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Memory

Extended Load/Store Instructions

This group includes:

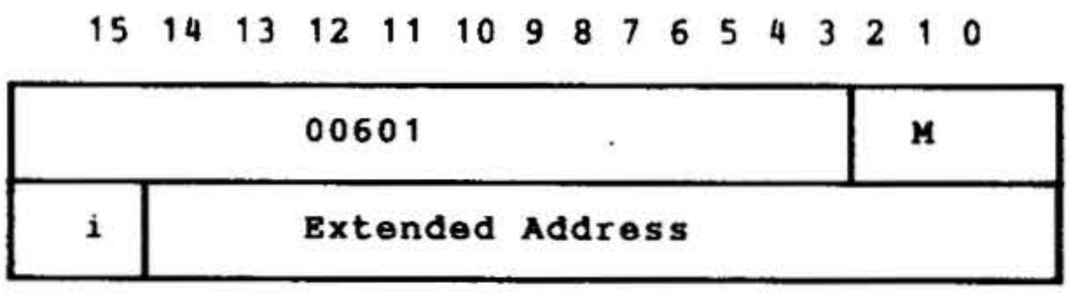
Mnemonic	Instruction
LDAE	Load A register extended
LDBE	Load B register extended
LDXE	Load X register extended
STAE	Store A register extended
STBE	Store B register extended
STXE	Store X register extended

These instructions have the following two-word addressing format:



These instructions have configuration 0 000 110 000 in bits 6-15. Bits 3-5 specify the manipulation. Note that the configuration of these bits is the same as the operation code for the corresponding one-word load/store instruction.

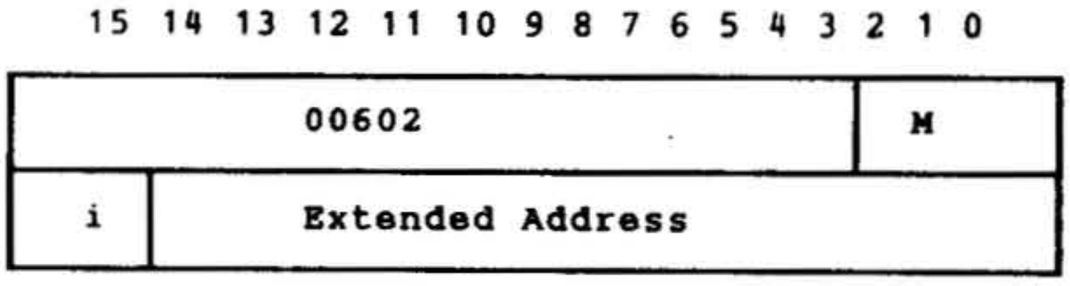
LDAE Load A Register Extended



Loads into the A register the contents of the effective memory address given by the operand address.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

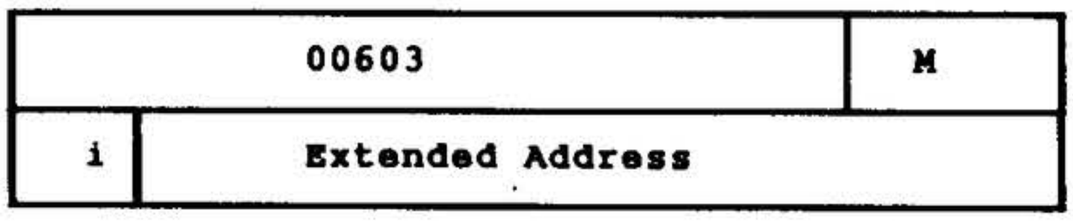
LDBE Load B Register Extended



Loads into the B register the contents of the effective memory address given by the operand address.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	B

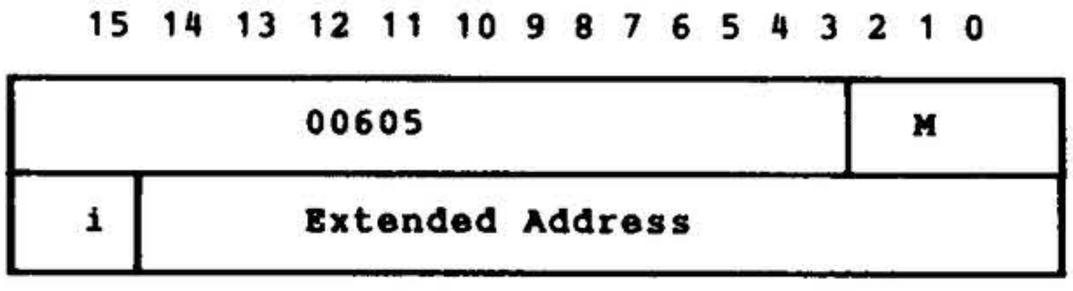
LDXE Load X Register Extended



Loads into the X register the contents of the effective memory address given by the operand address.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	X

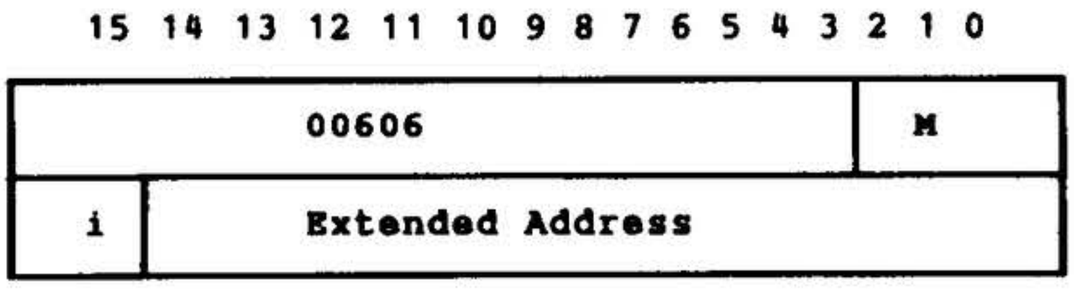
STAE Store A Register Extended



Stores the contents of the A register in the effective memory address given by the operand address.

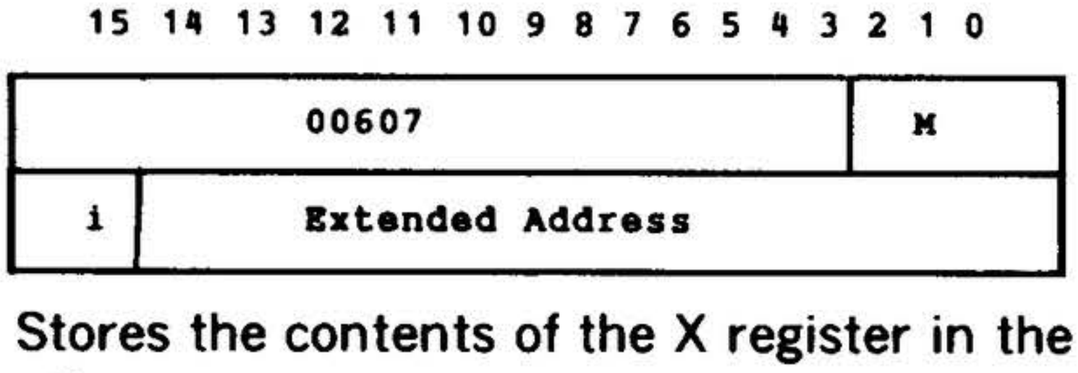
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Effective memory address

STBE Store B Register Extended



Stores the contents of the B register in the effective memory address given by the operand address.

STXE Store X Register Extended



Stores the contents of the X register in the effective memory address given by the operand address.

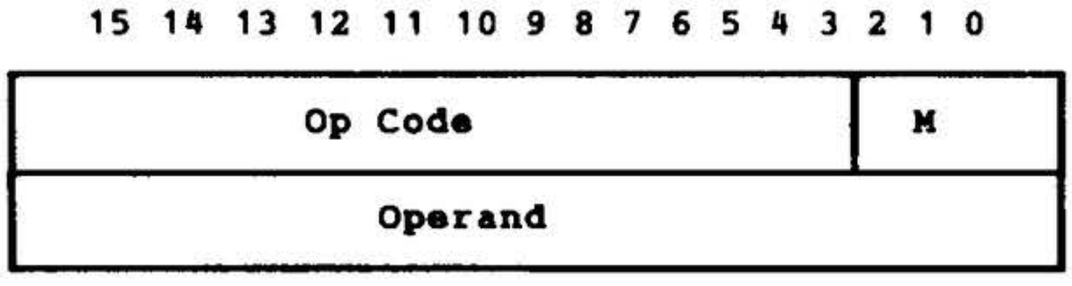
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	Effective memory address

Immediate Load/Store Instructions

This group includes:

Mnemonic	Instruction
LDAI	Load A register immediate
LDBI	Load B register immediate
LDXI	Load X register immediate
STAI	Store A register immediate
STBI	Store B register immediate
STXI	Store X register immediate

These instructions have the following two-word nonaddressing format.



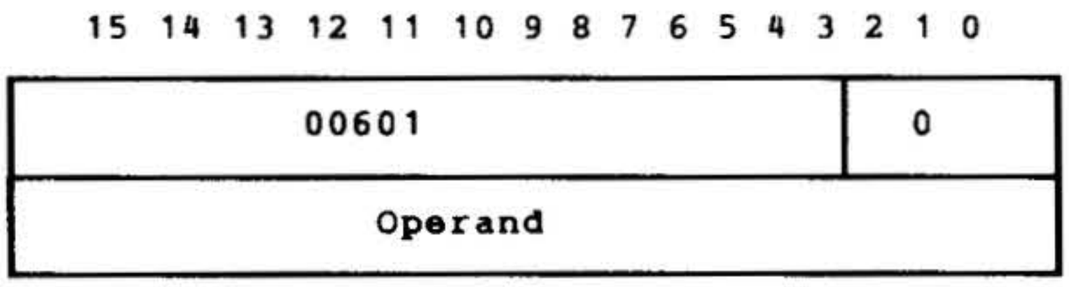
These instructions have configuration 0 000 110 000 in bits 6-15. Bits 3-5 specify

the manipulation. Note that the configuration of these bits is the same as that of the operation code for the corresponding one-word load/store instruction.

The M field is 000. This mode precludes relative or indirect addressing or indexing.

The second word of the instruction is always the data to be processed. It cannot be an address. In immediate addressing instructions the address is the second word for a storage operation thus, in LDAI, the contents of the second word are loaded into the A register.

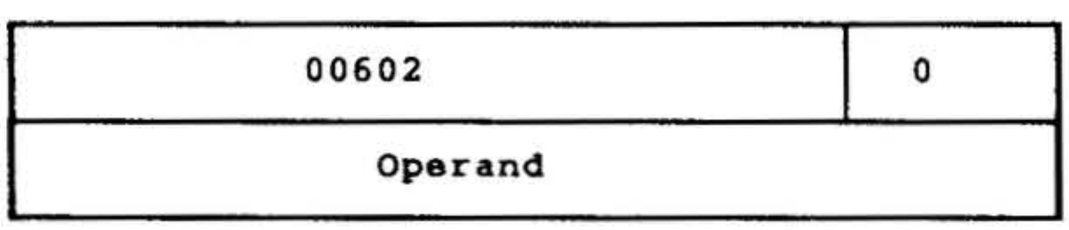
LDAI Load A Register Immediate



Loads into the A register the contents of the operand.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

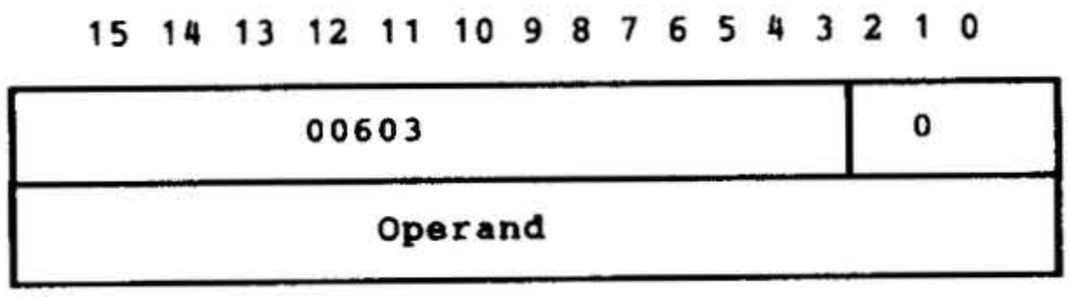
LDBI Load B Register Immediate



Loads into the B register the contents of the operand.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

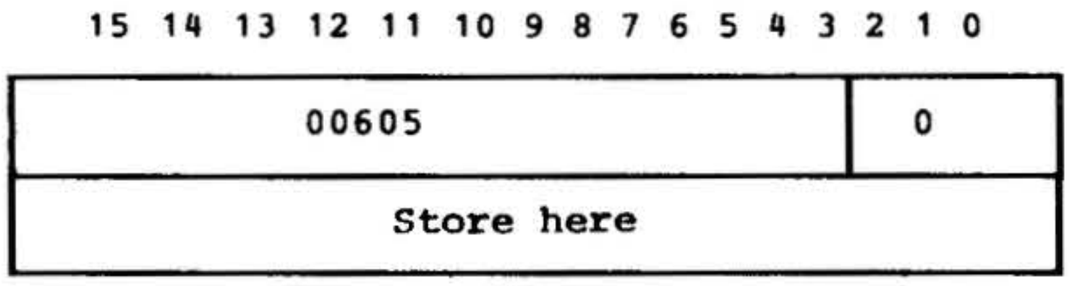
LDXI Load X Register Immediate



Loads into the X register the contents of the operand.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	X

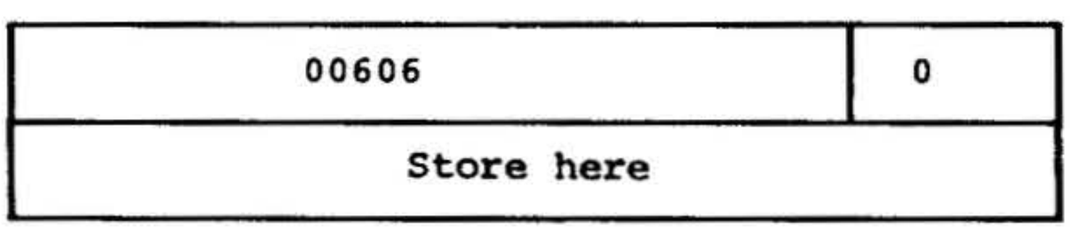
STAI Store A Register Immediate



Stores the contents of the A register in the second word.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

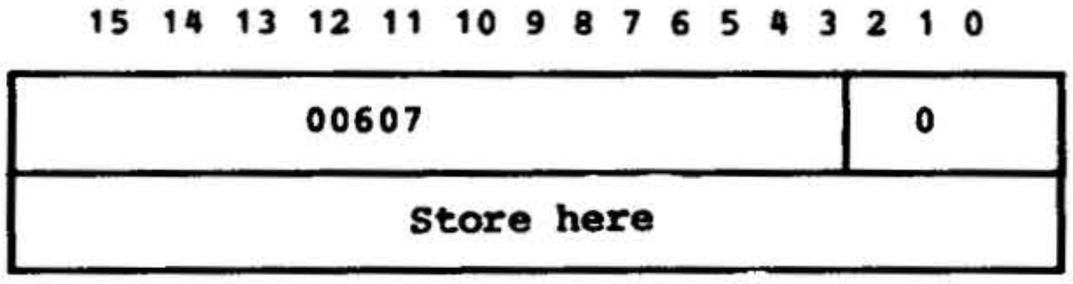
STBI Store B Register Immediate



Stores the contents of the B register in the second word.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

STXI Store X Register Immediate



Stores the contents of the X register in the second word.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

Arithmetic Instructions

This group comprises the instructions for incrementation of the contents of a memory address for performing the arithmetic functions of addition, subtraction, multiplication, and division; and the extended- and immediate-addressing counterparts of these instructions.

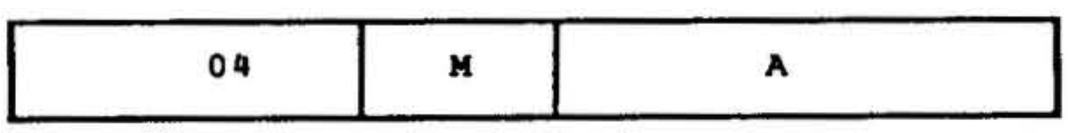
Mnemonic	Instruction
INR	Increment memory and replace
INRE	Increment memory and replace extended
INRI	Increment memory and replace immediate
ADD	Add memory to A register
ADDE	Add memory to A register extended
ADDI	Add memory to A register immediate
SUB	Subtract memory from A register
SUBE	Subtract memory from A register extended
SUBI	Subtract memory from A register immediate
MUL	Multiply
MULE	Multiply extended
MULI	Multiply immediate
DIV	Divide
DIVE	Divide extended
DIVI	Divide immediate

In the one-word instructions, bits 12-15 specify the arithmetic function:

0	100	Increment
1	010	Add
1	100	Subtract
1	110	Multiply
1	111	Divide

In the two-word instructions, the same configurations appear in bits 3-6 of the first word.

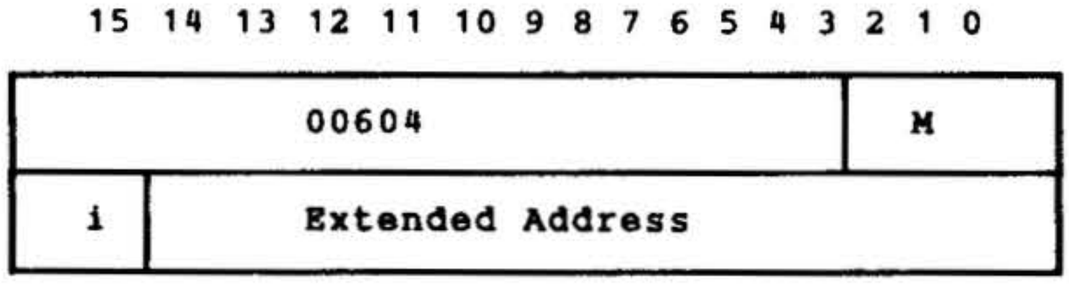
INR Increment Memory and Replace



Increments (by one) the contents of the effective memory address. Sets the overflow indicator (OF) if the maximum positive number (077777) is exceeded. The value in the memory address is then negative (0100000).

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Registers altered: Memory and OF

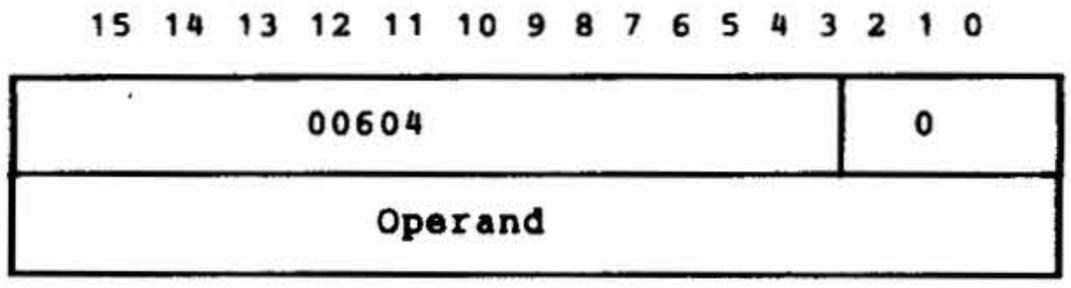
INRE Increment Memory and Replace Extended



Increments (by one) the contents of the effective memory address designated by the operand in the second word and sets OF as for INR.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Registers altered: Memory and OF

INRI Increment and Replace Immediate

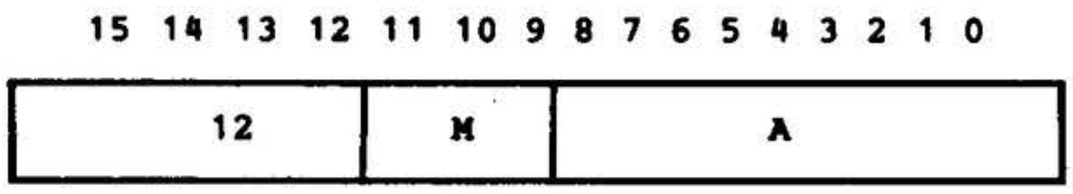


Increments (by one) the operand in the second word and sets OF as for INR.

INSTRUCTION SET

Relative addressing: No
Indirect addressing: No
Indexing: No
Registers altered: Memory and OF

ADD **Add Memory to A Register**

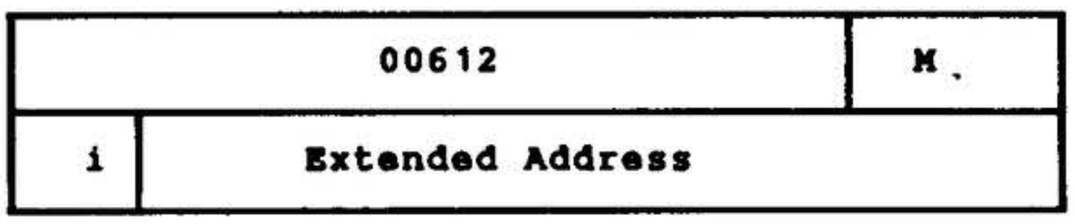


Adds the contents of the effective memory address to the contents of the A register, places the sum in the A register. If the value of the A register is 077777 (maximum positive number) and any non-zero positive number (x) is added to it, OF is set and the sign (bit 15) is set to one. The contents of bits 0 to 14 will be x – 1.

For example, adding 000002 to 077777 results in OF being set to one and the A register containing 100001 (octal).

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A and OF

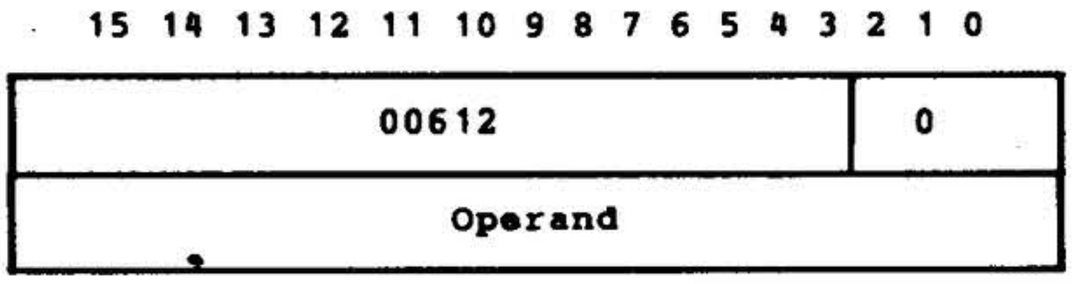
ADDE **Add to A Register Extended (Optional)**



Similar to ADD using as addends the contents of the A register and the contents of the effective memory address designated by the operand address in the second word.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A and OF

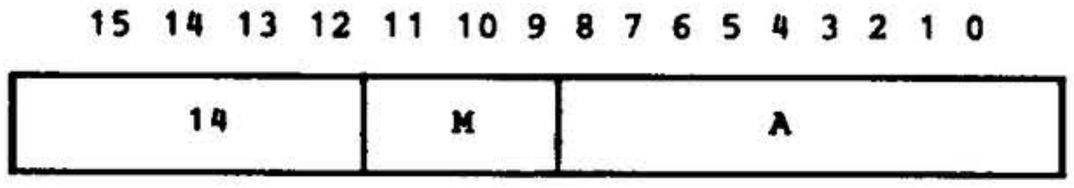
ADDI **Add to A Register Immediate**



Similar to ADD using as addends the contents of the A register and the operand in the second word.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A and OF

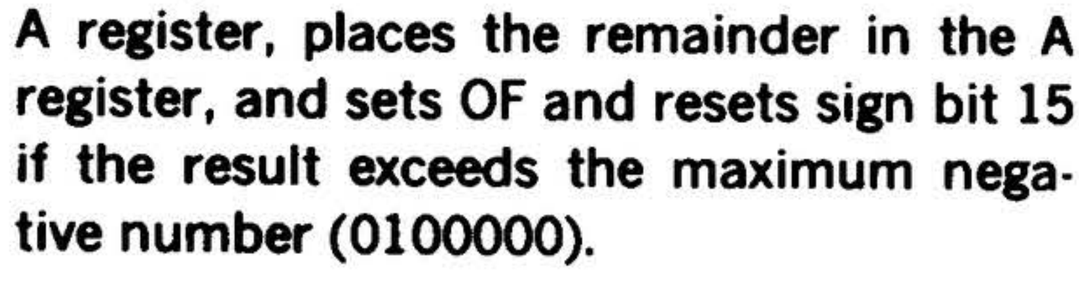
SUB **Subtract Memory From A Register**



Subtracts the contents of the effective memory address from the contents of the A register, places the remainder in the A register, and sets OF and resets sign bit 15 if the result exceeds the maximum negative number (0100000).

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A and OF

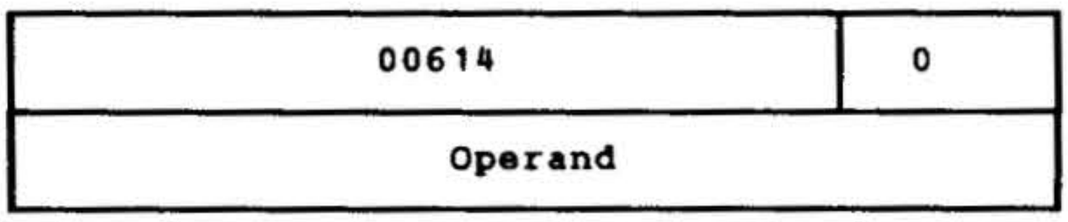
SUBE **Subtract From A Register Exteneed**



Similar to SUB using as subtrahend the effective memory address designated by the operand address in the second word.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A and OF

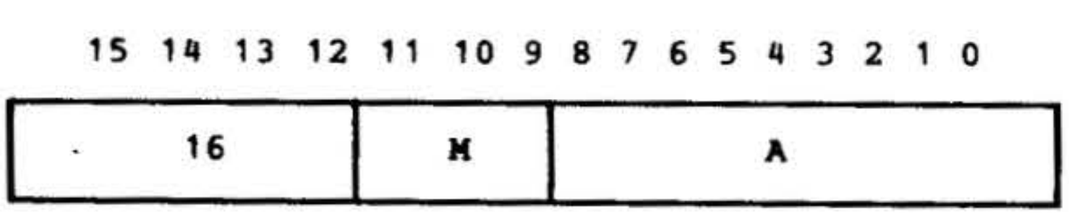
SUBI **Subtract From A Register Immediate**



Similar to SUB using as subtrahend the operand in the second word.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A and OF

MUL **Multiply**

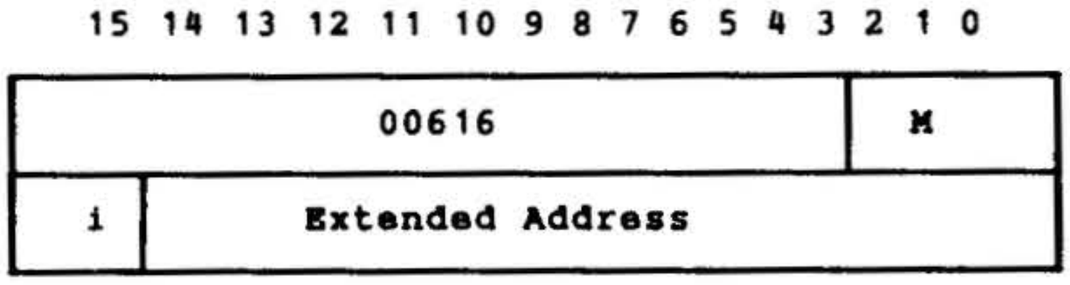


Multiplies the contents of the effective memory address by the contents of the B register, adds the contents of the A register to the product, and places the result in the A and B registers with the most significant portion in the A register. The sign bit of the A register gives the sign of the result. The sign bit of the B register is reset to zero. OF sets if the original contents of the B register and the effective memory address were the greatest possible negative number and the original contents of the A register were positive.

The algorithm is: R · B + A A,B.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A, B, and OF

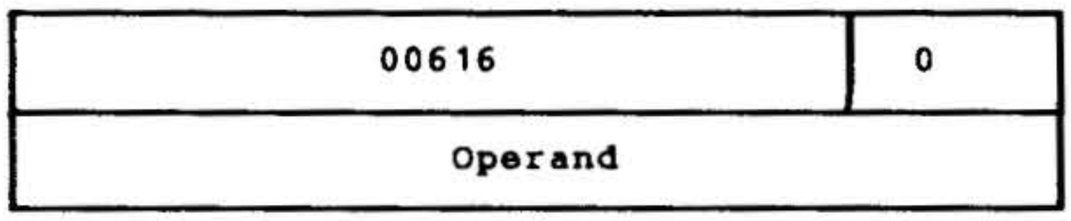
MULE **Multiply Extended**



Similar to MUL except that the multiplicand is the contents of the effective memory address designated by the operand and address in the second word.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A, B, and OF

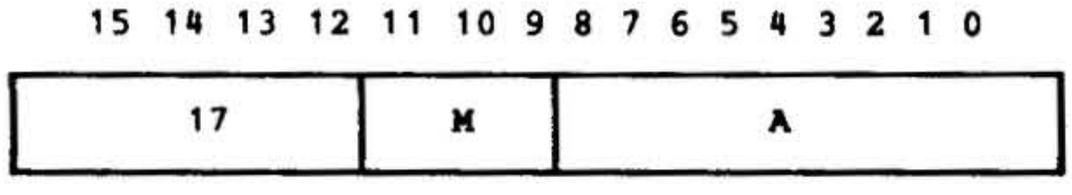
MULI **Multiply Immediate**



Similar to MUL using as multiplicand the operand in the second word.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A, B, and OF

DIV **Divide**

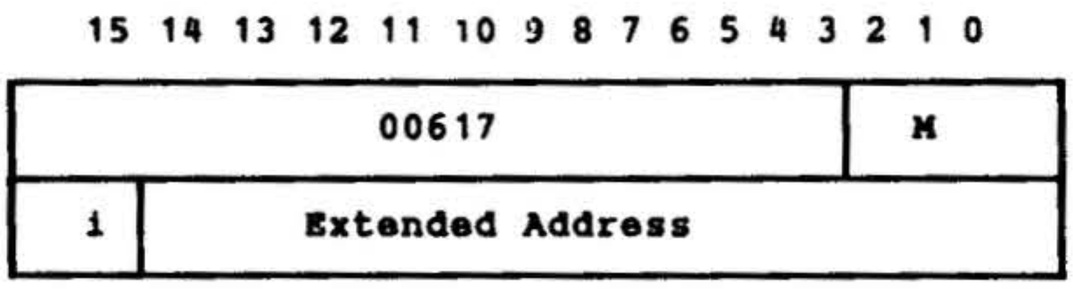


Divides the combined contents of the A and B registers by the contents of the effective memory address, and places the signed quotient in the B register and the remainder (with the sign of the dividend) in the A register. Sets OF if necessary indicating an invalid and unpredictable

quotient; overflow cannot occur if the divisor is greater than the dividend.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A, B, and OF

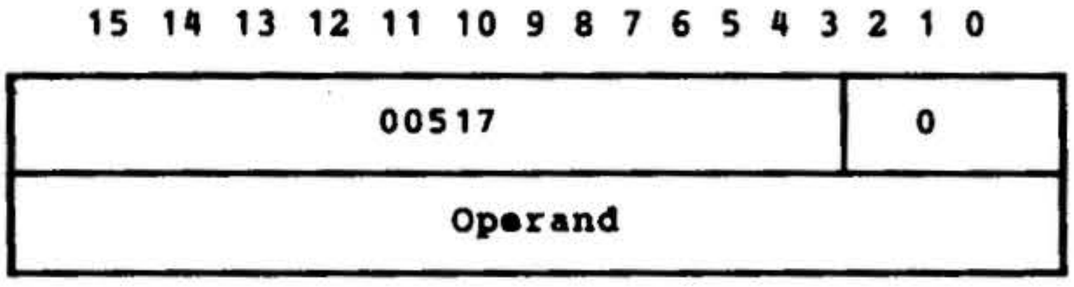
DIVE — Divide Extended



Similar to DIV using as divisor the contents of the effective memory address designated by the operand address in the second word.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A, B, and OF

DIVI — Divide Immediate



Similar to DIV using as divisor the operand in the second word.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A, B, and OF

Logic Instructions

This group comprises the inclusive-OR, exclusive-OR, and AND instructions and their extended- and immediate-addressing counterparts.

Mnemonic	Instruction
ORA	Inclusive-OR memory and A register
ORAE	Inclusive-OR extended
ORAI	Inclusive-OR immediate
ERA	Exclusive-OR memory and A register
ERAE	Exclusive-OR Extended
ERAI	Exclusive-OR immediate
ANA	AND memory and A register
ANAE	AND extended
ANAI	AND immediate
SRE	Skip if register equal

In the one-word instructions, bits 12-15 specify the logic function:

1	001	Inclusive-OR
1	011	Exclusive-OR
1	101	AND

In the two-word instructions, the same configurations appear in bits 3-6 of the first word.

ORA **Inclusive-OR Memory and A Register**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

11	M	A

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for inclusive-OR shown below. In the truth table, n = bit position.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A

Inclusive-OR Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	1
1	0	1
1	1	1

ORAE **Inclusive-OR Extended**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

00611	M
Extended Address	

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the effective memory address given in the second word, and places the result in the A register according to the truth table above.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A

ORAI **Inclusive-OR Immediate**

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

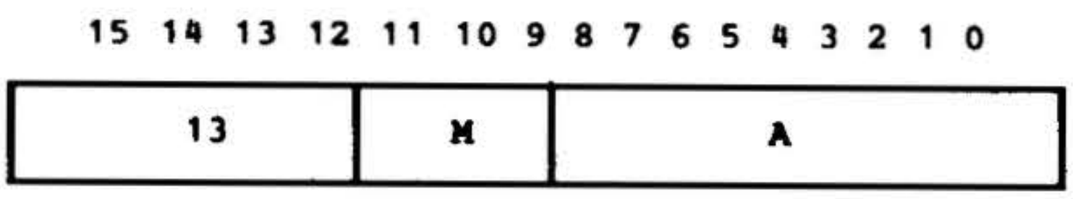
00611	0
Operand	

Performs an inclusive-OR between each bit of the A register and the corresponding bit of the operand, in the second word, and

the result in the A register according to the truth table above.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

ERA **Exclusive-OR Memory and A Register**



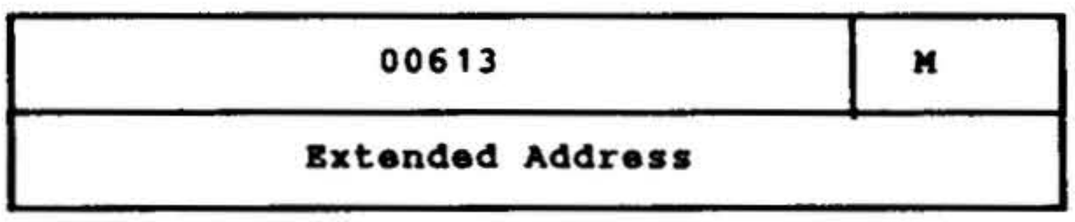
Performs an exclusive-OR between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for exclusive-OR shown below. In the truth table, n = bit position.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

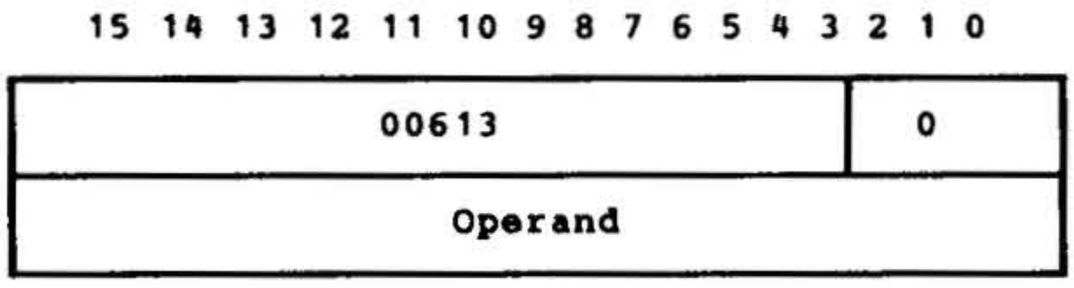
Exclusive-OR Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	1
1	0	1
1	1	0

ERAE **Exclusive-OR Extended**



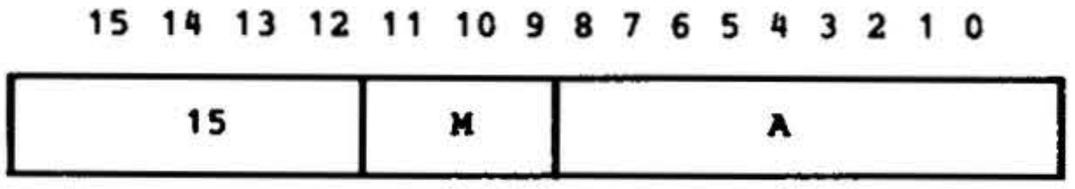
ERAI **Exclusive-OR Immediate**



Performs an exclusive-OR between each bit of the A register and the corresponding bit of the operand in the second word, and places the result in the A register according to the truth table above.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

ANA **AND Memory and A register**



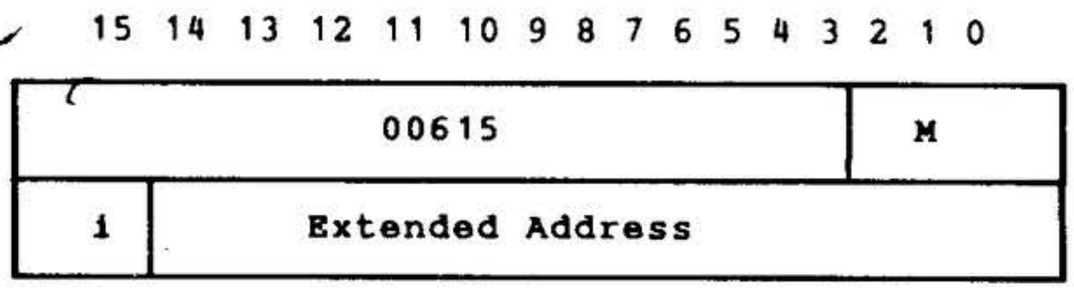
Performs an AND between each bit of the A register and the corresponding bit of the effective memory address, and places the result in the A register according to the truth table for AND shown below. In the truth table, n = bit position.

Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Yes
Register altered:	A

AND Truth Table

Condition		Result
A(n)	Effective Memory Address (n)	A(n)
0	0	0
0	1	0
1	0	0
1	1	1

ANAE **AND Extended**



Performs an AND between each bit of the A register and the corresponding bit of the effective memory address given in the second word, and places the result in the A register according to the truth table above.

Relative addressing: Yes
Indirect addressing: Yes
Indexing: Yes
Register altered: A

ANAI **AND Immediate**

Performs an AND between each bit of the A register and the corresponding bit of the operand in the second word, and places the result in the A register according to the truth table above.

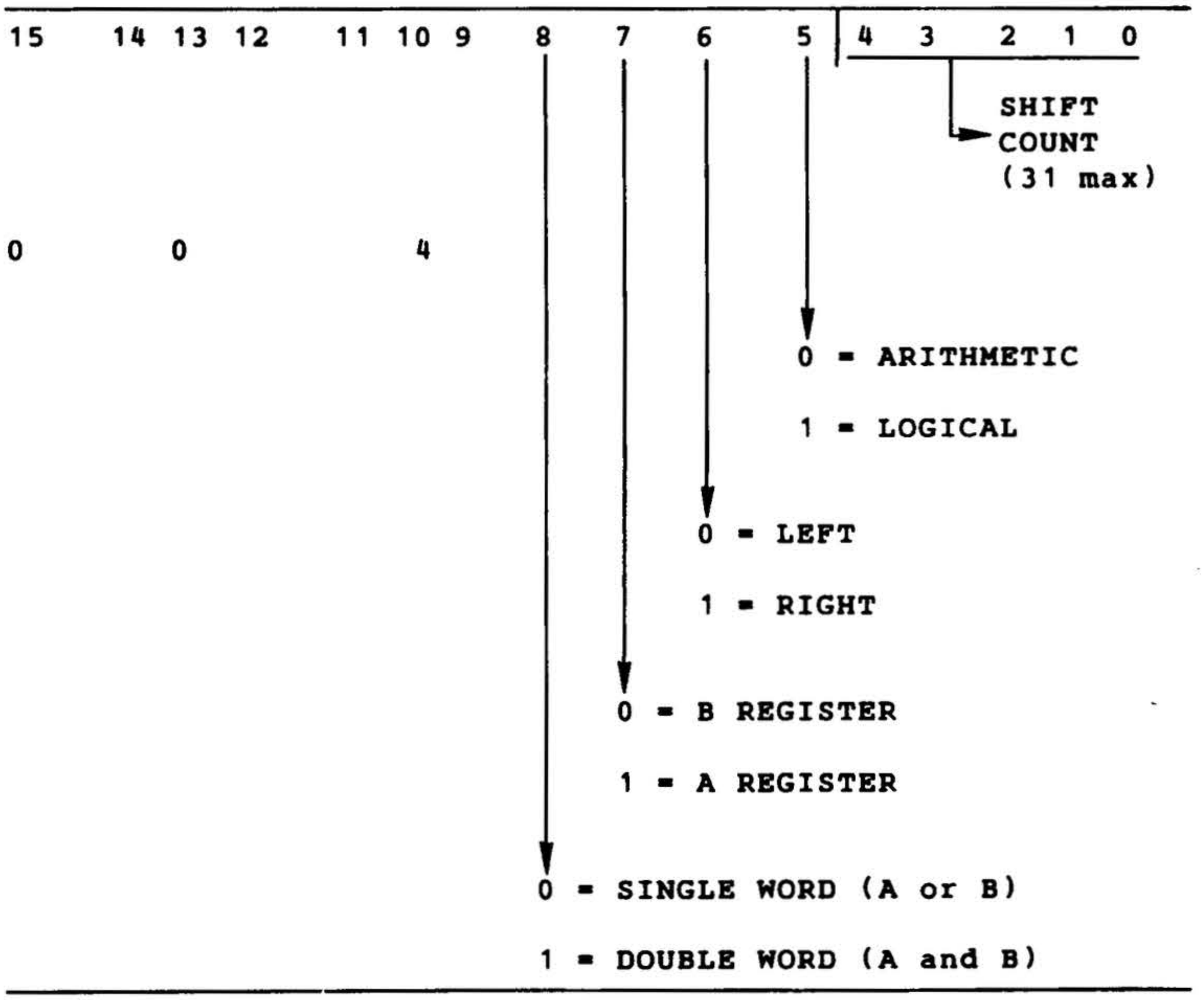
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A

Shift/Rotation Instructions

This group comprises the instructions that shift or rotate the contents of registers. In shifts, the bits shifted out of the register are lost; but in rotation, they are loaded one at a time into the opposite end of the register.

Mnemonic	Instruction
LSRA	Logical shift right A register
LSRB	Logical shift right B register
LRLA	Logical rotation left A register
LRLB	Logical rotation left B register
LLSR	Long logical shift right
LLRL	Long logical rotation left
ASRA	Arithmetic shift right A register
ASRB	Arithmetic shift right B register
ASLA	Arithmetic shift left A register
ASLB	Arithmetic shift left B register
LASR	Long airthmitic shift right
LASL	Long arithmetic shift left

These instructions have the following one-word nonaddressing format:



These instructions have configuration 0 000 100 in bits 9-15. Bits 0-4 specify the number of bits to be shifted/rotated. Bits 5-8 specify the parameters of the shift/ rotation operation, as follows:

Bit 5 = 0	Arithmetic shift
Bit 5 = 1	Logical shift/ rotation
Bit 6 = 0	Left shift/rotation
Bit 6 = 1	Right shift/rotation
Bit 7 = 0	B register shift/ rotation
Bit 7 = 1	A register shift/ rotation
Bit 8 = 0	Shift/rotate one register only
Bit 8 = 1	Long shift/roatation

Rotations occur when bit 5 = 1 and bit 6 = 0; otherwise, there is a shift. Note that for long shifts/rotations, bit 7 = 0.

LSRA Logical Shift Right A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004340 + x

Shifts the contents of the A register x places (x = 0 to 037) to the right and loads the vacated high-order bit(s) with zeros. Information shifted out of the low-order bit(s) is lost.

LSRB Logical Shift Right B Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004140 + x

Shifts the contents of the B register x places (x = 0 to 037) to the right and loads the vacated high-order bit(s) with zeros. Information shifted out of the low-order bit(s) is lost.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: B

LRLA Logical Rotate Left A Register

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

004240 + x

Rotates the contents of the A register x places (x = 0 to 037) to the left. Bit 0 receives each high-order bit as it is rotated out during the execution.

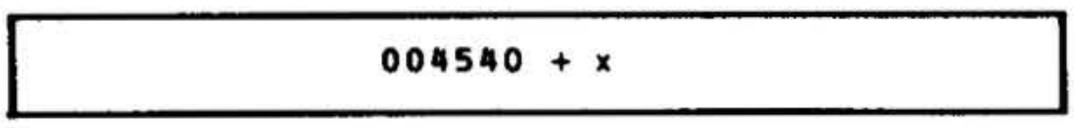
Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: A

LRLB Logical Rotate Left B Register

Rotates the contents of the B register x places (x = 0 to 037) to the left. Bit 0 receives each high-order bit as it is rotated out during the execution.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

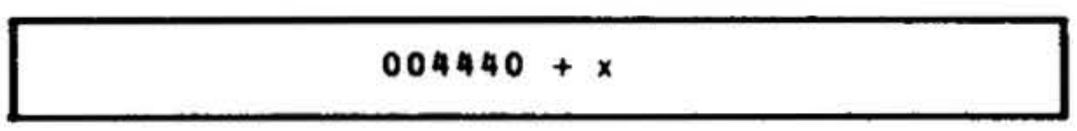
LLSR Long Logical Shift Right



Shifts the contents of the A and B registers x places (x = 0 to 037) to the right. Loads the vacated high-order bit(s) of the A register with zeros. Information shifted out of the low-order bit(s) of the B register is lost.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

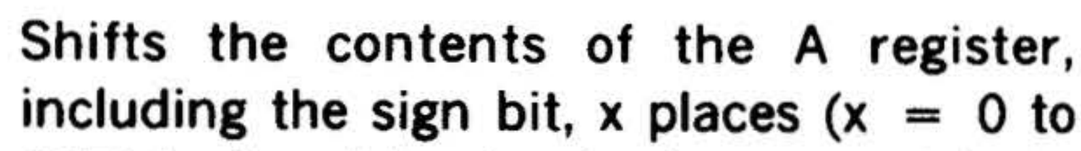
LLRL Long Logical Rotation Left



Rotates the contents of the A and B registers x places (x = 0 to 037). Bit 0 of the B register receives each high-order bit of the A register to the left as it is rotated out during the execution.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

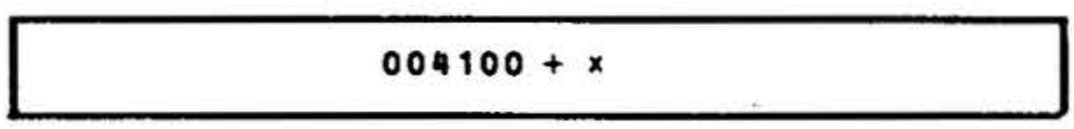
ASRA Arithmetic Shift Right A Register



Shifts the contents of the A register, including the sign bit, x places (x = 0 to 037) to the right. Loads the vacated high-order bit(s) with the value of the sign bit. Information shifted out of the low-order bit(s) is lost.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

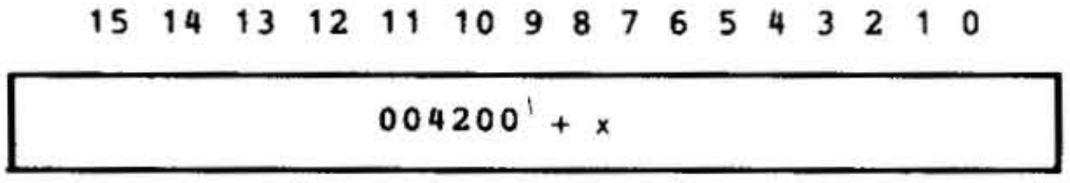
ASRB Arithmetic Shift Right B Register



Shifts the contents of the B register, including the sign bit, x places (x = 0 to 037) to the right. Loads the vacated high-order bit(s) with the value of the sign bit. Information shifted out of the low-order bit(s) is lost.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

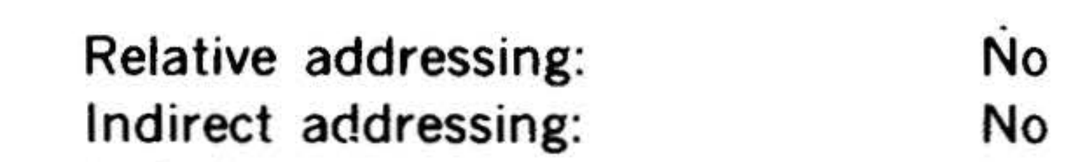
ASLA **Arithmetic Shift Left A Register**



Shifts the contents of the A register x places (x = 0 to 037) to the left. Loads the vacated low-order bit(s). Information shifted out of the high-order bit(s) is lost, but the sign bit is unaffected.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

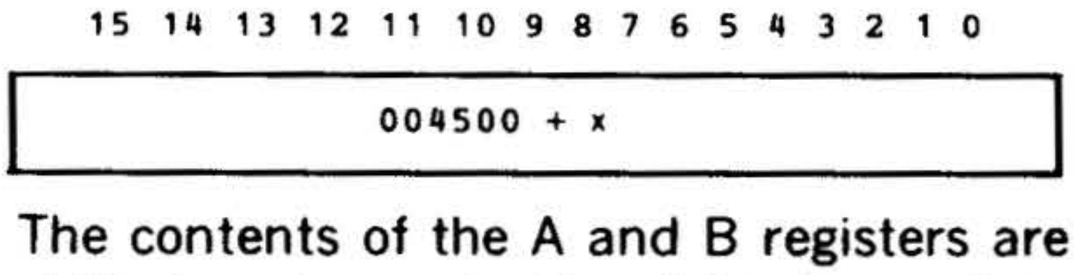
ASLB **Arithmetic Shift Left B Register**



Shifts the contents of the B register x places (x = 0 to 037) to the left loads the vacated low-order bit(s) with zero(s). Information shifted out of the high-order bit(s) is lost, but the sign bit is unaffected.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

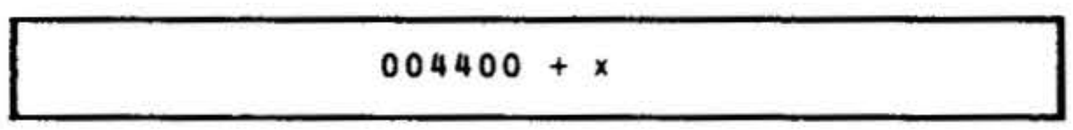
LASR **Long Arithmetic Shift Right**



The contents of the A and B registers are shifted x places to the right (x = 0 to 037). Bit position 0 of the A register is shifted into bit position 14 of the B register. The sign bit of the A register (bit 15) is extended x places to the right. The sign bit (bit 15) of the B register remains unchanged.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

LASL **Long Arithmetic Shift Left**



The contents of the A and B registers are shifted x places to the left (x = 0 to 037). Bit position 14 of the B register is shifted into bit position 0 of the A register; the sign bit (bit 15) of the B register remains unchanged. The sign bit (bit 15 of the A register also remains unchanged. The bit shifted out of bit position 14 of the A register is lost, and zeros are shifted into the low-order positions of the B register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A and B

Register Transfer/Modification Instructions

This group comprises the unmodified register transfers; instructions for incrementing, decrementing, and complementing registers and for adjusting the contents of registers with the overflow indicator; and microcoded combinations of these operations.

Overflow Indicator

An overflow condition is detected by the following steps:

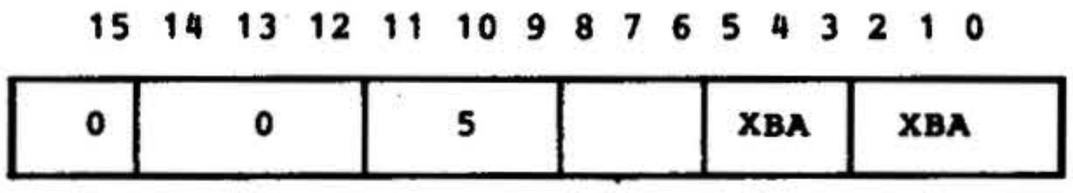
a. Ensure computer is in the step mode.

b. Actuate the STATUS switch.

c. If bit 8 of the register display indicators is on, an overflow condition exists.

d. If the indicator is off, overflow does not exist.

These instructions have the following one-word pseudoaddressing format:



This group of instructions does not address memory. However, this format is included in the one-word addressing grouping because registers are, in effect, addressed and their contents modified by the operations, much as in operations involving memory.

These instructions have configuration 0 000 101 in bits 9-15.

Bit 8 specifies conditional execution of the instruction depending upon the overflow indicator:

0cc	Execute instruction unconditionally
1cc	Execute only if the overflow indicator is set

Bits 6 and 7 specify that data be transferred from one register to another as follows:

w00	Unmodified
w01	Incremented
w10	Complemented
w11	Decremented

Bits 3-5 specifies the location from which the data to be transferred or modified are obtained (source register):

000	Clears destination register
001	A register
010	B register
100	X register

Bits 0-2 specify the location in which the modified and/or transferred data are placed (destination register):

000	No operation
001	A register
010	B register
100	X register

Additional transfers and modifications can be microcoded. Thus, 110 in bits 0-2 places data in both the X and B registers. If bits 3-5 specify more than one source register, the result is the inclusive-OR of the group of registers.

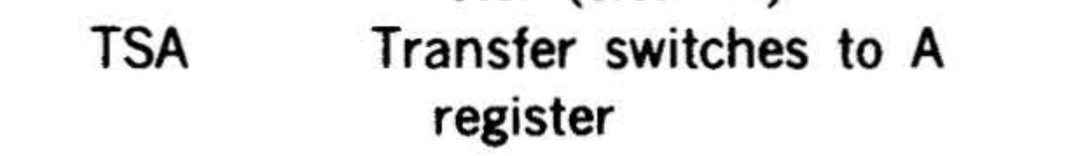
The instructions of this group are summarized immediately preceding the appropriate subgrouping.

Unmodified Register Transfer

Instructions

Mnemonic	Instruction
TAB	Transfer A register to B register
TAX	Transfer A register to X register
TBA	Transfer B register to A register
TBX	Transfer B register to X register
TXA	Transfer X register to A register
TXB	Transfer X register to B register
TZA	Transfer zeros to A register (clear A)
TZB	Transfer zeros to B register (clear B)
TZX	Transfer zeros to X register (clear X)
TSA	Transfer switches to A register

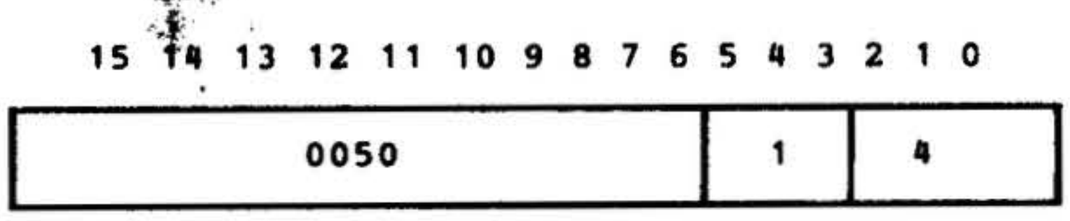
TAB Transfer A Register to B Register



Transfers the contents of the A register to the B register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

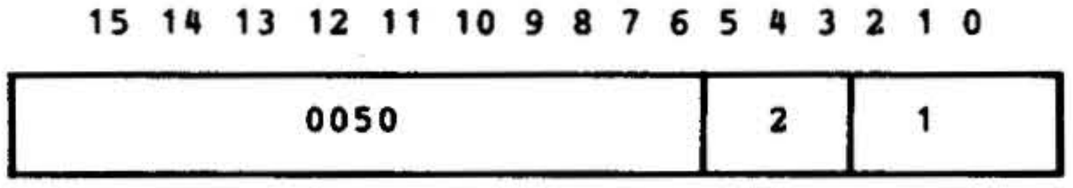
TAX Transfer A Register to X Register



Transfers the contents of the A register to the X register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	X

TBA Transfer B Register to A Register



Transfers the contents of the B register to the A register.

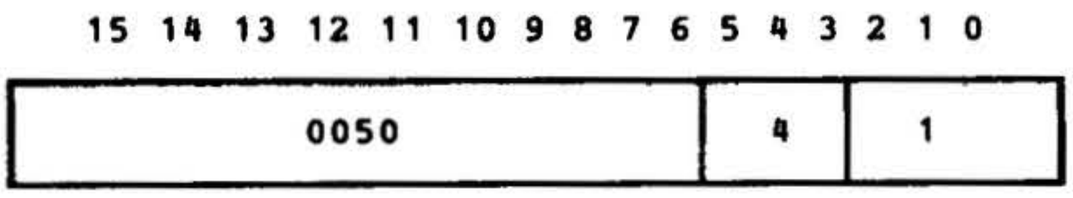
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

TBX Transfer B Register to X Register

Transfers the contents of the B register to the X register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	X

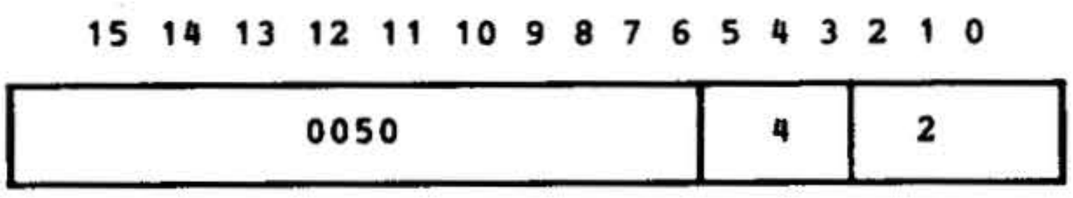
TXA Transfer X Register to A Register



Transfers the contents of the X register to the A register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

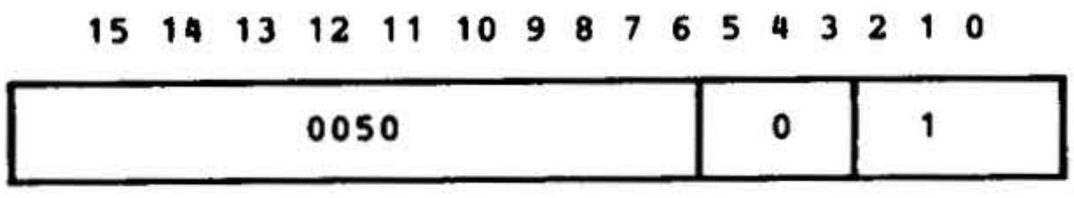
TXB Transfer X Register to B Register



Transfers the contents of the X register to the B register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

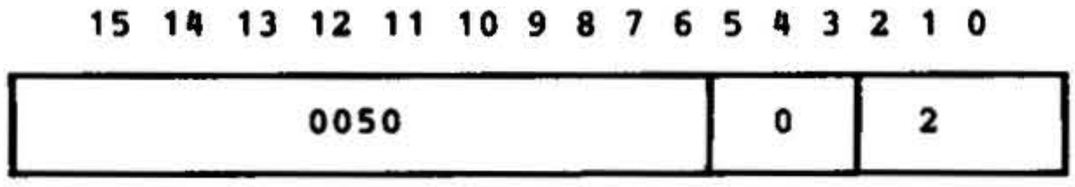
TZA Transfer Zeros to A Register (Clear A)



Clears the A register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

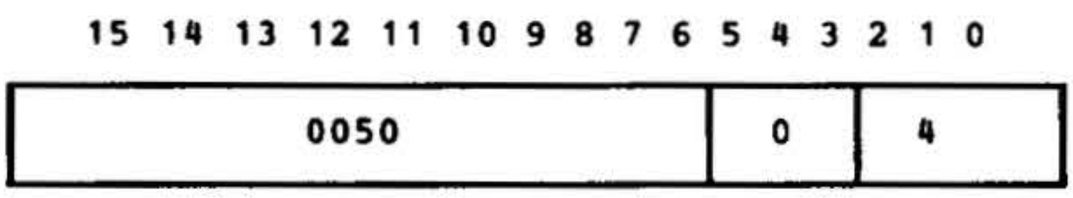
TZB Transfer Zeros to B Register (Clear B)



Clears the B register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	B

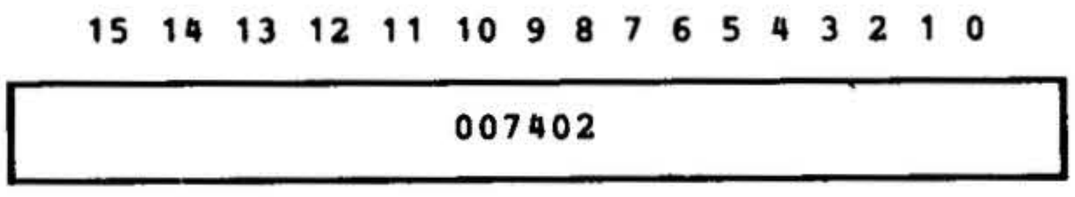
TZX Transfer Zeros to X Register (Clear X)



Clears the X register.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	X

TSA Transfer Switches to A Register



Transfers the contents of the register entry switches to the A register.

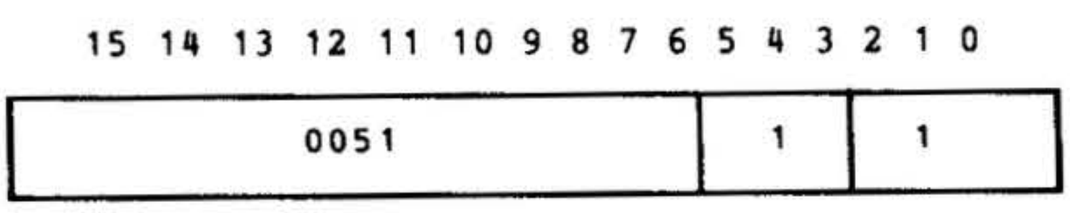
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	A

Register Modification Instructions

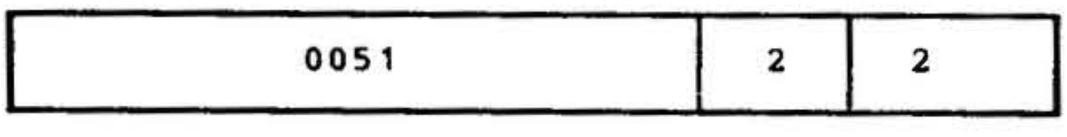
Mnemonic	Instruction
IAR	Increment A register
IBR	Increment B register
IXR	Increment X register
DAR	Decrement A register
DBR	Decrement B register
DXR	Decrement X register
CPA	Complement A register
CPB	Complement B register
CPX	Complement X register
AOFA	Increment A register if overflow indicator set
AOFB	Increment B register if overflow indicator set
AOFX	Increment X register if overflow indicator set
SOFA	Decrement A register if overflow indicator set
SOFB	Decrement B register if overflow indicator set
SOFX	Decrement X register if overflow indicator set

Note in the following examples that one instruction in each grouping shows the overflow conditional execution that is also applicable to other instructions in the group.

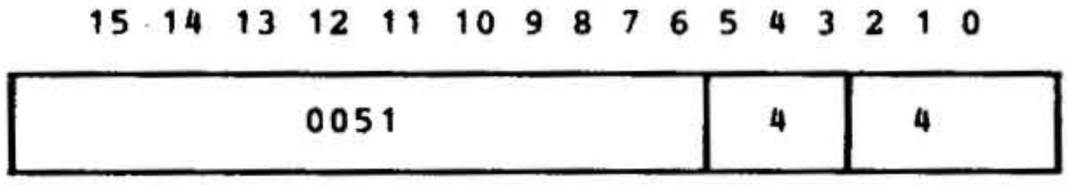
IAR Increment A Register



IBR Increment B Register



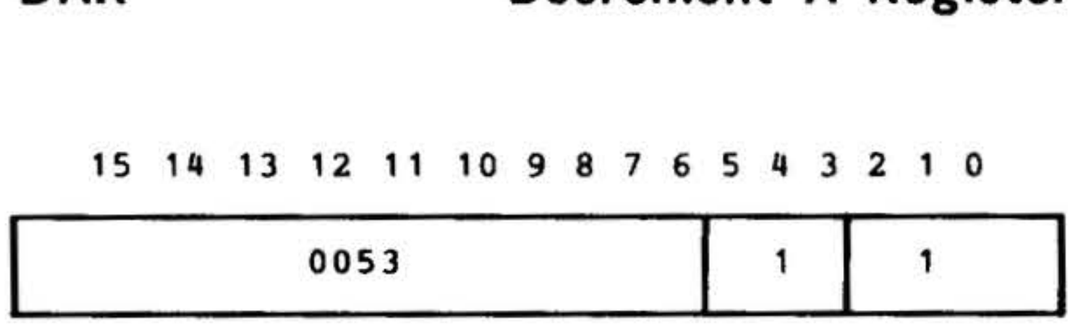
IXR Increment X Register



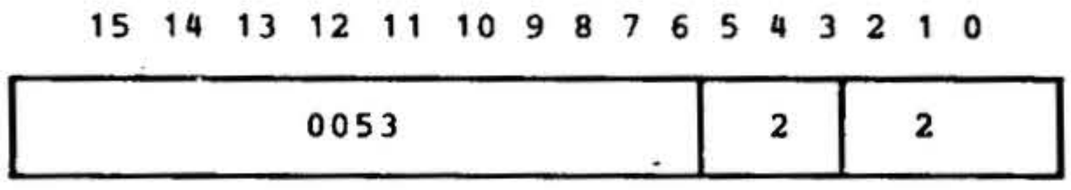
Increments (by one) the contents of the specified register. Sets the overflow indicator (OF) if the register increments the maximum positive number (077777); changes the contents to the maximum negative number (0100000).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF and the register specified by the second letter of the mnemonic

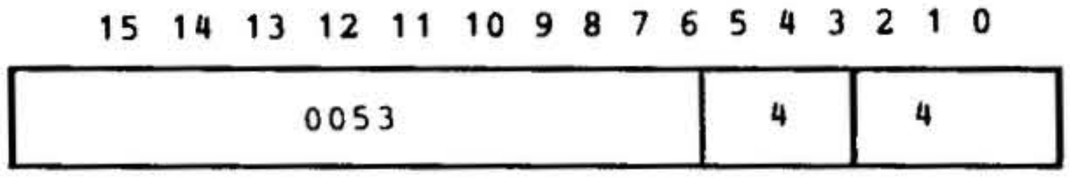
DAR Decrement A Register



DBR Decrement B Register



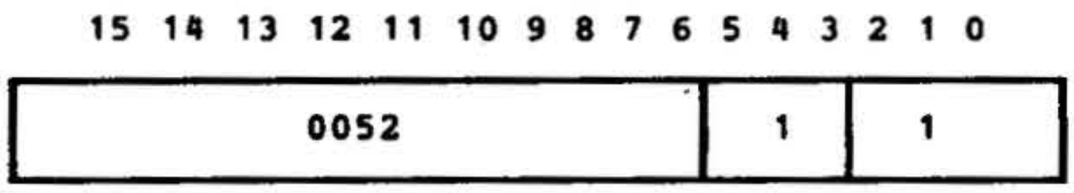
DXR Decrement X Register



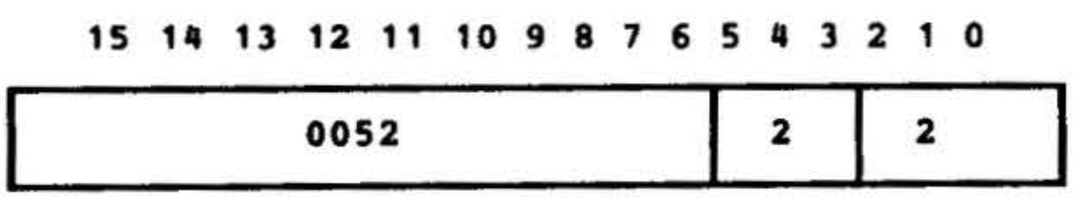
Decrements (by one) the contents of the specified register. Sets OF if the register contents are 0100000 when executed and changes the contents to 077777.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: OF and the register specified by the second letter of the mnemonic

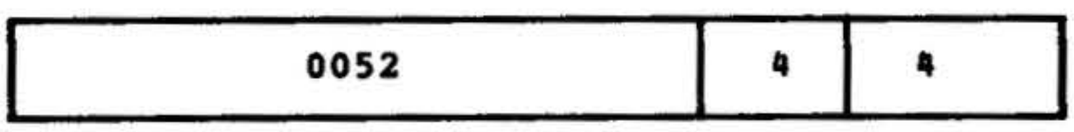
CPA **Complement A Register**



CPB **Complement B Register**



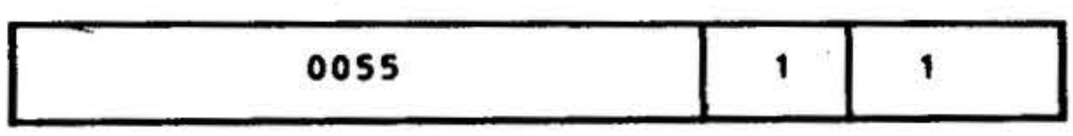
CPX **Complement X Register**



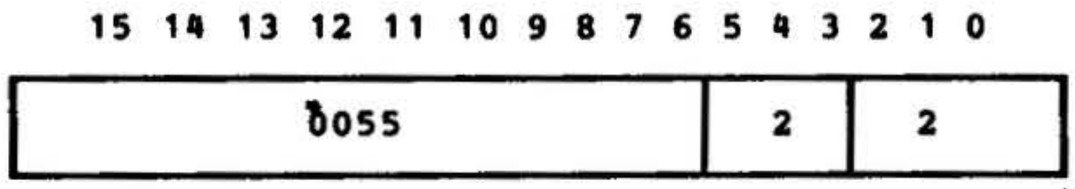
One's-complements the contents of the specified register.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: The register specified by the third letter of the mnemonic

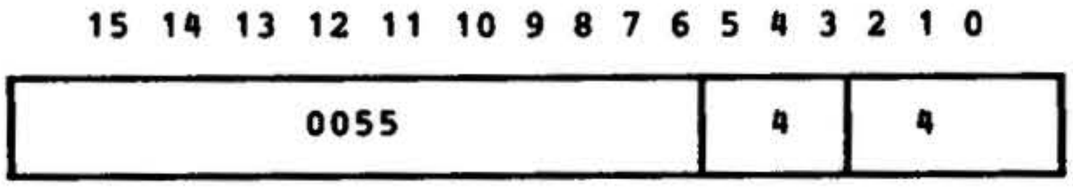
AOFA **Increment A Register if Overflow Indicator Set**



AOFB **Increment B Register if Overflow Indicator Set**



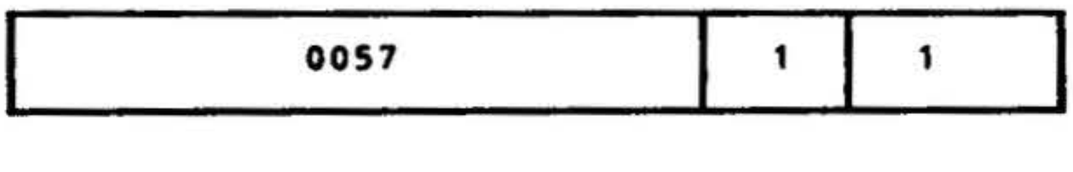
AOFX **Increment X Register if Overflow Indicator Set**



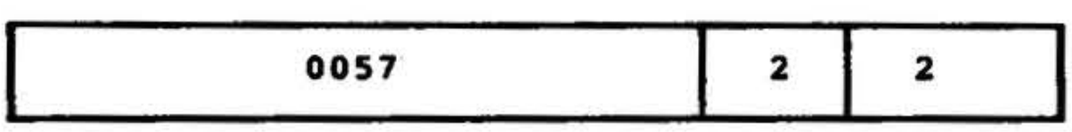
Adds one to the contents of the specified register only if OF is set. Does not change the setting of OF.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: The register specified by the fourth letter of the mnemonic

SOFA **Decrement A Register if Overflow Indicator Set**



SOFB **Decrement B Register if Overflow Indicator Set**



SOFX **Decrement X Register if Overflow Indicator Set**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0057											4			4	

Subtracts one from the contents of the specified register only if OF is set. Does not change the setting of OF.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	The register specified by the fourth letter of the mnemonic

Combined Register Transfer/Modification Instructions

These instructions are used to specify combinations of the register transfer and modification instructions to perform simultaneous multiple operations. The combined conditions are established in the variable field of the DAS assembler statement (section 15) when the program is written.

Mnemonic	Instruction
MERG	Merge source to destination registers
INCR	Increment source to destination registers
DECR	Decrement source to destination registers
COMP	Complement source to destination registers
ZERO	Zero (clear) registers

Figure 14-1 illustrates the instruction bit parameters.

The instruction bits specify the parameters indicated below.

Figure 14-1. Instruction Bit Meaning

Bits 3-5 specify the source register(s) as follows:

000	Clears destination register(s)
001	A register
010	B register
100	X register

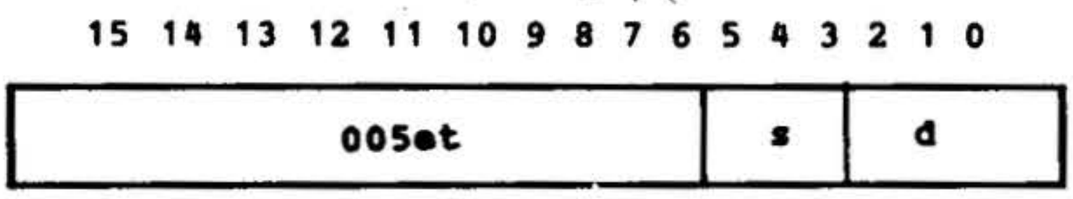
Thus, a configuration of 110 specifies that the contents of the B and X registers are to be inclusively ORed before the transfer/ modification operation is performed.

Bits 0-2 specify the destination register(s) as follows:

000	No operaton
001	A register
010	B register
100	X register

Thus, a configuration of 011 places data in both the A and B registers when the instructions is executed.

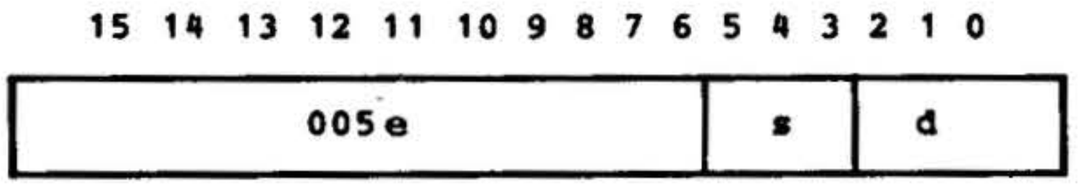
MERG **Merge Source to Destination Registers**



Transfers the inclusive-OR of the contents of the source register(s) to the destination register(s). Bits six through eight are established by the DAS assemblers when specifying these combined register instructions.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Those specified

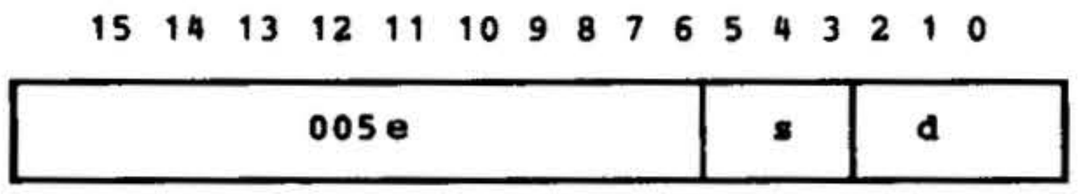
INCR **Increment Source to Destination Registers**



Adds one to the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Those specified

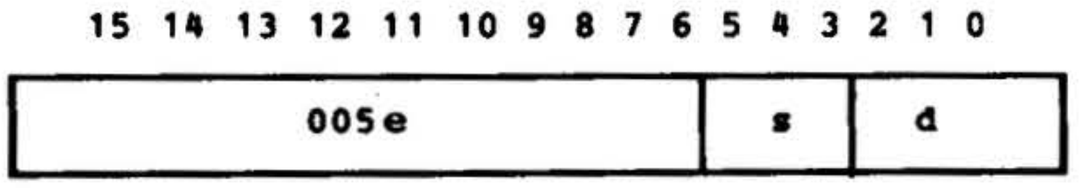
DECR **Decrement Source to Destination Registers**



Subtracts one from the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Those specified

COMP **Complement Source to Destination Registers**



One's-complements the inclusive-OR of the contents of the source register(s), and places the result in the destination register(s).

Relative addressing: No
Indirect addressing: No
Indexing: No
Registers altered: Those specified

ZERO Zero (Clear) Registers

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0			
005	o	o	d

Clears the destination register(s).

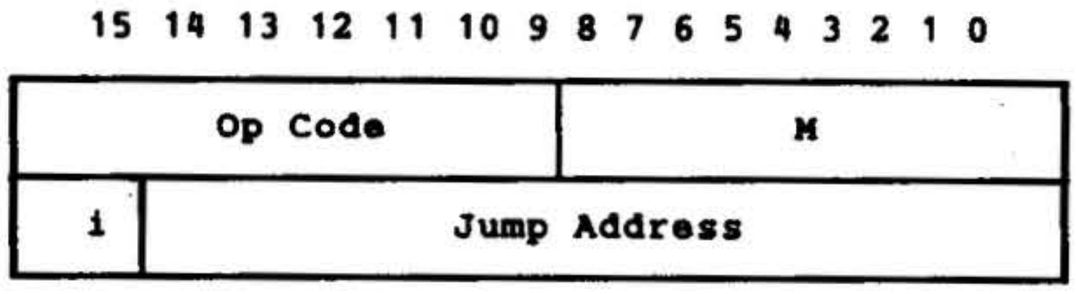
Relative addressing: No
Indirect addressing: No
Indexing: No
Registers altered: Those specified

Jump Instructions

This group comprises the instructions that direct the program to a nonsequential address for execution of the instruction located there, but they neither mark the location (as do the jump-and-mark instructions) nor do they bring the program back to the main program sequence (as do the execution instructions).

Mnemonic	Instruction
JMP	Jump unconditionally
IJMP	Indexed jump
JOF	Jump if overflow indicator set
JOFN	Jump if overflow indicator not set
JAP	Jump if A register positive
JAN	Jump if A register negative
JAZ	Jump if A register zero
JBZ	Jump if B register zero
JXZ	Jump if X register zero
JANZ	Jump if A register not zero
JBNZ	Jump if B register not zero
JXNZ	Jump if X register not zero
JSS1	Jump if SENSE switch 1 set
JSS2	Jump if SENSE switch 2 set
JSS3	Jump if SENSE switch 3 set
JS1N	Jump if SENSE switch 1 not set
JS2N	Jump if SENSE switch 2 not set
JS3N	Jump if SENSE switch 3 not set
JIF	Jump if conditions(s) met
JSR	Jump and return in indexing register
BT	Bit test

These instructions have the following two-word addressing format:



This format comprises a seven-bit operation code (bits 9-15), a nine-bit M field (bits 0-8) in the first word, and an effective jump address in the second.

These instructions have configuration 0 000 001 in bits 9-15. Bits 0-8 specify the jump as follows.

All M field bits are zeros for an unconditional jump instruction. For other instructions in this group, each switch, register, or indicator to be examined is represented by a specific M field bit. Bits 1 and 2 specify the condition of the examined switch, register, or indicator to be met for a jump. If bit 1 or 2 is zero, the jump condition is met if the switch or indicator is set, or if the register contains all zeros. If bits 1 and 2 are one, the jump condition is met if the switch or indicator is not set, or if the register contains any number but positive zero.

If only bit 1 of the M field is set, the jump condition is met when the A register contains a positive value (bit 15 off). If only bit 2 is set, the jump condition is met when the A register contains a negative value (bit 15 set). Note that, in these two cases, the A register bit (bit 3) is zero. Bit 15 in the other registers is not always an arithmetic sign. Therefore, setting only bit 1 or bit 2 does not affect the B or X register since there are no analogous instructions for these registers.

Figure 14-2 illustrates the meaning of the M field bits.

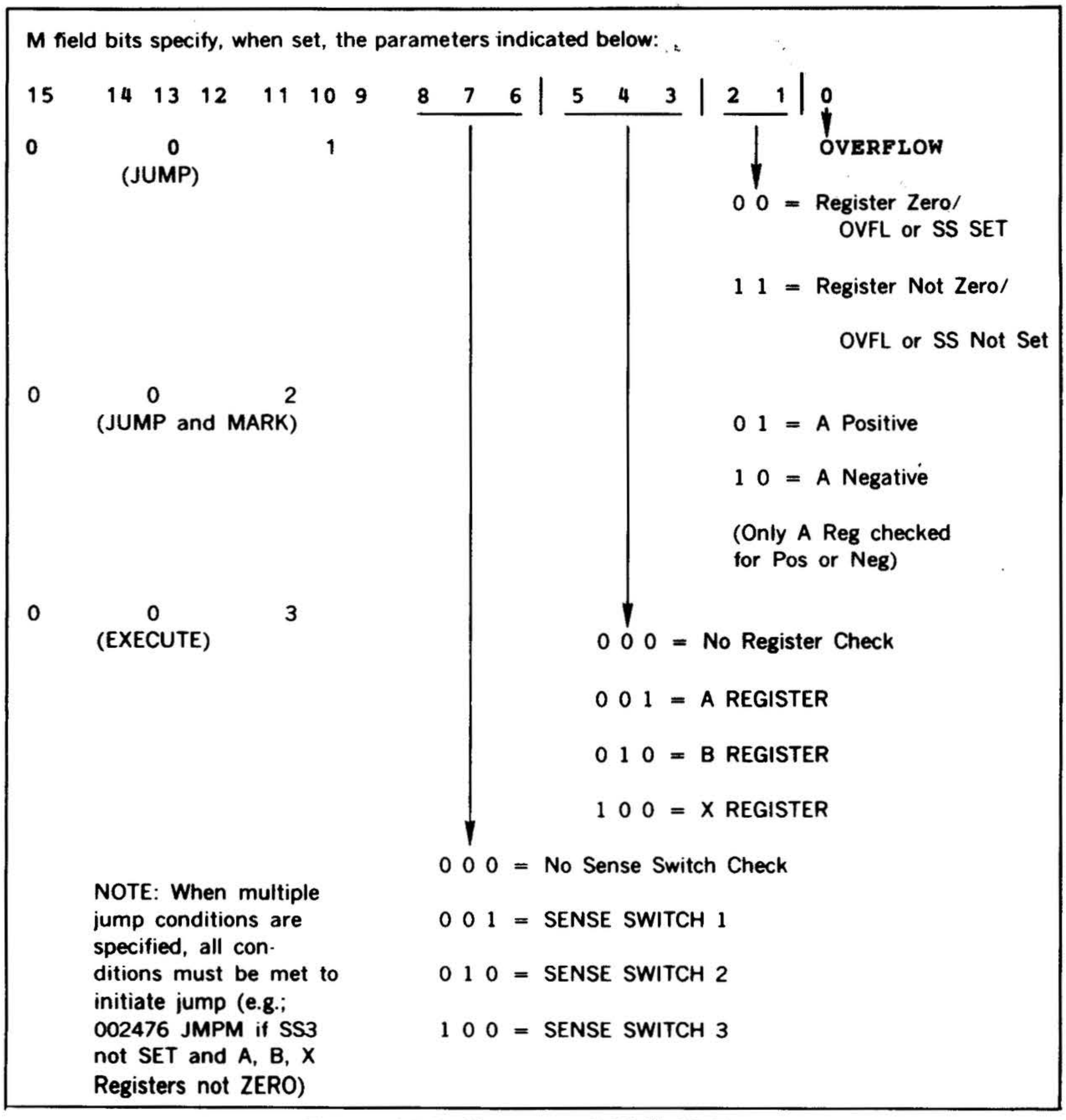


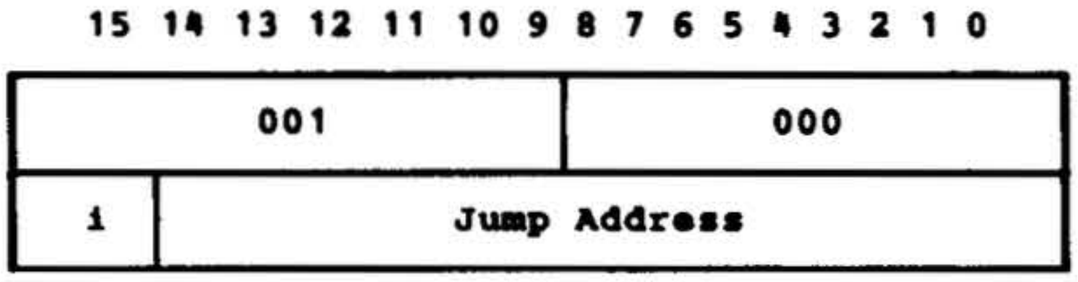
Figure 14-2. Meaning of M Field Bits

Refer to the discussion of assembler instruction types in section 15 for details of microcoding multiple instruction conditions.

In these instructions, the effective jump address in the second word is the address of the next instruction to be executed if the jump condition is met. The program then continues to execute instructions following the jump address.

If the jump condition is not met, the program executes the instruction immediately following the second word of the jump instruction.

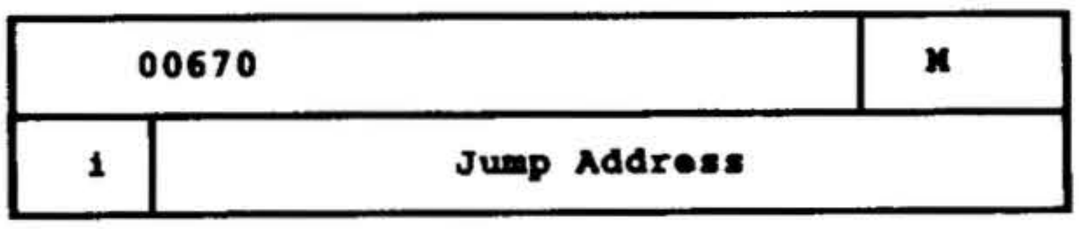
JMP Jump Unconditionally



Jumps unconditionally to the instruction at the effective jump address and executes it next.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

IJMP Indexed Jump



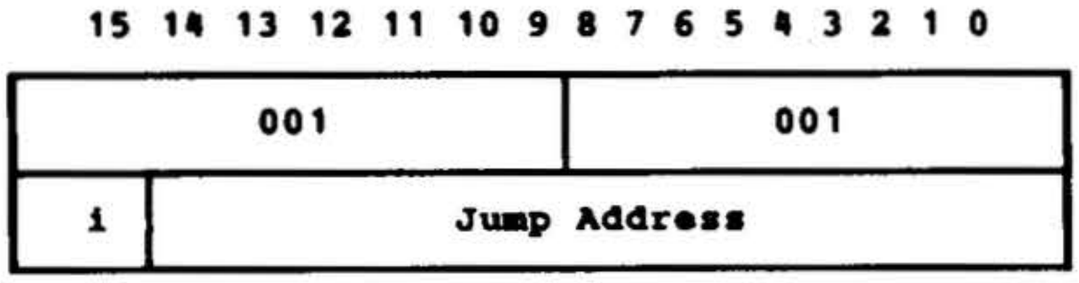
Jumps unconditionally to the instruction at the effective postindexed jump address and executes it next.

The effective postindexed jump address is formed by indexing the effective jump address given in the second word with the contents of the register specified by bits 0-2.

If bits 0-2 = 101, the indexing register is the X register; if bits 0-2 = 110, the indexing register is the B register.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	Postindexing
Register altered:	P

JOF Jump if Overflow Indicator Set

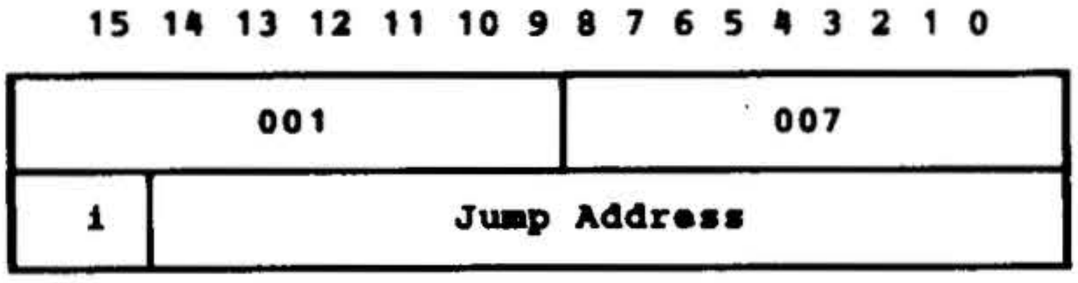


If the overflow indicator is set, jumps to the instruction at the effective jump address and executes it next. Resets the overflow indicator.

If the overflow indicator is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and OF

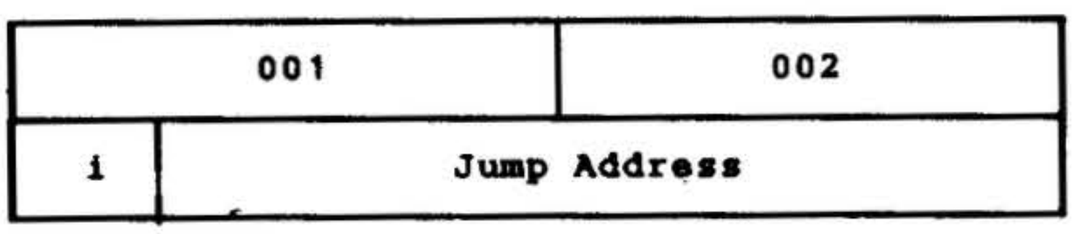
JOFN Jump if Overflow Indicator Not Set



If the overflow indicator is not set, jumps to the instruction at the effective jump address and executes it next. If the overflow indicator is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers Altered:	P

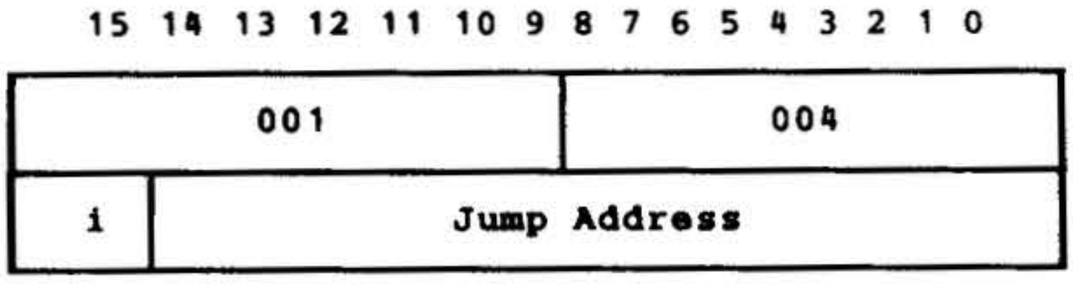
JAP Jump if A Register Positive



If the A register contains a positive value (including zero), jumps to the instruction at the effective jump address and executes it next. If the A register contains a negative value, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

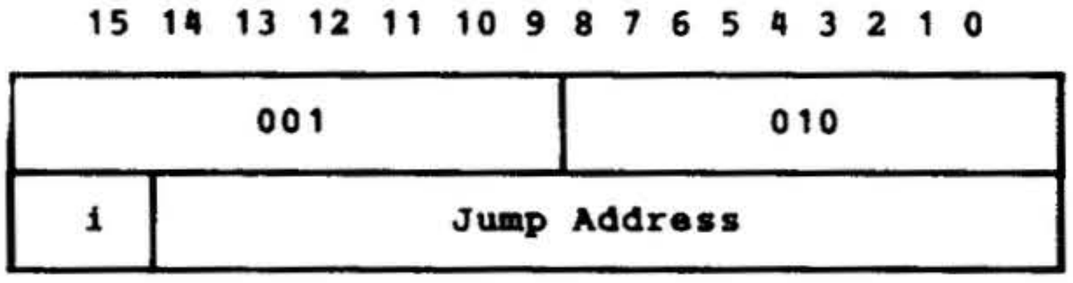
JAN Jump if A Register Negative



If the A register contains a negative value, jumps to the instruction at the effective jump address and executes it next. If the A register contains a positive value (including zero), executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

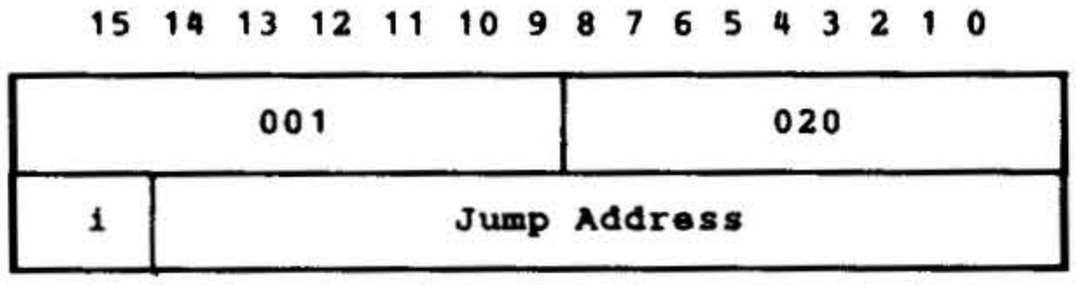
JAZ Jump if A Register Zero



If the A register contains zero, jumps to the instruction at the effective jump address and executes it next. If the A register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

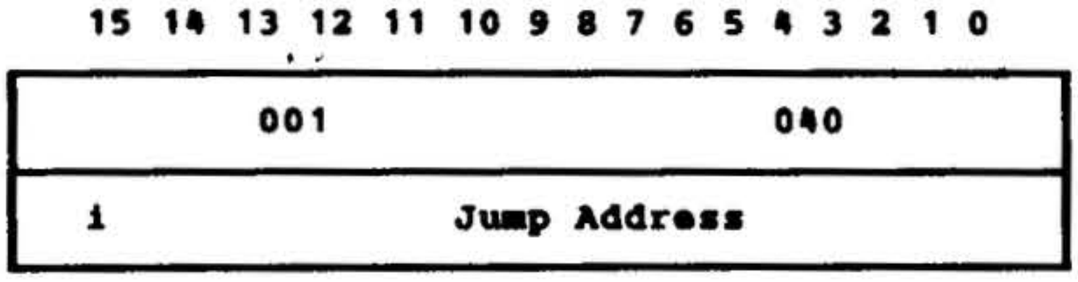
JBZ Jump if B Register Zero



If the B register contains zero, jumps to the instruction at the effective jump address and executes it next. If the B register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

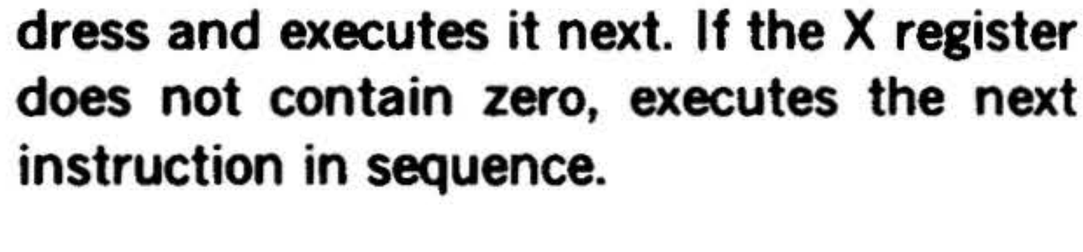
JXZ Jump if X Register Zero



If the X register contains zero, jumps to the instruction at the effective jump address and executes it next. If the X register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

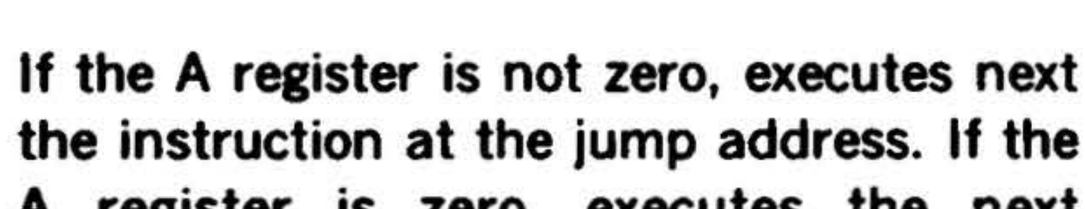
JANZ Jump if A Register Not Zero



If the A register is not zero, executes next the instruction at the jump address. If the A register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

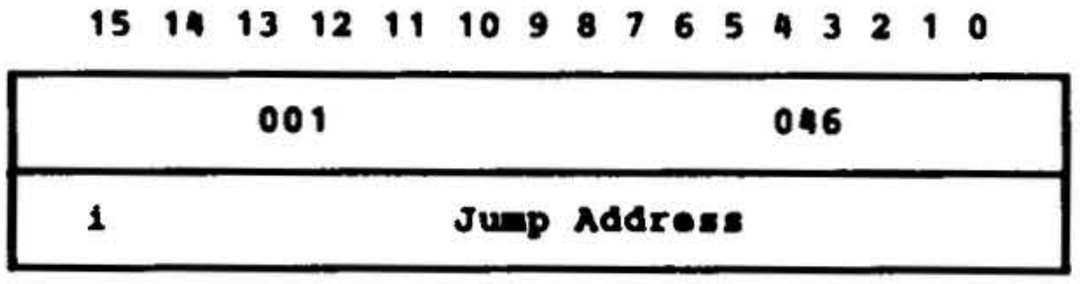
JBNZ Jump if B Register Not Zero



If the B register is not zero, executes next the instruction at the jump address. If the B register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

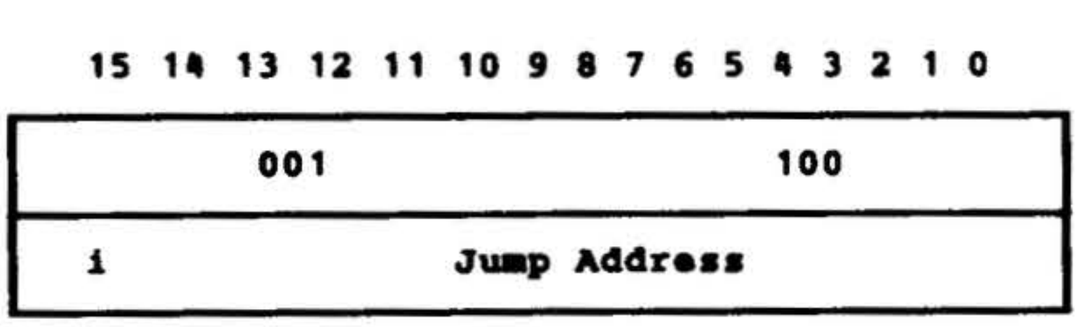
JXNZ Jump if X Register Not Zero



If the X register is not zero, executes next the instruction at the jump address. If the X register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

JSS1 Jump if SENSE Switch 1 Set



If SENSE switch 1 is set, jumps to the instruction at the effective jump address and executes it next. if SENSE switch 1 is not set, executes the next instruction in sequence.

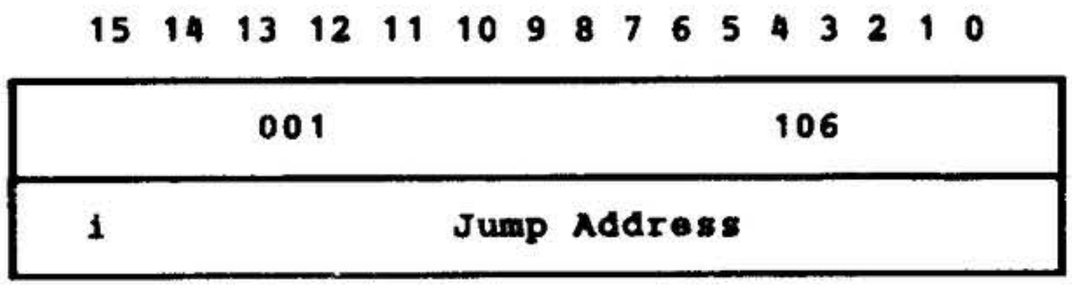
Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

JSS2 Jump if Sense Switch 2 Set

If SENSE switch 2 is set, jumps to the instruction at the effective jump address and executes it next. If SENSE switch 2 is not set, executes the next instruction in sequence.

Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

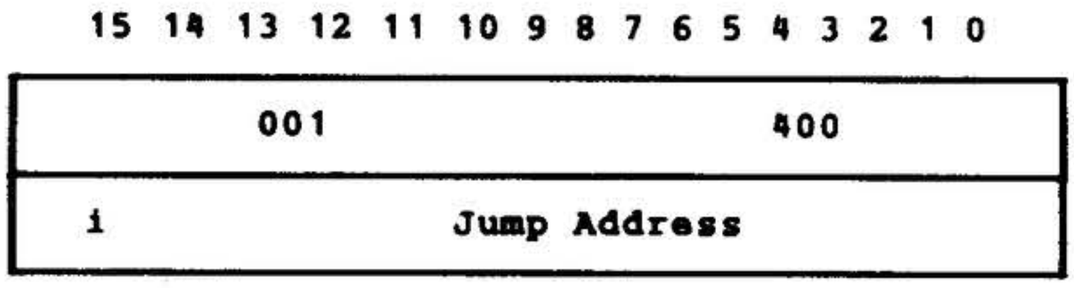
JS1N Jump if SENSE Switch 1 Not Set



If SENSE switch 1 is not set, executes next the instruction at the jump address. If SENSE switch 1 is set, executes the next instruction in sequence.

Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

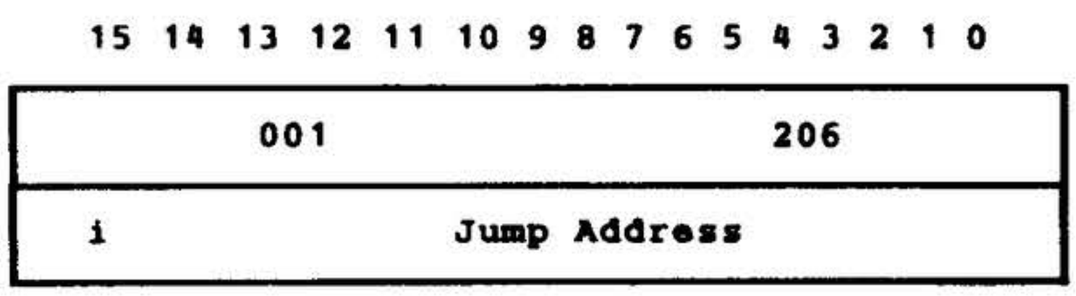
JSS3 Jump if SENSE Switch 3 Set



If SENSE switch 3 is set, jumps to the instruction at the effective jump address and executes it next. If SENSE switch 3 is not set, executes the next instruction in sequence.

Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

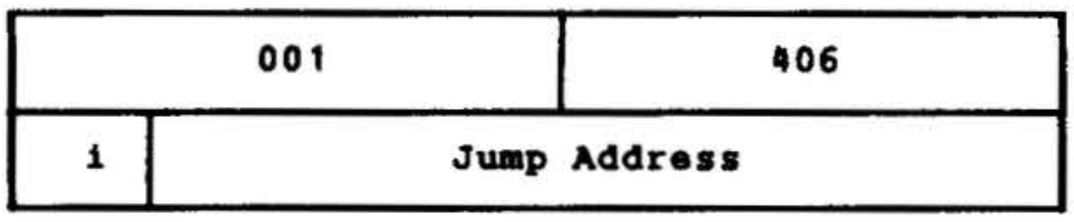
JS2N Jump if SENSE Switch 2 Not Set



If SENSE switch 2 is not set, executes next the instruction at the jump address. If SENSE switch 2 is set, executes the next instruction in sequence.

Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

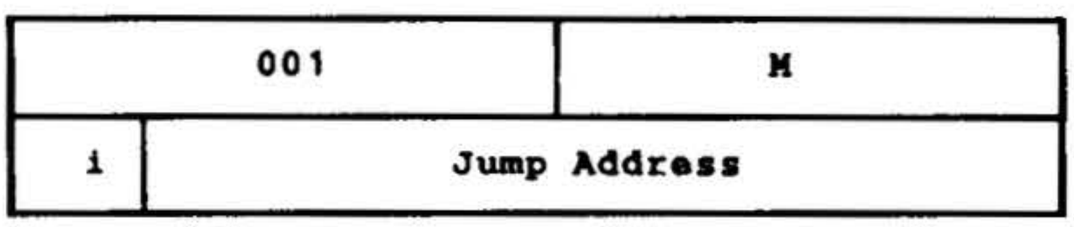
JS3N **Jump if SENSE Switch 3 Not Set**



If SENSE switch 3 is not set, executes next the instruction at the jump address. If SENSE switch 3 is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

JIF **Jump if Condition(s) Met**



If all the conditions specified by bits 0-8 are met, jumps to the instruction at the effective jump address and executes it next. The condition specified by setting combinations of M field bits is the AND of each bit specification as given in the preceding nine instructions (excluding JMP). JIF is used to microcode such combined conditions. A summary of the jump conditions is illustrated in figure 14-3.

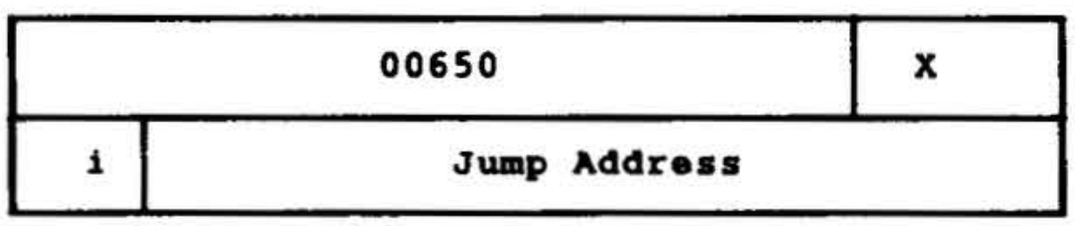
Compound conditions are specified in the first expression of the DAS assembler variable field (section 17).

Note that some combinations are impossible, e.g., jump if overflow indicator is not set and the A register positive (because of the conflicting use of bit 1).

If not all the jump conditions are met, JIF executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	Those specified

JSR **Jump and Set Return In Indexing Register**



Stores the contents of the P register in the indexing register specified by bits 0-2, then jumps unconditionally to the instruction at

M Field			Code	Jump Condition
000	000	001	0001	Overflow indicator set
000	000	010	0002	A register contents ≥ 0
000	000	110	0006	Jump if specified condition Not met.
000	000	100	0004	A register contents < 0
000	001	000	0010	A register contents = 0
000	010	000	0020	B register contents = 0
000	100	000	0040	X register contents = 0
001	000	000	0100	SENSE switch 1 set
010	000	000	0200	SENSE switch 2 set
100	000	000	0400	SENSE switch 3 set

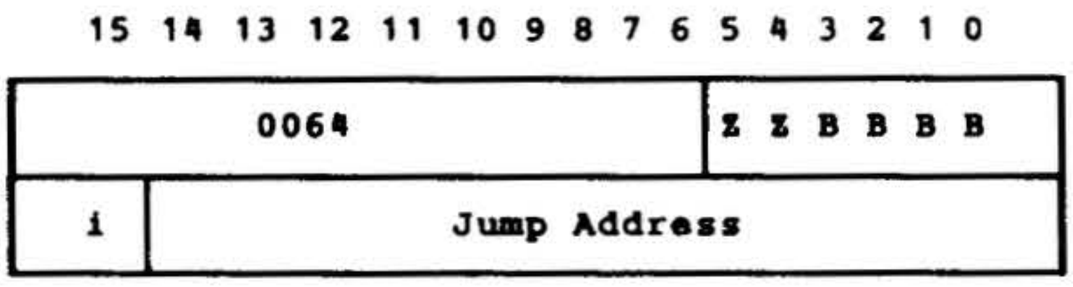
Figure 14-3. Summary of Jump Conditions

the effective jump address and executes it next.

If bits 0-2 = 101, the return register is the X register; if bits 0-2 = 110, the return register is the B register.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and B or X

BT **Bit Test**



Tests the condition of a selected bit in the A or B register, and jumps if the condition is met. Bits 0-3 select the bit to be tested.

Bits 4-5 define the condition to be met as follows:

If bits 5-4 =	BT jumps when:
00	The selected bit of the A register is 1
01	The selected bit of the B register is 1
10	The selected bit of the A register is 0
11	The selected bit of the B register is 0

If the specified condition is not met, the program executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P

Jump-and-Mark Instructions

This group comprises the instructions that direct the program to a nonsequential jump address, store the contents of the P register there, and next execute the instruction following the jump address.

Mnemonic	Instruction
JMPM	Jump and mark unconditionally
JOFM	Jump and mark if overflow indicator set
JOFNM	Jump and mark if overflow indicator not set
JAPM	Jump and mark if A register positive
JANM	Jump and mark if A register negative
JAZM	Jump and mark if A register zero
JBZM	Jump and mark if B register zero
JXZM	Jump and mark if X register zero
JANZM	Jump and mark if A register not zero
JBNZM	Jump and mark if B register not zero
JXNZM	Jump and mark if X register not zero
JS1M	Jump and mark if SENSE switch 1 set
JS2M	Jump and mark if SENSE switch 2 set
JS3M	Jump and mark if SENSE switch 3 set
JS1NM	Jump and mark if SENSE switch 1 not set
JS2NM	Jump and mark if SENSE switch 2 not set
JS3NM	Jump and mark if SENSE switch 3 not set
JIFM	Jump and mark if condition(s) met

These instructions have the same two-word addressing format as the jump instructions.

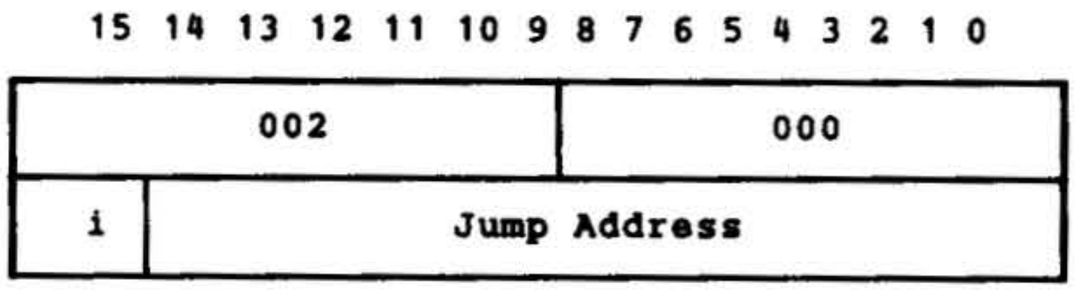
The operation code in bits 9-15 of these instructions has the configuration 0 000 010.

The M field (bits 0-8) has the same significance as the equivalent jump instruction.

The effective jump address in the second word is the address where the contents of the P register are stored if the jump-and-mark condition is met. The instruction next to be executed is located in the jump address plus one. The program then continues to execute instructions following the one in the jump address plus one.

If the jump-and-mark condition is not met, the program executes the instruction following the jump-and-mark instruction.

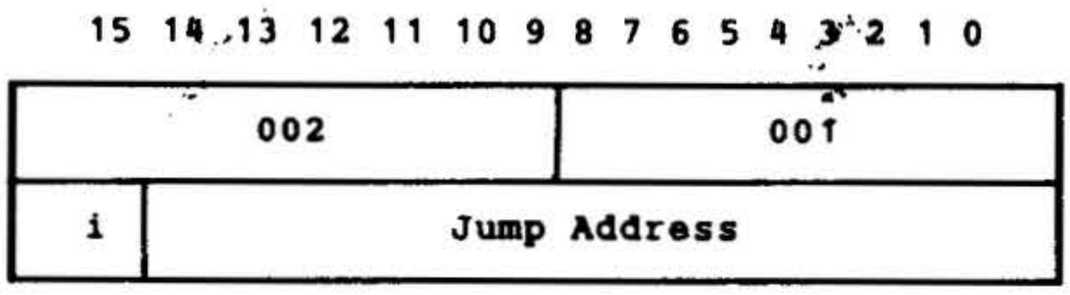
JMPM Jump and Mark Unconditionally



Jumps unconditionally to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

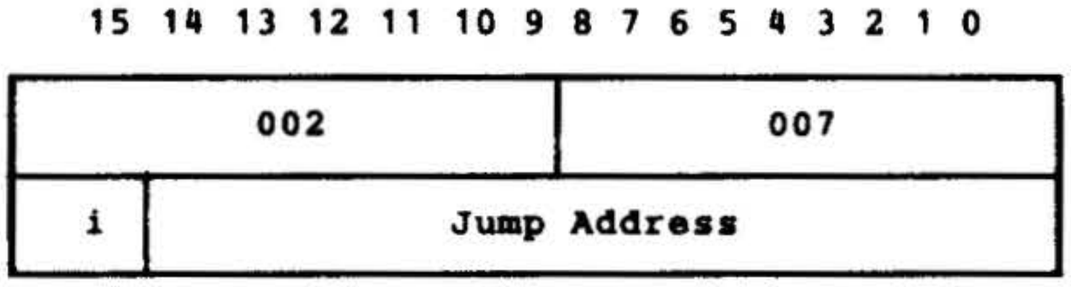
JOFM Jump and Mark if Overflow Indicator Set



If the overflow indicator is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. Resets the overflow indicator. If the overflow indicator is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P, OF, and effective jump address

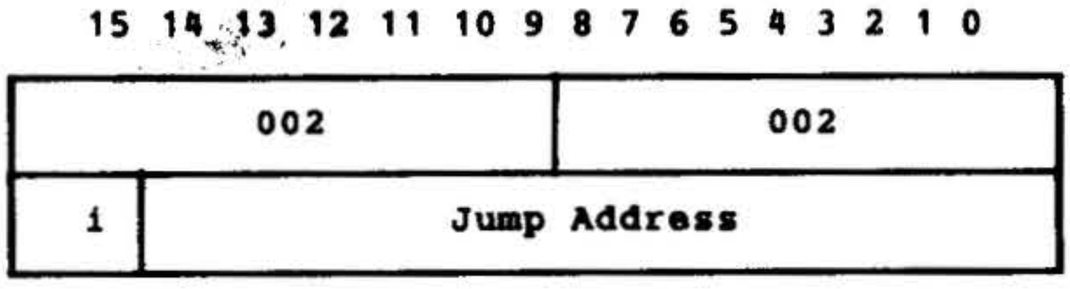
JOFNM Jump and Mark if Overflow Indicator Not Set



If the overflow indicator is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the overflow indicator is set, executes the next instruction in sequence. Does not reset OF.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

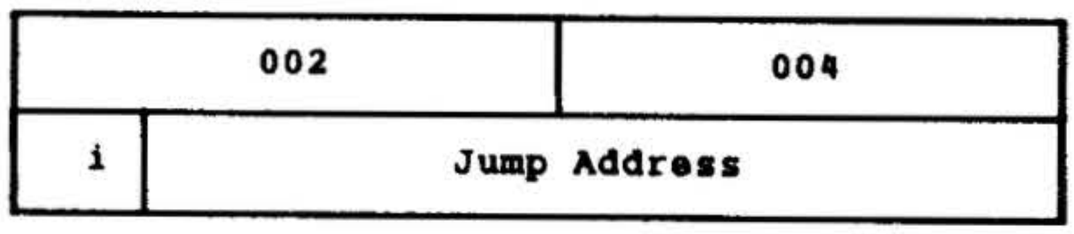
JAPM Jump and Mark if A Register Positive



If the A register contains a positive value (including zero), jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register contains a negative value, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

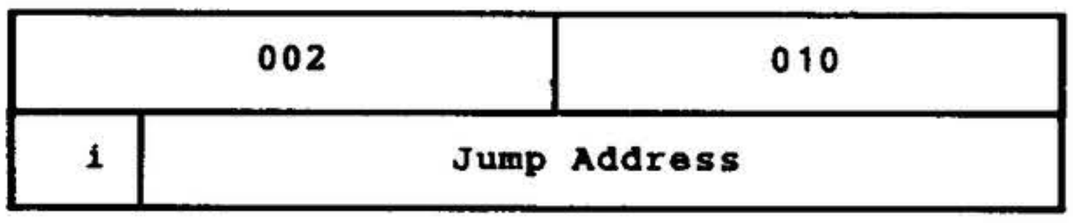
JANM Jump and Mark if A Register Negative



If the A register contains a negative value, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register contains a positive value (including zero), executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

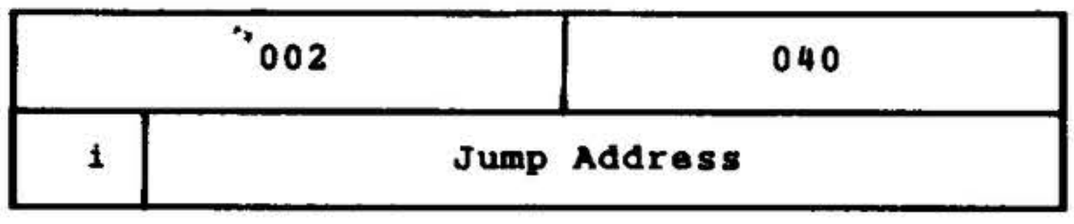
JAZM Jump and Mark if A Register Zero



If the A register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the A register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

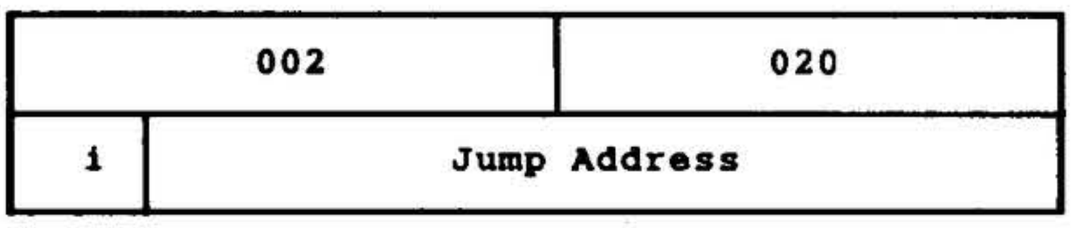
JXZM Jump and Mark if X Register Zero



If the X register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the X register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

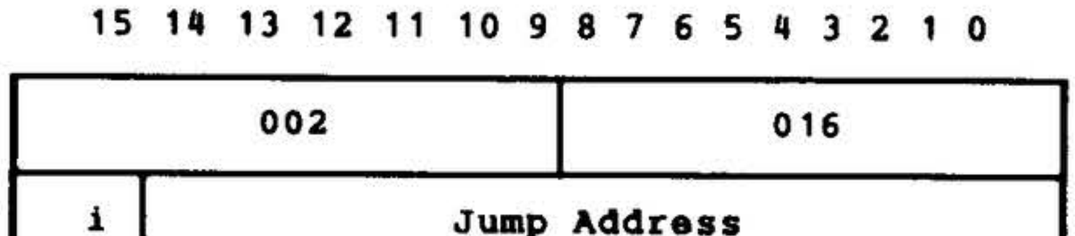
JBZM Jump and Mark if B Register Zero



If the B register contains zero, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If the B register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

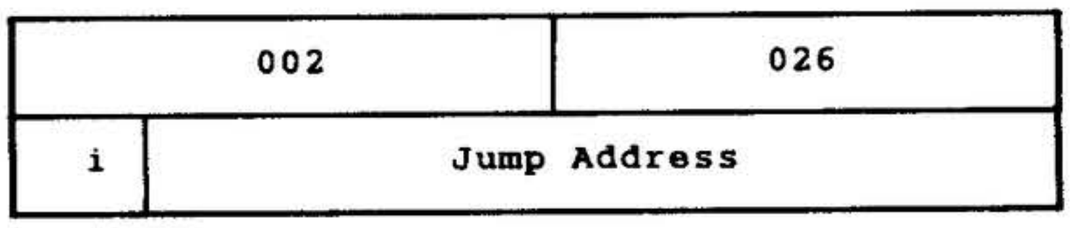
JANZM Jump and Mark if A Register Not Zero



If the A register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the A register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

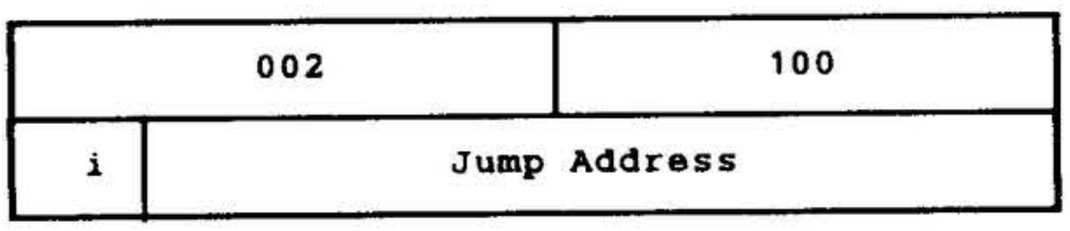
JBNZM Jump and Mark if B Register Not Zero



If the B register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the B register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

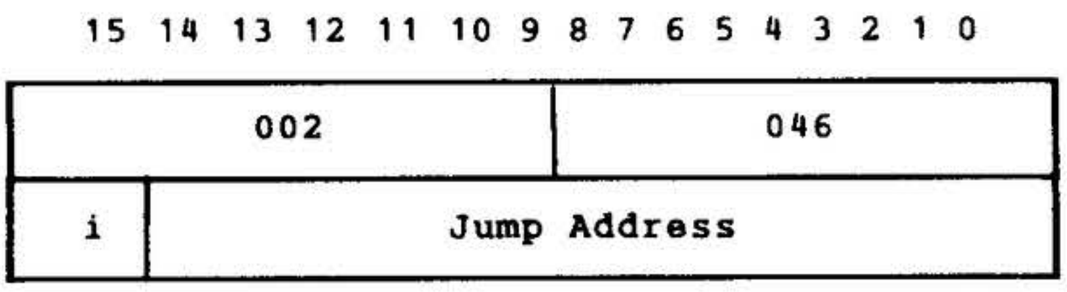
JS1M Jump and Mark if SENSE Switch 1 Set



If SENSE switch 1 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 1 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

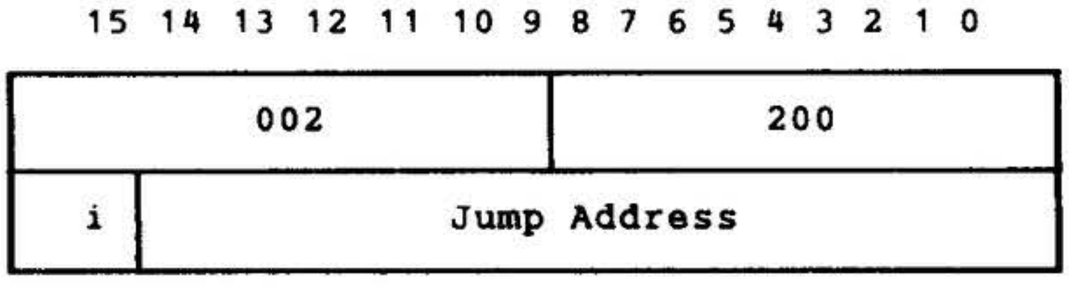
JXNZM Jump and Mark if X Register Not Zero



If the X register is not zero, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If the X register is zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

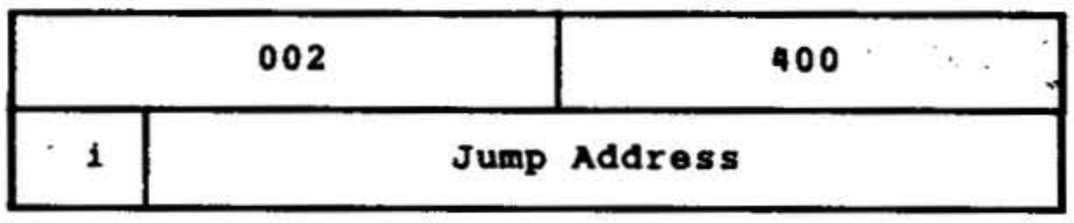
JS2M Jump and Mark if SENSE Switch 2 Set



If SENSE switch 2 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 2 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

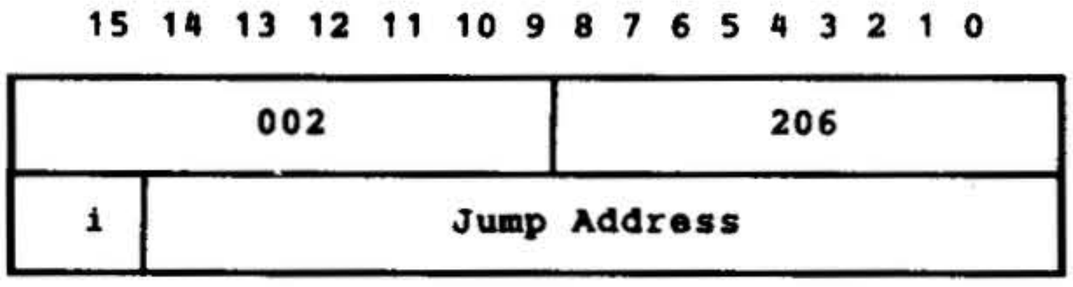
JS3M Jump and Mark if SENSE Switch 3 Set



If SENSE switch 3 is set, jumps to the effective jump address, stores the contents of the P register there, and next executes the instruction at the location following the effective jump address. If SENSE switch 3 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	P and effective jump address

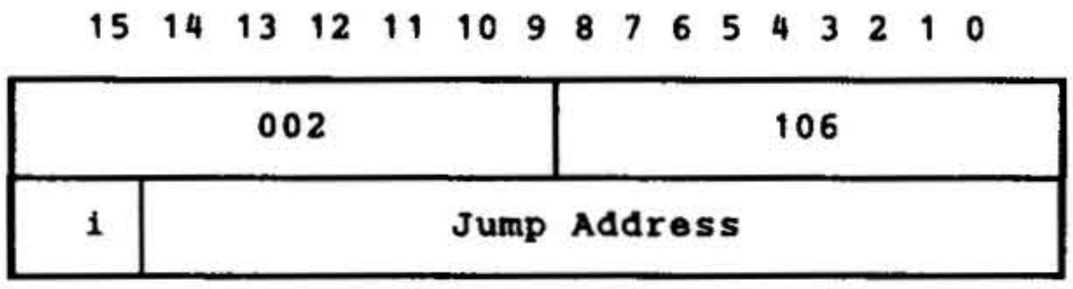
JS2NM Jump and Mark if Sense Switch 2 Not Set



If SENSE switch 2 is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If SENSE switch 2 is set, exeecutes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

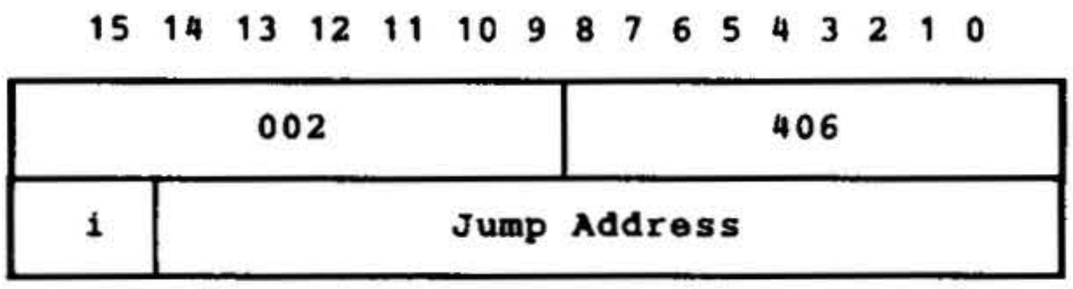
JS1NM Jump and Mark if Sense Switch 1 Not Set



If SENSE switch 1 is not set, stores the contents of the P register at the jump address, and executes the instruction at jump address plus one. If SENSE switch 1 is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

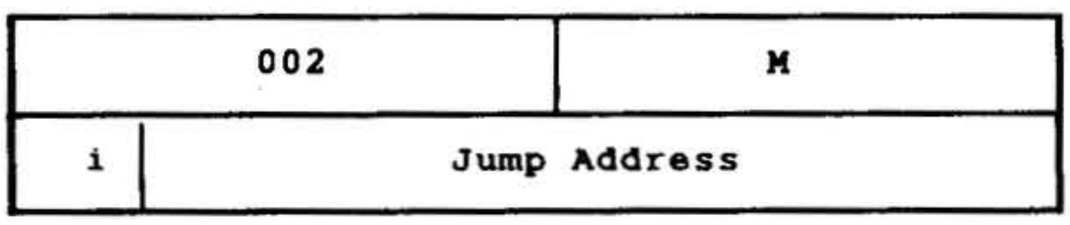
JS3NM Jump and Mark if Sense Switch 3 Not Set



If SENSE switch 3 is not set, stores the contents of the P register at the jump address, and executes the instruction at the jump address plus one. If SENSE switch is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	P and effective jump address

JIFM **Jump and Mark if Condition(s) Met**

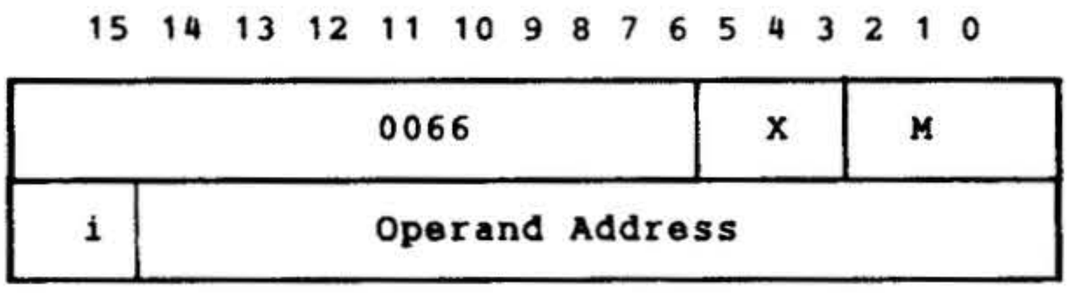


Analogous to JIF. If all jump conditions are met, jumps to the effective jump address, stores the contents of the P register there, and executes the instruction at the location following the effective jump address. If the jump conditions are not met, executes the next instruction in sequence.

The microcoded jump conditions are established and implemented as described for the JIF instruction in the previous subsection.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	Those specified

SRE **Skip if Register Equal**



Makes a logical comparison between the register specified by bits 3-5 and the word at the address specified by the second word of the instruction. Bits 3-5 = 001 specifies the A register, bits 3-5 = 010 specifies the B register, and bits 3-5 = 100 specifies the X register.

Bits 0-2 specify the addressing mode: 000 or 100 = relative to P, 001, or 101 = indexed with X, 010 or 110 = indexed with B, and 011 or 111 = direct/indirect (postindexing).

If the compared quantities are equal, the program skips the next two locations and executes the instruction in the third location. If the compared quantities are unequal, the program executes the instruction immediately following SRE.

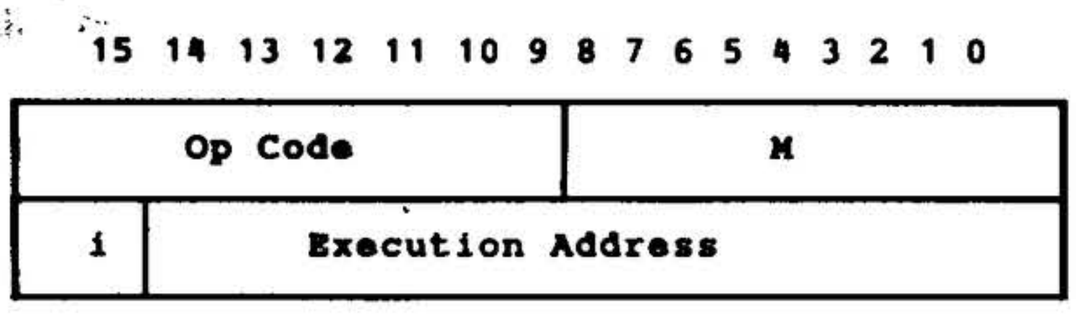
Relative addressing:	Yes
Indirect addressing:	Yes
Indexing:	Postindexing
Register altered:	None

Execution Instructions

This group comprises the instructions that direct the program to a nonsequential address for execution of the instruction located there, and then direct the program back to the main sequence to execute the instruction following the two-word execution instruction.

Mnemonic	Instruction
XEC	Execute unconditionally
XOF	Execute if overflow indicator set
XOFN	Execute if overflow indicator not set
XAP	Execute if A register positive
XAN	Execute if A register negative
XAZ	Execute if A register zero
XBZ	Execute if B register zero
XXZ	Execute if X register zero
XANZ	Execute if A register not zero
XBNZ	Execute if B register not zero
XXNZ	Execute if X register not zero
XS1	Execute if SENSE switch 1 set
XS2	Execute if SENSE switch 2 set
XS3	Execute if SENSE switch 3 set
XS1N	Execute if SENSE switch 1 not set
XS2N	Execute if SENSE switch 2 not set
XS3N	Execute if SENSE switch 3 not set
XIF	Execute if condition(s) met

These instructions have the following two-word addressing format.



This format comprises a seven-bit operation code (bits 9-15), a nine-bit M field (bits 0-8) in the first word, and an effective execution address in the second.

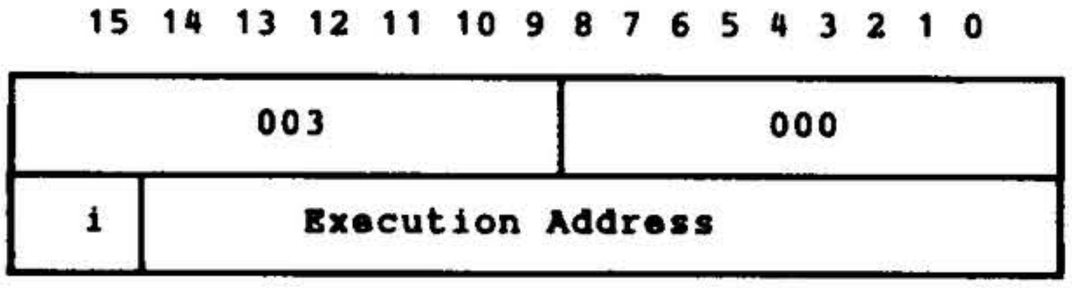
These instructions have configuration 0 000 011 in bits 9-15.

The M field bits have the same significance as the equivalent jump instruction.

The effective address in the second word is the address of the next instruction to be executed if the execution condition is met. After executing that instruction, the program returns to the main sequence and executes the next instruction in sequence.

Note that only one-word instructions that do not specify relative addressing can be contained in the execution address.

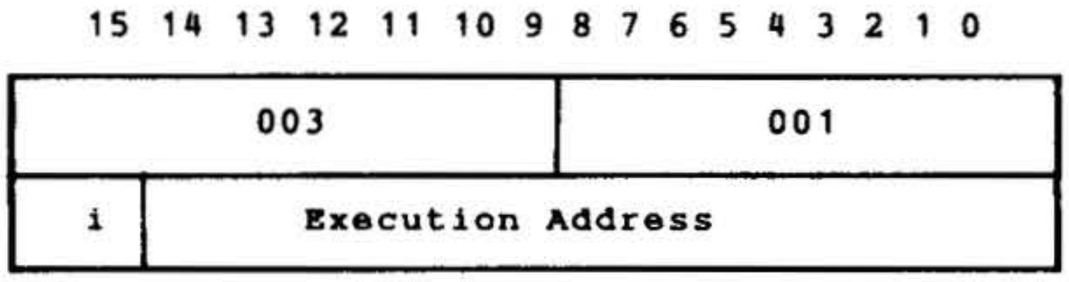
XEC Execute Unconditionally



Executes the instruction at the effective execution address in the second word, and then returns to execute the instruction following XEC.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

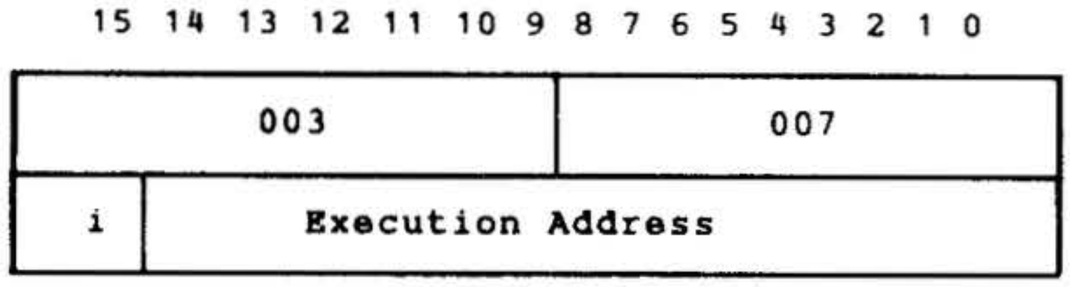
XOF Execute if Overflow Indicator Set



If the overflow indicator is set, executes the instruction at the effective execution address, and then returns to execute the instruction following XOF. Resets the overflow indicator. If the overflow indicator is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	OF

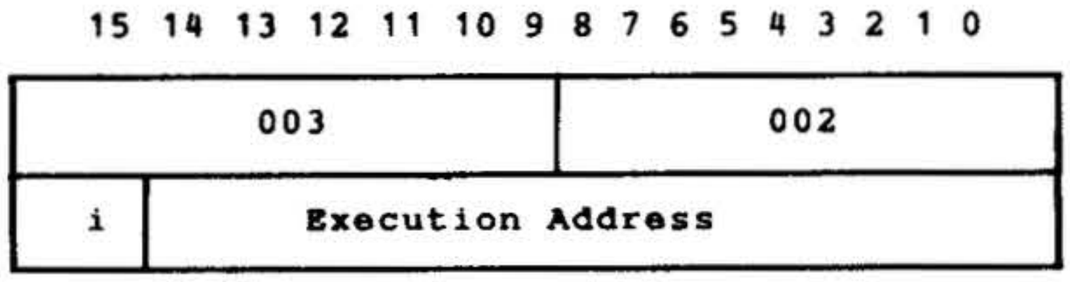
XOFN Execute if Overflow Indicator Not Set



If the overflow indicator is not set, executes the instruction at the effective execution address, and then returns to execute the instruction following XOFN. If the overflow indicator is set, executes the next instruction in sequence. Does not reset OF.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

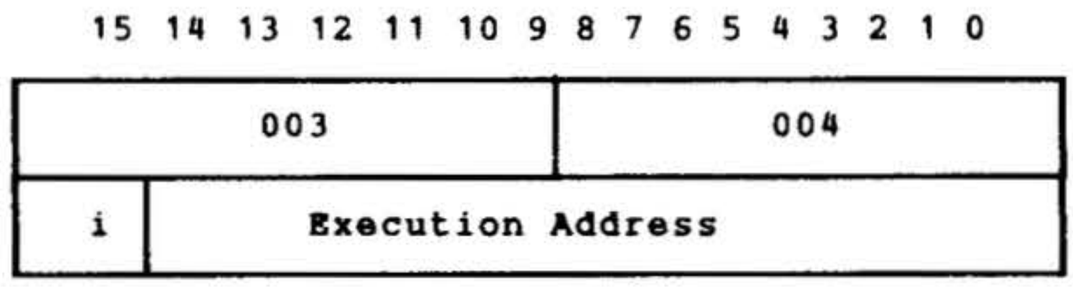
XAP Execute if A Register Positive



If the A register contains a positive value (including zero), executes the instruction at the effective execution address, and then returns to execute the instruction following XAP. If the A register contains a negative value, executes the next instruction in sequence.

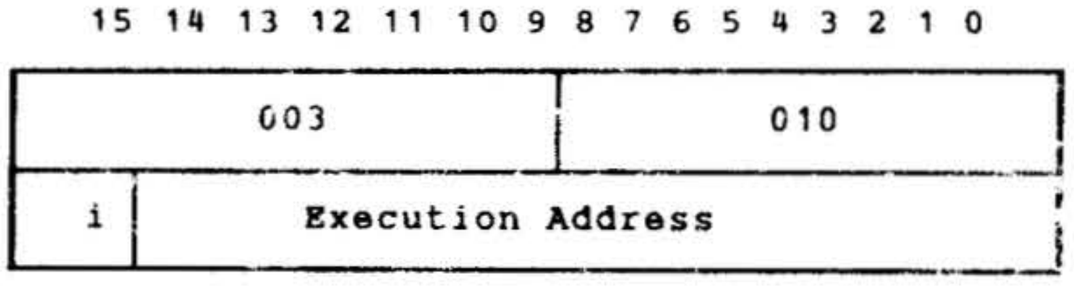
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

XAN Execute if A Register Negative



If the A register contains a negative value, executes the instruction at the effective execution address, and then returns to execute the instruction following XAN. If the A register contains a positive value (including zero), executes the next instruction in sequence.

XAZ Execute if A Register Zero

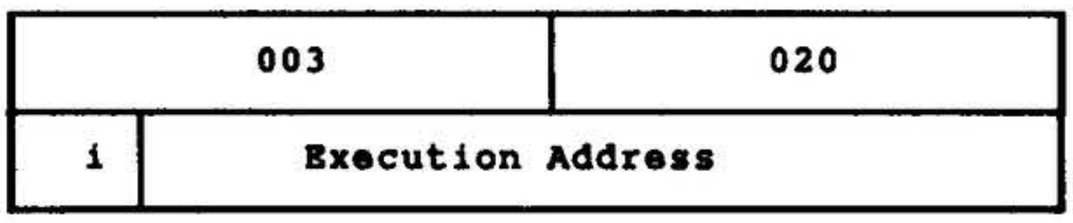


If the A register contains zero, executes the instruction at the effective execution

address, and then returns to execute the next instruction. If the A register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

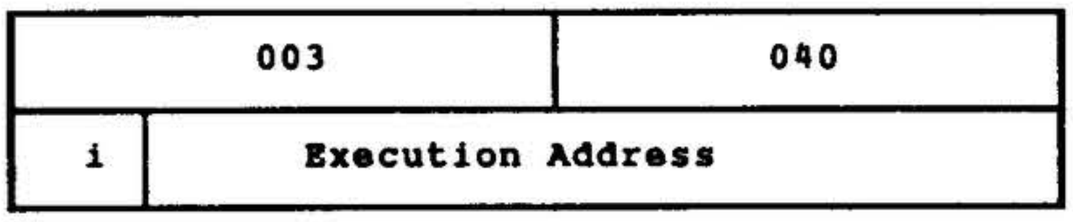
XBZ Execute if B Register Zero



If the B register contains zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the B register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

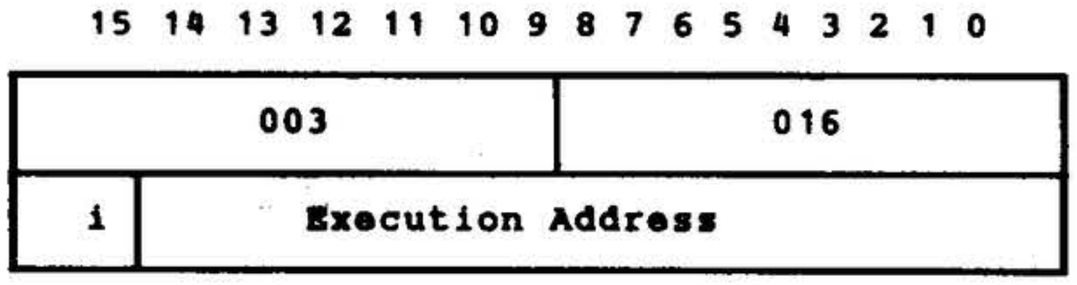
XXZ Execute if X Register Zero



If the X register contains zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the X register does not contain zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

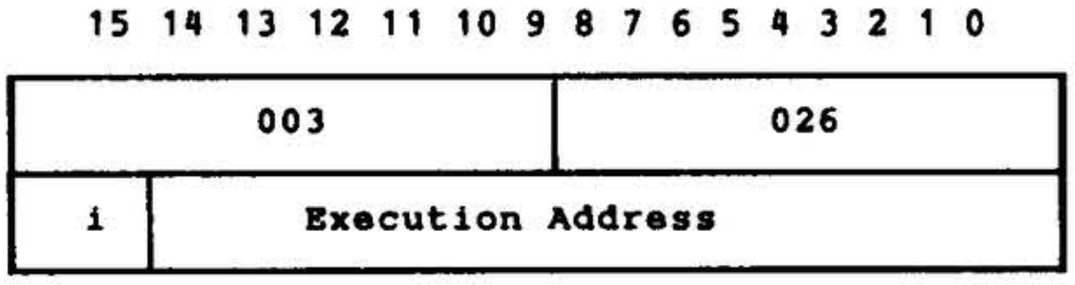
XANZ Execute if A Register Not Zero



If the A register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the A register contains zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

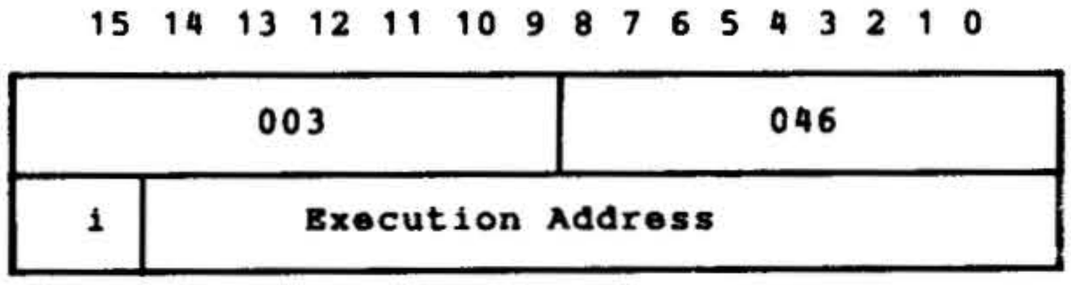
XBNZ Execute if B Register Not Set



If the B register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the B register contains zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

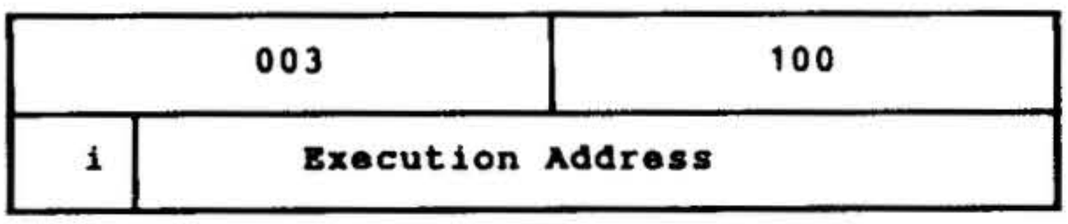
XXNZ Execute if X Register Not Zero



If the X register contents are not zero, executes the instruction at the effective execution address, and then returns to execute the next instruction. If the X register contains zero, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

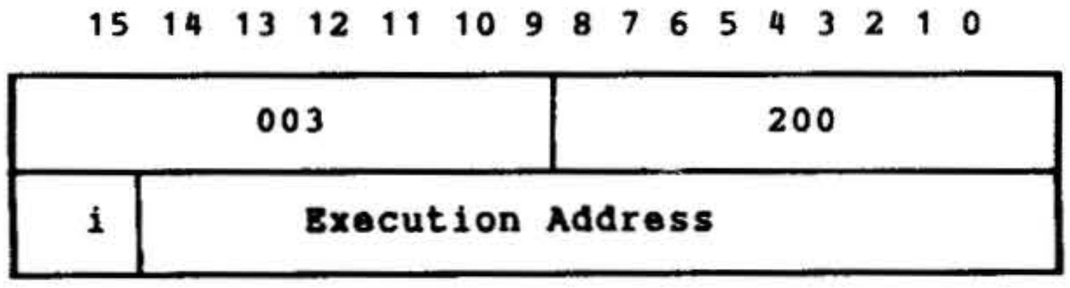
XS1 Execute if SENSE Switch 1 Set



If SENSE switch 1 is set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 1 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

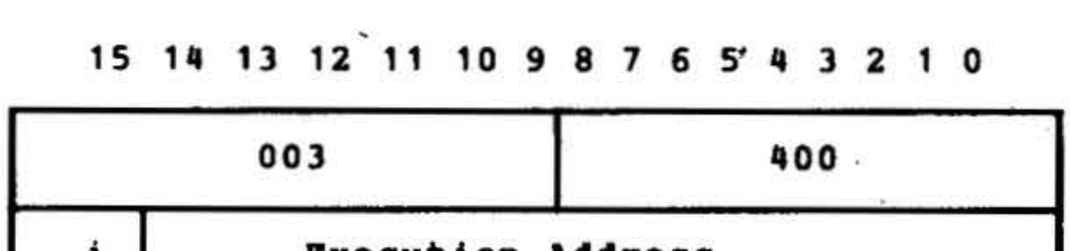
XS2 Execute if SENSE Switch 2 Set



If SENSE switch 2 is set, executes the instruction at the effective execution address; and then returns to execute the next instruction. If SENSE switch 2 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

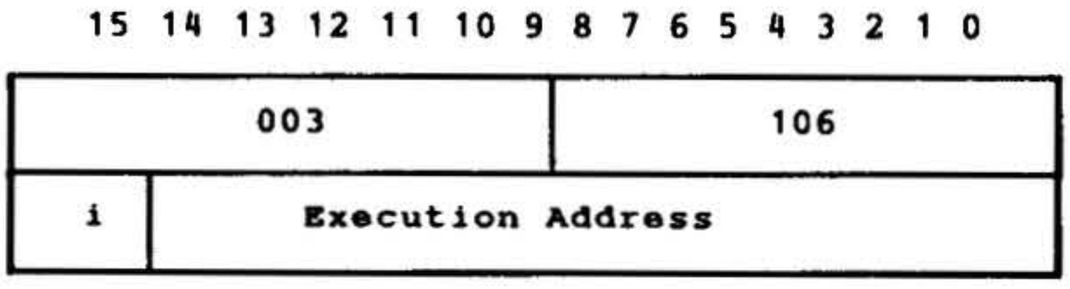
XS3 Execute if SENSE Switch 3 Set



If SENSE switch 3 is set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 3 is not set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

XS1N Execute if SENSE Switch 1 Not Set



If SENSE switch 1 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 1 is set, executes the next instruction in sequence.

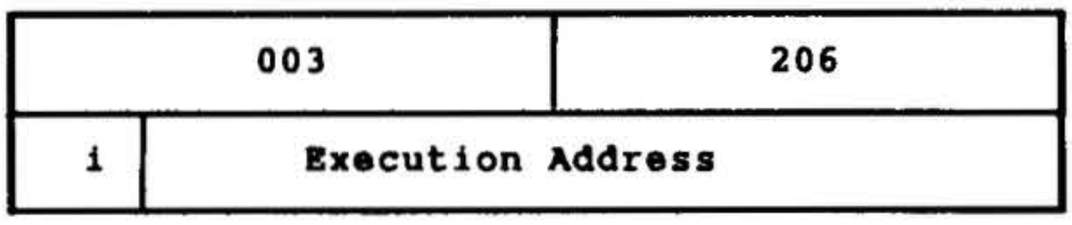
Relative addressing:	No
Indirect addressing:	Yes

Indexing: No
Register altered: None

dress, and then returns to execute the next instruction. If SENSE switch 3 is set, executes the next instruction in sequence.

XS2N Execute if SENSE Switch 2 Not Set

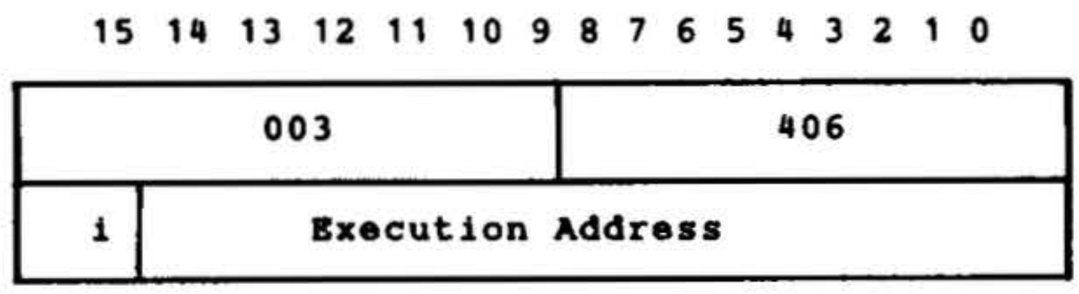
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None



If SENSE switch 2 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 2 is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

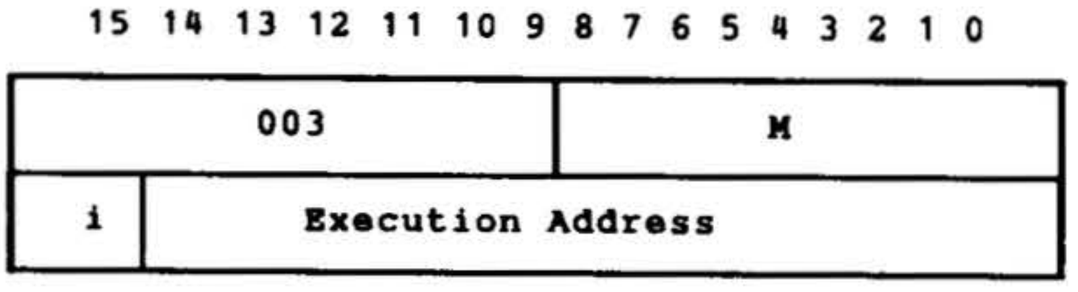
XS3N Execute if SENSE Switch 3 Not Set



If SENSE switch 3 is not set, executes the instruction at the effective execution address, and then returns to execute the next instruction. If SENSE switch 3 is set, executes the next instruction in sequence.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

XIF Execute if Condition(s) Met



Analogous to JIF. If all execution conditions are met, executes the instruction at the effective execution address, and then returns to execute the instruction following XIF. If all execution conditions are not met, executes the next instruction in sequence.

The microcoded execution conditions are established and implemented as described for the JIF instruction.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Register altered:	None

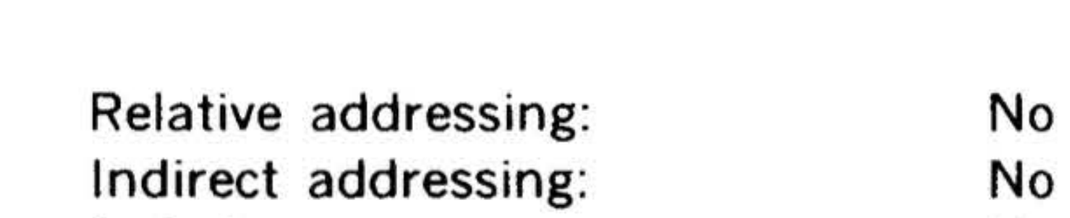
Control Instructions

This group comprises general control instructions.

Mnemonic	Instruction
HLT	Halt
NOP	No operation
ROF	Reset overflow indicator
SOF	Set overflow indicator

These instructions have one-word, nonaddressing formats.

HLT — Halt

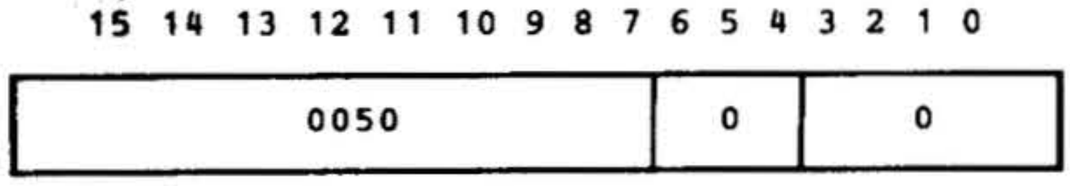


Stops computation and places the computer in step mode. To restart computation with the next instruction in sequence, press START on the control panel.

Bits 0-8 can contain any value, e.g., a value assigned to a specific halt can be displayed on the control panel in the instruction register after the halt occurs. This display will identify the halt condition.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

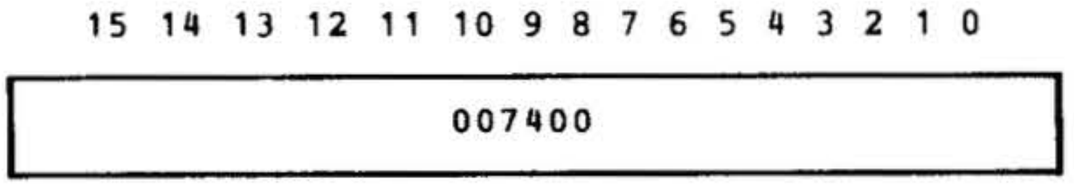
NOP — No Operation



Waits one cycle. The P register is incremented by one, but the A, B, and X registers and memory are not affected.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

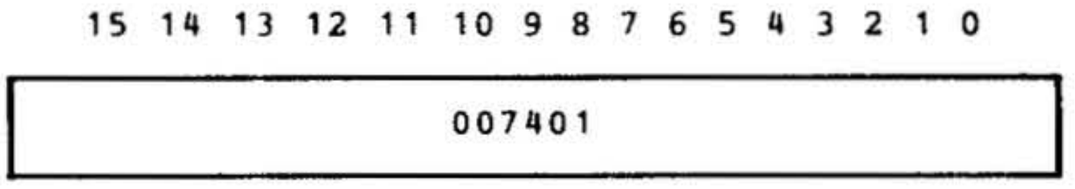
ROF — Reset Overflow Indicator



Resets the overflow indicator (OF).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF

SOF — Set Overflow Indicator



Sets the overflow indicator.

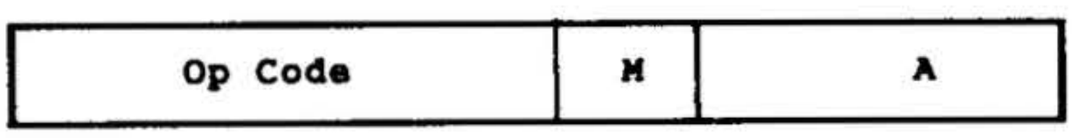
Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	OF

I/O Instructions

This group comprises the instructions for implementing communication between the computer and the peripheral devices that supply and receive data.

Mnemonic	Instruction
EXC	External control
EXC2	Auxiliary external control
SEN	Program sense
CIA	Clear and input to A register
CIB	Clear and input to B register
CIAB	Clear and input to A and B registers
INA	Input to A register
INB	Input to B register
INAB	Input to A and B registers
OAR	Output from A register
OBR	Output from B register
OAB	Output from A and B registers
IME	Input to memory
OME	Output from memory

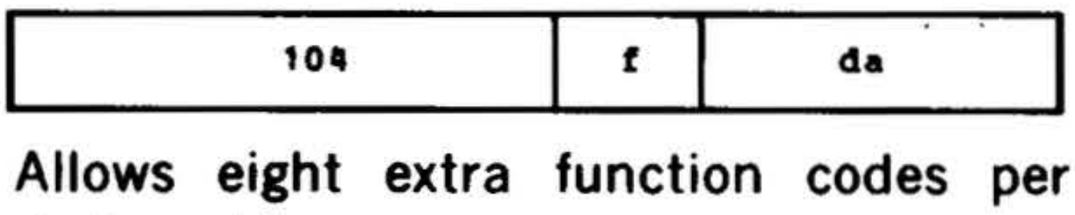
The format of one-word I/O instructions is:



where the M field designates the register, line, or function involved, and the A field, a peripheral device address (da).

The formats of two-word I/O instructions are identical with those of analogous operation instructions previously described.

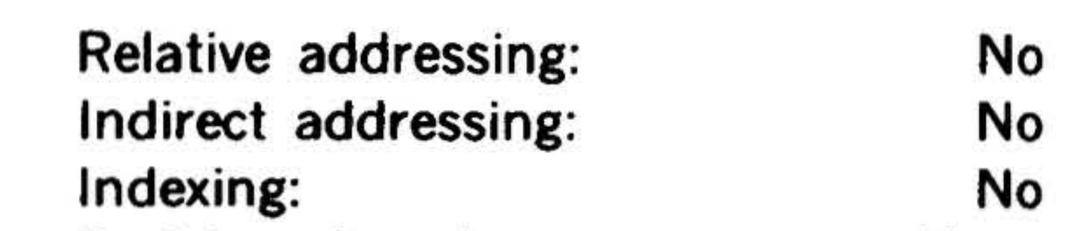
EXC External Control



Places on the E bus a nine-bit order for the peripheral device to perform function f.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: None

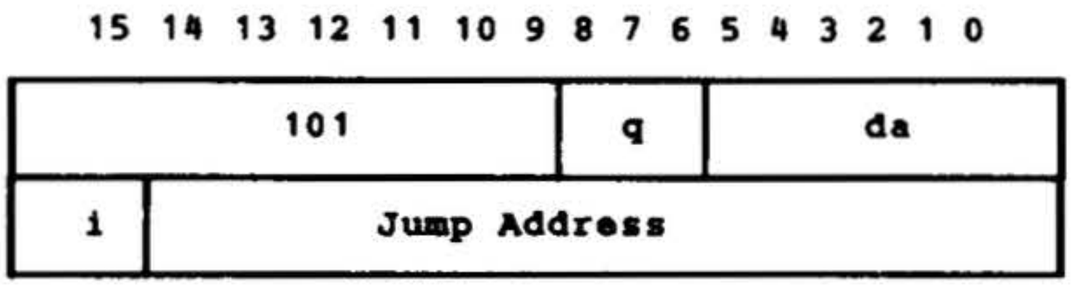
EXC2 Auxiliary External Control



Allows eight extra function codes per device address.

Relative addressing: No
Indirect addressing: No
Indexing: No
Register altered: None

SEN Program Sense

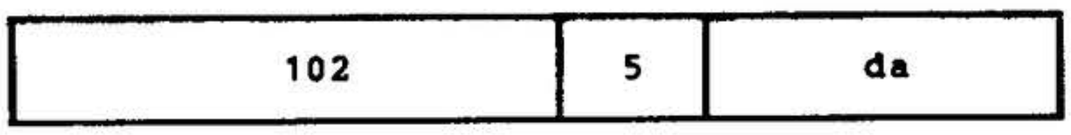


Places on the E bus a nine-bit order to sense the status of line q in the peripheral device.

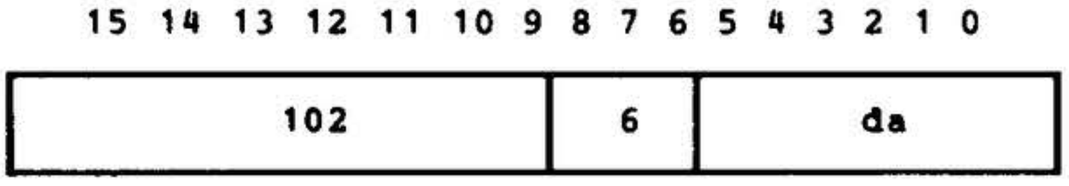
If the computer receives a true response signal, jumps to the instruction at the effective jump address and executes it next. If the response signal is false, executes the next instruction in sequence.

Relative addressing: No
Indirect addressing: Yes
Indexing: No
Register altered: P

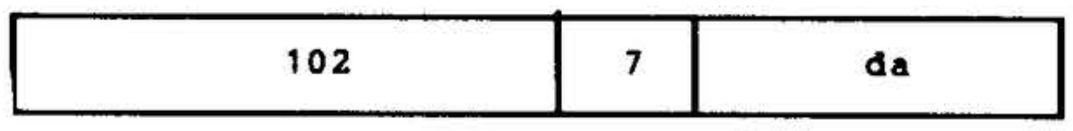
CIA Clear and Input to A Register



CIB Clear and Input to B Register



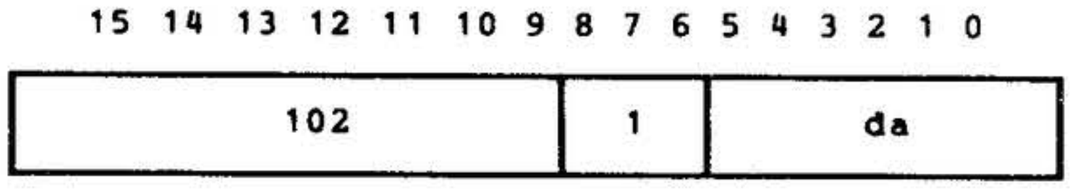
CIAB Clear and Input to A and B Registers



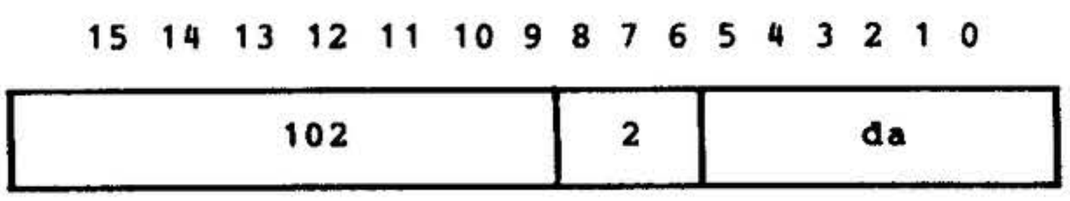
Clears the specified register(s) and inputs a data word from the peripheral device to the specified register(s).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Only those specified

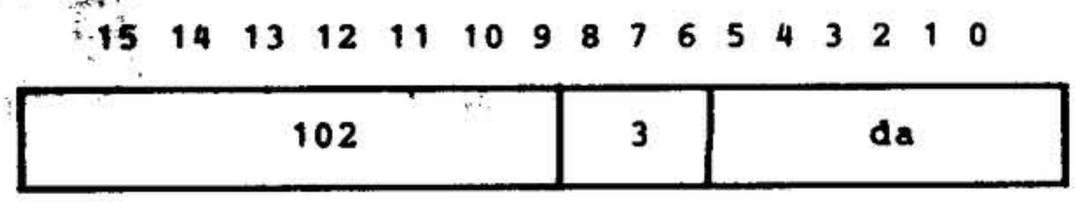
INA Input to A Register



INB Input to B Register



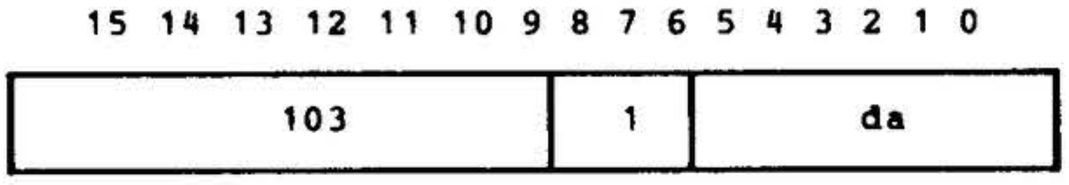
IAB Input to A and B Registers



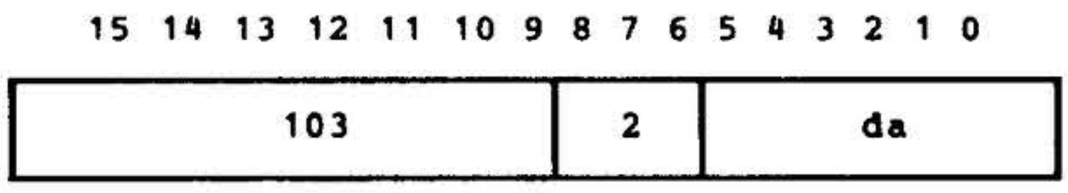
Inclusively ORs a data word from the peripheral device with the contents of the specified register(s), and places the result in the specified register(s).

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Registers altered:	Only those specified

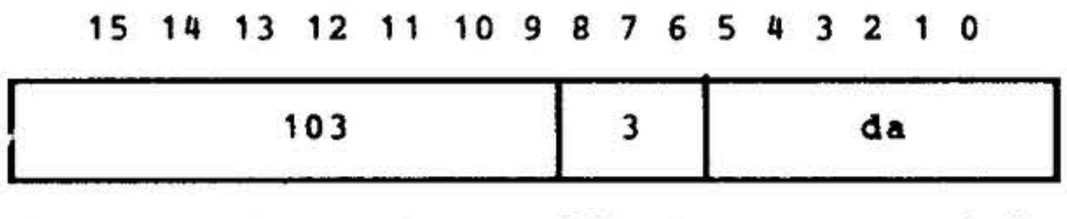
OAR Output From A Register



OBR Output From B Register



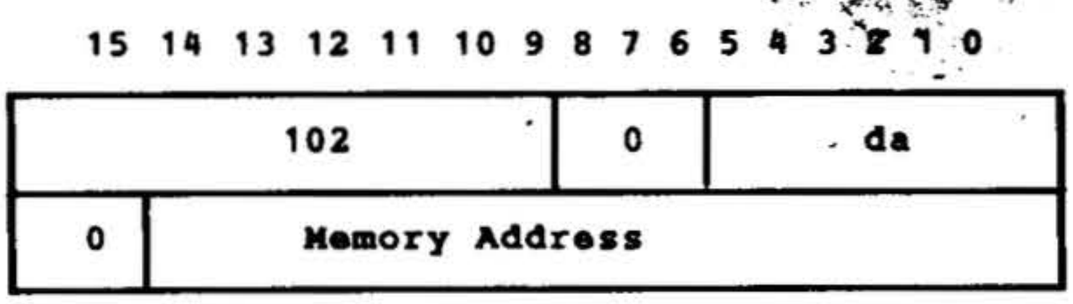
OAB Output Inclusive-OR of A and B Registers



Outputs the inclusive-ORed contents of the specified register(s) to the peripheral device.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

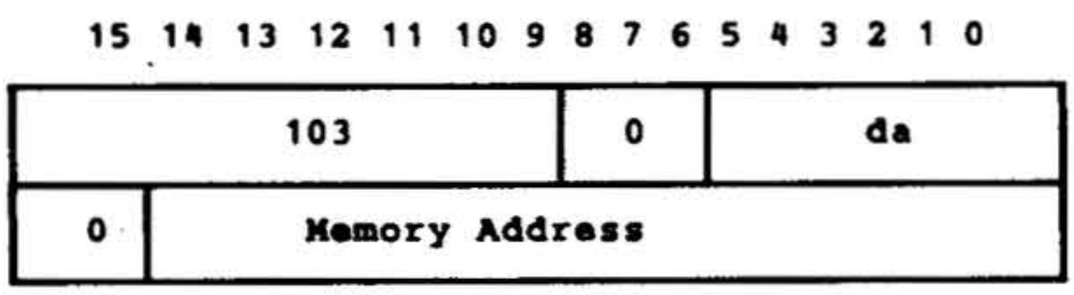
IME — Input [illegible] Memory



Inputs a data word from the peripheral device to the cleared memory address in the second word.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	Memory

OME — Output From Memory



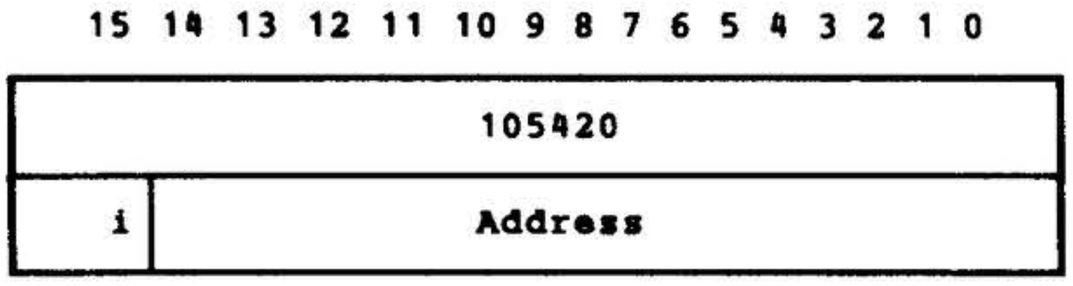
Outputs the contents of the memory address in the second word to the peripheral device.

Relative addressing:	No
Indirect addressing:	No
Indexing:	No
Register altered:	None

Floating Point Processor

This group comprises the instructions that are used with the floating point processor option. Included are instructions to load the floating point accumulator, store floating point accumulator in memory, floating point add, subtract, multiply, and divide.

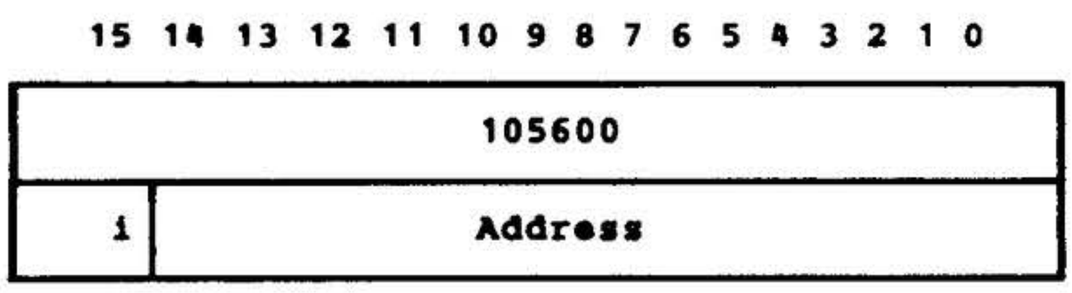
FLD Single Precision Load



Loads the single precision number at the effective memory address into the floating point accumulator.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

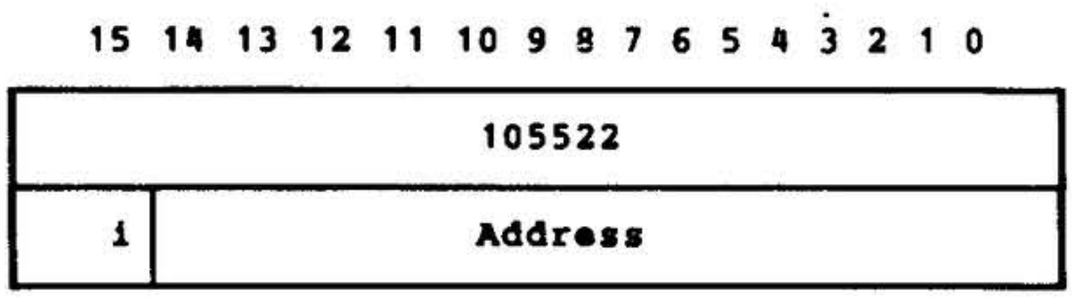
FST Single Precision Store



The floating point accumulator is rounded and stored at the effective memory address in single precision format.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

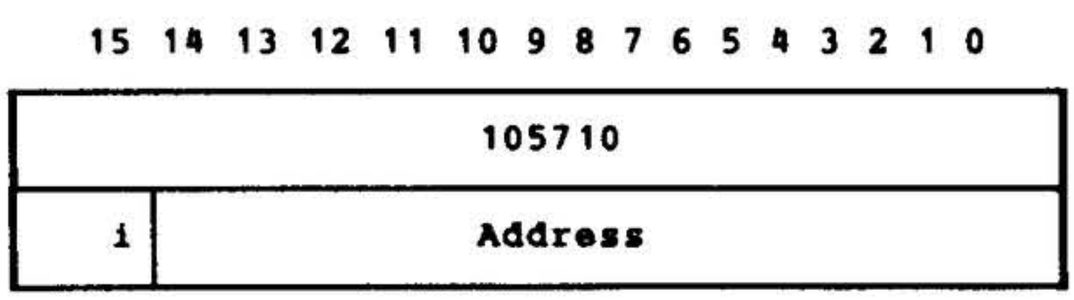
FLDD Double Precision Load



Loads the double precision number at the effective memory address into the floating point accumulator.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

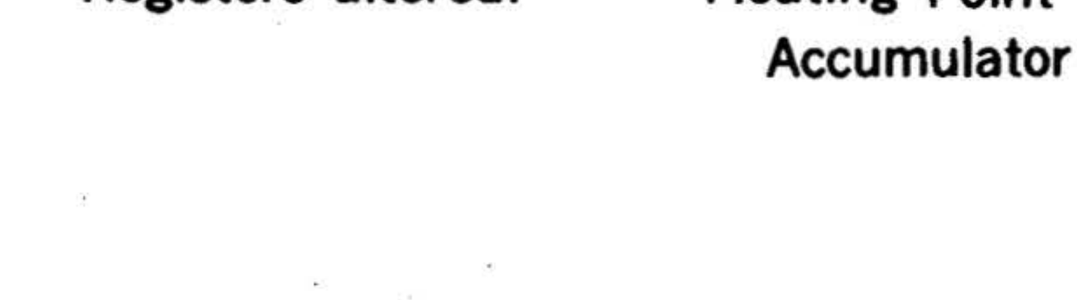
FSTD Double Precision Store



The floating point accumulator is rounded and stored at the effective memory address in double precision format.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

FLT Reformat to Floating Point



The single precision integer number (16 bits) at the effective memory address is reformated to floating point notation and loaded into the floating point accumulator.

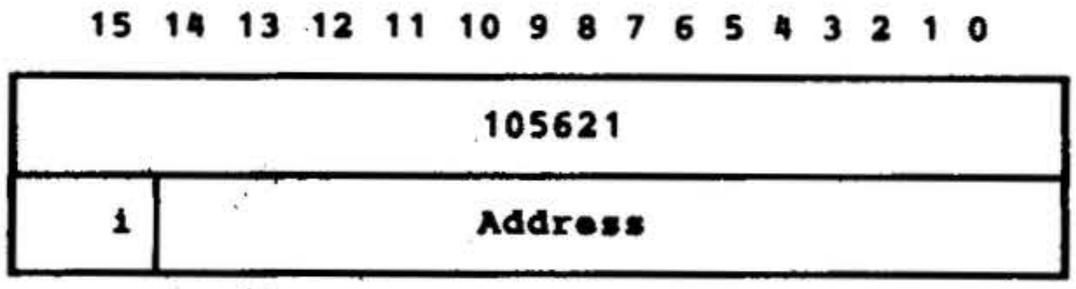
Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

FAD Floating Add Single Precision

Adds the single precision number at the effective memory address to the contents of the floating point accumulator and places the sum in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

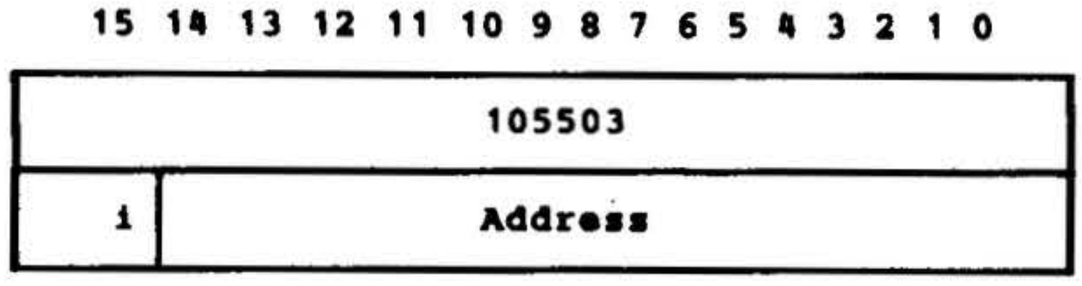
FIX Reformat to Fixed Point



The floating point accumulator is reformated to the 16 bit integer notation and then the integer is stored at the effective memory address.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

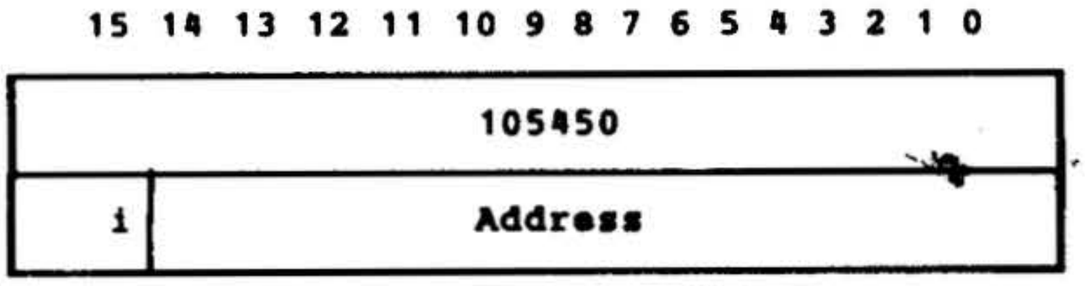
FADD Floating Add Double Precision



Adds the double precision number at the effective memory address to the contents of the floating point accumulator and places the sum in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

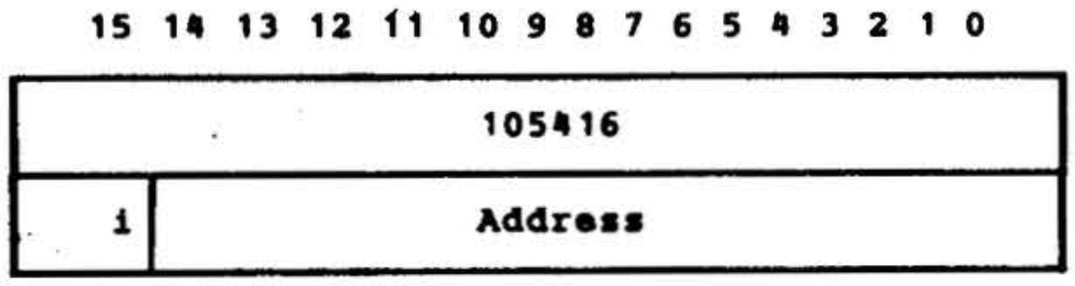
FSB Floating Subtract Single Precision



Subtracts the single precision number at the effective memory address from the floating point accumulator and stores the result in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

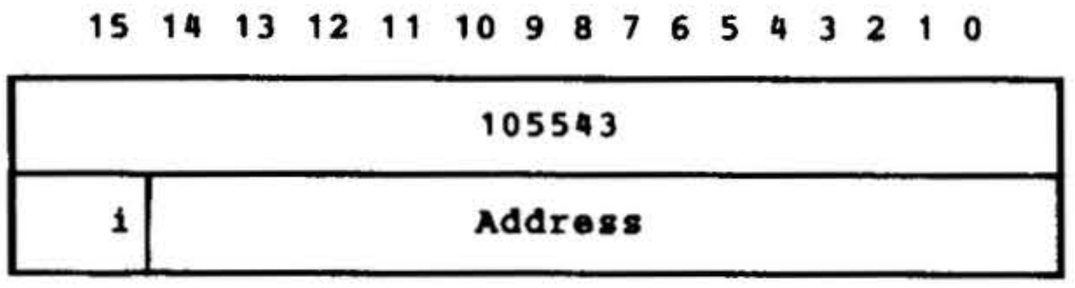
FMU Floating Multiply Single Precision



Multiplies the single precision number at the effective memory address by the contents of the floating point accumulator and stores the result in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

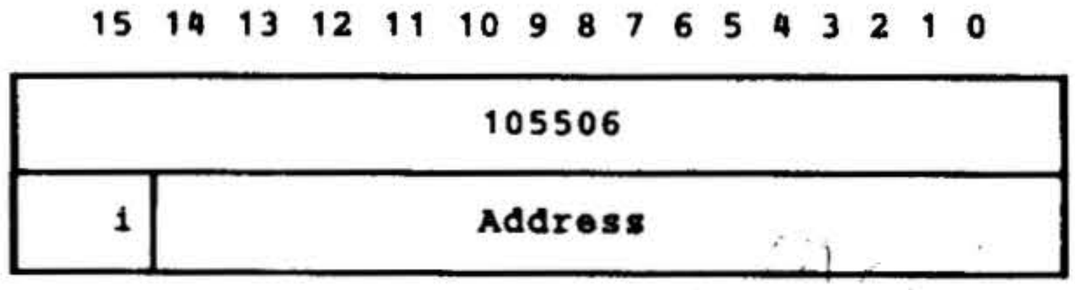
FSBD Floating Subtract Double Precision



Subtracts the double precision number at the effective memory address from the floating point accumulator and stores the result in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

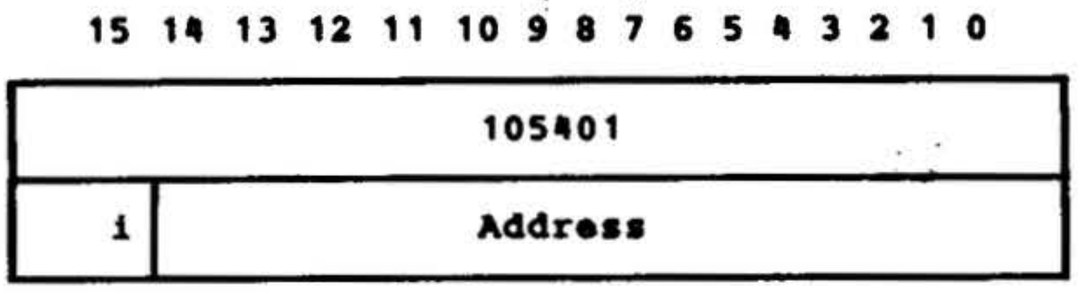
FMUD Floating Multiply Double Precision



Multiplies the double precision number at the effective memory address by the contents of the floating point accumulator and stores the result in the floating point accumulator. The result is normalized.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

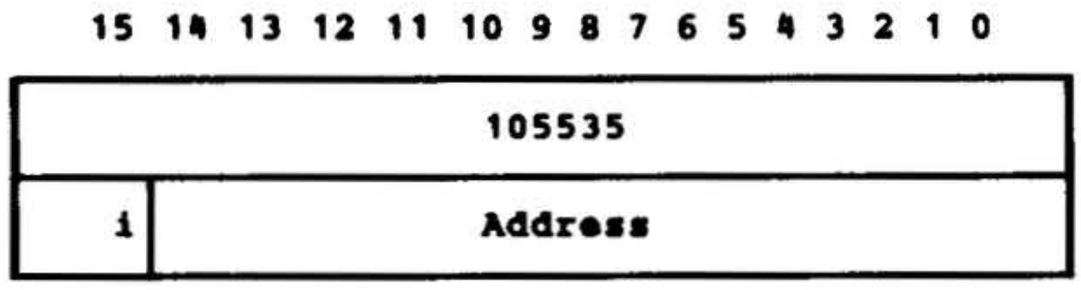
FDV Floating Divide Single Precision



Divides the single precision number at the effective memory address into the number in the floating point accumulator and places the result in the floating point accumulator.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	Floating Point Accumulator

FDVD Floating Divide Double Precision



Divides the double precision number at the effective memory address into the number in the floating point accumulator and places the result in the floating point accumulator.

Relative addressing:	No
Indirect addressing:	Yes
Indexing:	No
Registers altered:	No

SECTION 15 - VARIAN 73 DAS ASSEMBLERS

The **Varian 73 assembler language (DAS)** translates symbolically coded instructions, directives, and data (source program) into their binary machine-language equivalents (object program). DAS allows the programmer to specify instructions, addresses, address modifications, and constants in a manner that is straightforward and meaningful to the computer.

Using DAS, the programmer generates a source program by coding instruction and directive mnemonics rather than numerical values. Memory addresses can be referenced symbolically, thus providing flexibility not attainable with absolute addressing. Constants can be used without prior conversion to binary or octal values. For ease in checkout and program documentation, comments can be added between symbolic source statements, or appended to the statements themselves.

DAS coding reduces machine-language bookkeeping to fully utilize computer capabilities without a corresponding compromise of an increase in the time required for programming.

Two versions of DAS are available:

a. DAS 8A requires a minimum memory and can operate with additional system peripherals.

b. DAS MR is a macro assembler, which produces relocatable object code, that can be loaded into any area of memory. DAS MR is available either as a free-standing program or as an integral part of the MOS or VORTEX/ VORTEX II operating system.

DAS processes source programs in two passes. The first pass defines user-designated symbols. The second pass produces an assembly listing and the object program.

Character Set

The DAS character set comprises:

Alphabetical characters

ABCDEFGHIJKLMNOPQ
RSTUVWXYZ

Numerical characters

0123456789

Teletype characters

CR (Carriage return)
LF (line feed)

Special characters

+	(plus sign)
–	(minus sign)
*	(asterisk)
/	(slash)
.	(period)
	(blank)
@	(at sign)
[	(left bracket)
]	(right bracket)
<	(less than)
>	(greater than)
↑	(up arrow)
←	(left arrow)
=	(equal sign)
,	(comma)
'	(prime)
(	(left parenthesis)
)	(right parenthesis)
\	(backslash)
!	(exclamation point)

"	(quotation mark)
#	(pound sign)
%	(percent sign)
&	(ampersand)
:	(colon)
;	(semicolon)
?	(question mark)
$	(dollar sign)

Format

DAS source programs are sequences of source statements (records). Each source statement comprises a combination of label, operation, variable, and comment fields, depending on the requirements of the computer instruction or assembler directive, and except in certain cases (described later in this section) generates one computer word.

Label Field

Symbols in the label field identify program points for reference by other parts of the program. They make a program point or particular numeric value more easily identifiable. The first appearance of a symbol in the label field establishes its identity throughout the remainder of the program. A previously established symbol is referenced by placing it in the variable field of the source statement, where DAS substitutes the previously assigned value from its symbol table.

For DAS 8A, symbols in the label field comprise one to four alphanumeric characters; for DAS MR there are from one to six such characters. The first character of a symbol is an alphabetic character, pound sign (#), or dollar sign (the dollar sign and pound sign are used in the Varian software and should not be used in normal users programs).

While only the given number of characters are recognized by DAS, additional characters can be added for programming convenience and/or documentation.

Symbols are usually attached only to those source statements referenced elsewhere in the program, but this is not mandatory.

Operation Field

This source statement field contains mnemonics for computer instructions (section 14) and assembler directives (defined later in this section). An asterisk following the mnemonic specifies **indirect addressing** (section 13). The mnemonics can be redefined with OPSY assembler directives.

Variable Field

The purpose of this field varies according to the requirements of the operation defined by the source statement. The variable field can contain a symbol, a constant, or an expression combining symbols and constants.

DAS expressions are similar to arithmetic expressions except that parentheses are not used. The variable field can contain the following operators.

+	(addition)
−	(subtraction)
*	(multiplication)
/	(division)

Arithmetic operations always involve all 16 bits of the computer words, and are performed from left to right, with multiplication and division occurring before addition and subtraction. Thus, A + B/C * D in DAS is equivalent to A + (B/C) * D in conventional notation.

Coding an asterisk in the first position of the variable field gives access to the then current value of the program location counter. Such an asterisk immediately precedes another operator, and this is the only case in which two adjacent operators are permitted in DAS. The asterisk is translated as the current program location (e.g., * + 1 means the current program location plus one).

In the following descriptions of DAS constants, *unsigned numbers are considered positive* DAS recognizes decimal and octal integers; floating-point numbers; alpha, address, and indirect address constants; and literals.

A **decimal integer** is a signed or unsigned string of from one to five decimal digits, the first of which cannot be zero (so as not to be confused with octal integers).

Example:

1 20 −3 −9000

An **octal integer** is a signed or unsigned string of from one to seven octal digits, the first of which is zero.

Example:

07 −044 +022745

A **floating-point number** has the form:)± integer.fraction± exponent, where the right parenthesis, at least one digit, and the decimal point are always present. Other items in the format are optional.

Examples:

)03075.64E + 7)9.E− 2,).1E + 12
)− 4. + 20

An **alpha constant** is a string of characters within primes ('), where, within DAS each character is represented in eight-bit ASCII code. Thus, each 16-bit memory address can hold two characters. Note that blanks are also recognized as characters.

In DAS 8A, an alpha constant can be a term in an arithmetic expression. However, if more than one word is generated by the constant, only the last word is subject to arithmetic manipulation.

Examples:

'A'*0400 'AB' + 1 'ABCD' + 011

where, in the last example, two words are generated and 011 added to the second word.

An **address constant** is a symbol, number, or expression enclosed in parentheses. It generates a 15-bit direct address (bit 15 = 0). Addressing modes are discussed in section 12.

Examples:

(aaaa + 2) (31) (aaaa)

where aaaa is an address symbol whose value is taken from the symbol table by DAS.

An **indirect address constant** is an *address constant* followed by an asterisk. It generates a 15-bit indirect address (bit 15 = 1).

Examples:

```
(aaaa+2)*   (3)*   (aaaa)*
```

Literals provide a method for creating and referencing data by expressing the value of the information instead of its address. DAS determines the address and inserts it in the referencing statement and generates a literal table, discarding duplicate values in the table.

A literal is any format of a one-word constant preceded by an equal sign. In a statement requiring more than one literal, they are separated by commas.

Examples:

```
=29    =-044    =(aaaa+2)*
='GO'  ='A'
```

Comments Field

This field is used for programming notes. An entire source statement can be commentary if an asterisk is coded in the first position. The assembler ignores all comments in the assembly process, but lists them with the program listing output.

Computer Instructions

DAS assemblers recognize the complete instruction sets of all Varian 73 computers, even when the system on which they operate lacks the hardware for executing a particular instruction. **The programmer, therefore, must have a thorough knowledge of the instructions applicable to his system before attempting to assemble a program.**

Computer instructions, divided by function, are described in detail in section 14. In appendix A, they are listed, arranged alphabetically by mnemonic, with their octal codes, and indexed to the page of this Handbook in which they are discussed. Instruction formats are given in section 12, and section 13 discusses addressing modes.

In this section, all Varian 73 instructions are divided into five types, according to assembler format requirements.

All Varian 73 instructions in DAS have the general field format

Label Operation Variable Comments

where the label field is optional and contains a symbol when used; the operation field contains the instruction mnemonic; the variable field contains one, two, or three expressions (separated by commas when there is more than one), and the comments field is optional.

Addressing

If an assembler source statement specifies an address in the first 2,048 words of memory without indirect addressing, the assembler generates an instruction with direct addressing.

If indexing is specified, the assembler generates an indexed instruction.

Specifying indirect addressing with a data address lower than 512 generates an instruction with indirect addressing and the specified effective memory address.

In all other cases, including indirect addressing with an address higher than 511, the assembler generates an instruction with indirect addressing and the specified effective memory address, stores the address in a table, and inserts the storage address in the referencing instruction. Duplicate values in the table are discarded.

In the Varian 73, indirect addressing is limited to five levels with one-word instructions and to four levels with two-word instructions.

Instruction Types

Table 15-1 summarizes the characteristics of the five types of computer instructions for DAS use. Instruction mnemonics are given in the applicable type description below and summarized in table 15-2.

Assembler type 1 instructions are:

ADD	LDA	STA
ANA	LDB	STB
DIV	LDX	STX
ERA	MUL	SUB
INR	ORA	

An assembler type 1 instruction occupies one computer word and is memory-addressing.

Indirect addressing is specified by an asterisk after the mnemonic or after a variable field expressed in parentheses.

Examples:

```
LDA*  expression
LDA   (expression)*
```

Indexing is specified by two expressions in the variable field. The first is the indexing increment and is less than 0512. The second specifies the indexing register: X register = 1, and B register = 2. These instructions cannot be postindexed.

Example:

```
LDA        0300,1
```

loads the A register with the contents of the memory address specified by the sum of the X register contents and 0300. Thus, if the X register contains 0200, the operand for this instruction is in memory address 0500.

Assembler type 2 instructions are:

ADDI	JOFN	STXI
ANAI	JOFNM	SUBI
DIVI	JSS1	XAN
ERAI	JSS2	XANZ
INRI	JSS3	XAP
JAN	JS1M	XAZ
JANM	JS1NM	XBNZ
JANZ	JS2M	XBZ
JANZM	JS2NM	XEC
JAP	JS3M	XOF
JAPM	JS3NM	XOFN
JAZ	JXZ	XS1
JAZM	JXZM	XS1N
JBZ	LDAI	XS2
JBZM	LDBI	XS2N
JMP	LDXI	XS3
JMPM	MULI	XS3N
JOF	ORAI	XXNZ
JOFM	STAI	XXZ
	STBI	

Table 15-1. Assembler Instruction Type Characteristics

Parameter	Type 1	Type 2	Type 3	Type 4	Type 5
Words generated	1	2	2	1	2
Memory addressed	Yes	Yes*	Yes	No	Yes
Indirect addressing	Yes	Yes*	Yes	No	Yes
Indexing	Yes	No	No	No	Yes
Variable field expressions	1 or 2	1	2	0 or 1	1 to 3
Microcoding	No	No	Yes	Yes	No

* Except for immediate instructions.

Table 15-2. Summary of Assembler Instruction Types

Type 1	Type 2		Type 3	Type 4		Type 5
ADD	ADDI	JS3NM	BT	AOFA	LASR	ADDE
ANA	ANAI	JXZ	IME	AOFB	LLRL	ANAE
DIV	DIVI	JXZM	JIF	AOFX	LLSR	DIVE
ERA	ERAI	LDAI	JIFM	ASLA	LRLA	ERAE
INR	INRI	LDBI	JMIF	ASLB	LRLB	IJMP
LDA	JAN	LDXI	OME	ASRA	LSRA	INRE
LDB	JANM	MULI	SEN	ASRB	LSRB	JSR
LDX	JANZ	ORAI	XIF	CIA	MERG	LDAE
MUL	JANZM	STAI		CIAB	NOP	LDBE
ORA	JAP	STBI		CIB	OAB	LDXE
STA	JAPM	STXI		COMP	OAR	MULE
STB	JAZ	SUBI		CPA	OBR	ORAE
STX	JAZM	XAN		CPB	ROF	SRE
SUB	JBZ	XANZ		CPX	SEL	STAE
	JBZM	XAP		DAR	SEL2	STBE
	JMP	XAZ		DBR	SOF	STXE
	JMPM	XBNZ		DECR	SOFA	SUBE
	JOF	XBZ		DXR	SOFB	
	JOFM	XEC		EXC	SOFX	
	JOFN	XOF		EXC2	TAB	
	JOFNM	XOFN		HLT	TAX	
	JSS1	XS1		IAR	TBA	
	JSS2	XS1N		IBR	TBX	
	JSS3	XS2		INA	TXA	
	JS1M	XS2N		INAB	TXB	
	JS1NM	XS3		INB	TZA	
	JS2M	XS3N		INCR	TZB	
	JS2NM	XXNZ		LASL	ZERO	
	JS3M	XXZ				

An assembler type 2 instruction occupies two consecutive computer words and is memory-addressing. The second word is the address of a jump, jump-and-mark, execution instruction, or the operand specified by an immediate instruction.

Indirect addressing is specified as with an assembler type 1 instruction. These instructions cannot be indexed.

Assembler type 3 instructions are:

BT	JIFM	SEN
IME	JMIF	XIF
JIF	OME	

An assembler type 3 instruction occupies two consecutive computer words and is memory-addressing. It differs from an assembler type 2 instruction in that the variable field contains two expressions to implement instruction microcoding as described below.

For the JIF, JIFM, JMIF, and XIF instructions, the first expression specifies the conditions required for the jump, jump-and-mark, or execution. The conditions(s) are specified according to the rules given in section 14 and summarized below. As indicated, multiple conditions can be specified by setting additional bits.

Variable Field	Jump/Execute if:
0001	Overflow indicator is set
0002	A register contents are positive
0004	A register contents are negative
0006	NOT condition
0010	A register contents are zero
0020	B register contents are zero
0040	X register contents are zero
0100	SENSE switch 1 is set
0200	SENSE switch 2 is set
0400	SENSE switch 3 is set

Example:

```
JIF     0222,ALFA
```

takes the next instruction from symbolic address ALFA if the A register contains a positive number (0002), the B register contains zero (0020), and SENSE switch 2 is set (0200); i.e., 0002 + 0020 + 0200 = 0222.

For the SEN instruction, the first expression specifies the device address and the I/O function; for IME and OME, the device address.

For the BT instruction, the first expression specifies the register and bit to be tested (section 14).

Example:

```
BT          056,ADDR
```

takes the next instruction from symbolic address ADDR if bit 14 of the A register contents is zero.

Indirect addressing is specified by an asterisk after the mnemonic or after a variable field expression in parentheses as described for the type 1 instructions. **Note:** IME and OME cannot specify indirect addressing.

Assembler type 4 instructions are:

AOFA	EXC2	OAR
AOFB	HLT	OBR
AOFX	IAR	ROF
ASLA	IBR	SEL
ASLB	INA	SEL2
ASRA	INAB	SOF
ASRB	INB	SOFA
CIA	INCR	SOFB
CIAB	IXR	SOFX
CIB	LASL	TAB
COMP	LASR	TAX
CPA	LLRL	TBA
CPB	LLSR	TBX
CPX	LRLA	TXA
DAR	LRLB	TXB
DBR	LSRA	TZA
DECR	LSRB	TZB
DXR	MERG	TZX
EXC	NOP	ZERO
	OAB	

An assembler type 4 instruction occupies one computer word and does not address memory.

For COMP, DECR, INCR, MERG, ZERO, and the register transfer/modification instructions, the assembler generates an instruction as specified by the value in the variable field. This value is determined by coding the summed octal value of the possible binary configurations described for these instructions in section 14.

Example:

```
COMP        035
```

unconditionally takes the inclusive-OR and complements the contents of the A (0010) and B (0020) registers, and places the result in the A (0001) and X (0004) registers. Note that if bit 8 were one in the above example the instruction is executed only if the overflow indicator is set.

For EXC, SEL, EXC2, and SEL2, the expression specifies the I/O function and the device address; for the remainder of the I/O instructions in this group, the device address only (the I/O function being specified by the mnemonic).

Example:

```
CIB         030
```

clears the B register and loads it from peripheral specified by the device address 030 (standard device addresses are given in section 6).

Note: SEL/SEL2 are identical to EXC/EXC2 instructions.

Assembler type 5 instructions are:

ADDE	INRE	SRE
ANAE	LDAE	STAE
DIVE	LDBE	STBE
IJMP	LDXE	STXE
ERAE	MULE	SUBE
JSR	ORAE	

An assembler type 5 instruction occupies two consecutive computer words and is memory-addressing.

***Indirect addressing* is specified by an asterisk after the mnemonic or after a variable field expression in parentheses as described for the type 1 instructions.**

***Preindexing the V73 instructions* is specified as described for the type 1 instructions. Note that IJMP and SRE cannot be preindexed.**

Postindexing the V73 instructions is specified by three expressions in the variable field. The first expression is the data address, the second specifies the indexing register (X register = 1, and B register = 2), and the third is logically ORed with the instruction word to set bit 7 (which specifies postindexing). The assembler does not check the validity of the third expression, thus the value 0200 should always be used.

Example:

```
LDAE        ADDR,2,0200
```

loads the A register extended and postindexed with the B register.

JSR can be neither preindexed nor postindexed.

For SRE, the first expression in the variable field is the data address, the second specifies the type of addressing (1 = indexed with X, 2 = indexed with B, and 7 = direct/indirect), and the third is logically ORed with the instruction word to control bits 3-5 to specify the register to be compared (010 = A register, 020 = B register, and 040 = X register). Note that indirect addressing is specified by an asterisk following the instruction mnemonic.

Examples:

```
SRE         ADDR,7,020
```

compares the contents of the B register with the directly addressed word at ADDR, and, if equal, skips the next two locations.

```
SRE*        ADDR,1,010
```

compares the contents of the A register with the word at ADDR, using indirect addressing and postindexing with the X register.

Assembler Directives

Directives are instructions to the assembler. They are divided into the following functional groups:

- Symbol definition
- Instruction definition
- Location counter control
- Data definition
- Memory reservation
- Conditional assembly
- Assembler control
- Subroutine control
- List and punch control
- Program linkage
- MOS I/O control
- VORTEX I/O control
- Macro definition

Assembler directives have the same general format as the computer instructions. In the following descriptions of the individual directives, the field format

Label Operation Variable

is used, with the optional comment field being understood to follow the variable field when used. In cases where the variable field contains more than one item or expression, these are always separated by commas. Mandatory elements of the directive are in **bold type**, and optional items, in *italic type*.

Table 15-3 summarizes the assembler directives (arranged by function) and indicates those recognized by each DAS assembler.

Table 15-3. Directives Recognized by DAS Assemblers

Function	Directive	DAS 8A	DAS MR
Symbol definition	EQU	Yes	Yes
	SET	Yes	Yes
	MAX	Yes	No
	MIN	Yes	No
Instruction definition	OPSY	Yes	Yes
Location counter control	ORG	Yes	Yes
	LOC	Yes	Yes
	BEGI	Yes	No
	USE	Yes	No
Data definition	DATA	Yes	Yes
	PZE	Yes	Yes
	MZE	Yes	Yes
	FORM	Yes	Yes
Memory reservation	BSS	Yes	Yes
	BES	Yes	Yes
	DUP	Yes	Yes
Conditional assembly	IFT	Yes	Yes
	IFF	Yes	Yes
	GOTO	Yes	Yes
	CONT	Yes	Yes
	NULL	Yes	Yes
Assembler control	MORE	Yes	No
	END	Yes	Yes
Subroutine control	ENTR	Yes	Yes
	RETU*	Yes	Yes
	CALL	Yes	Yes
List and punch control	LIST	Yes	No
	NLIS	Yes	No
	SMRY	Yes	Yes
	DETL	Yes	Yes
	PUNC	Yes	No
	NPUN	Yes	No
	SPAC	Yes	Yes
	EJEC	Yes	Yes
	READ	No	No

Table 15-3. Directives by DAS Assemblers *(continued)*

Function	Directive	DAS 8A	DAS MR
Program linkage	NAME	No	Yes
	EXT	No	Yes
	COMN	No	Yes
MOS I/O control	Applicable to DAS MR **only**, refer to Varian 620 MOS Manual, document number 98 A 9952 09x.		
VORTEX I/O control	OPEN	No	Yes
	CLOSE	No	Yes
	READ	No	Yes
	WRITE	No	Yes
	REW	No	Yes
	WEOF	No	Yes
	SREC	No	Yes
	FUNC	No	Yes
	STAT	No	Yes
	DCB	No	Yes
	FCB	No	Yes
Macro definition	MAC	No	Yes
	EMAC	No	Yes

Symbol Definition Directives

These directives assign arbitrary values to symbols in the symbol table. This table is a list of symbols appearing in the source program. For each symbol in the table, there is a corresponding value, usually an address in memory. Symbol table capacities are summarized in table 15-4.

Table 15-4. DAS Symbol Table Capacities

Assembler	8K Memory	> 8K Memory
DAS 8A	440	440 + n (800)
DAS MR	20	20 + n (800)

where n = number of 4K memory increments above 8K.

EQU (DAS 8A, DAS MR)

This directive has the format

symbol EQU expression

It places the **symbol** in the assembler's symbol table and assigns it the value of the **expression**. If the symbol has already been entered in the symbol table, DAS outputs error message ***DD** (described later in this section), and the expression replaces the value in the symbol table. If a symbol is used as the variable field expression, it must have been previously defined. **The label field symbol is mandatory.**

SET (DAS 8A, DAS MR)

This directive has the format

symbol SET expression

It is the same as EQU, except that there is no error message output if the symbol has already been entered in the symbol table.

MAX (DAS 8A)

This directive has the format

symbol MAX expression,*expression(s)*

It assigns the largest algebraic value found among the **expressions** to the **symbol**. If a symbol is used as a variable field expression, it must have been previously defined. **The label field symbol is mandatory.** Use SET to redefine the symbol.

MIN (DAS 8A)

This directive has the format

symbol MIN expression,*expression(s)*

It is the same as MAX, except that the symbol is assigned the smallest algebraic value found among the **expressions**.

Instruction Definition Directive

This directive redefines a standard instruction mnemonic.

OPSY (DAS 8A, DAS MR)

This directive has the format

symbol OPSY mnemonic

It makes the symbol a mnemonic with the same definition as the variable field mnemonic.

Example:

```
CLA   OPSY LDA
      CLA  BETA
```

Location Counter Control Directives

These directives control the program location counter(s), which control memory area assignments and always point to the next available word.

DAS 8A has several location counters and directives to modify or preset their values. Table 15-5 lists the five standard DAS 8A location counter symbols and their uses. They need not be created by the user. However, up to eight other location counters can be created, thus providing complex relocatable and overlay programs within a single assembly. Relocatability rules are given later in this section.

There are no user-created location counters at the beginning of an assembly. The assembler uses three location counters for program location assignment. Thus, IAOR (indirect pointer assignments) and LTOR (literal assignments) are always in use, as is a third counter used to assign locations to generated instructions and data. The blank location counter performs this task until the USE directive specifies another counter.

In a straightforward program using only one location counter, the ORG and LOC directives completely control the counter.

ORG (DAS 8A, DAS MR)

This directive has the format

symbol **ORG expression**

It sets the location counter currently in use to the value of the **expression**. If a symbol is present in the label field, it is also set to the value of the expression.

Any symbol used as the variable field expression must have been previously defined.

LOC (DAS 8A, DAS MR)

This directive has the format

symbol **LOC expression**

It is used if the data and instructions following this LOC address are to be moved to the LOC address by the object program before execution, i.e., to keep a block of data or instructions undisturbed by assembly. Data or instructions following LOC are generated as if an ORG directive had changed the current location counter value. However, this value is not actually changed.

Any symbol used as a variable field expression must have been previously defined. LOC cannot be used in a relocatable program.

BEGI (DAS 8A)

This directive has the format

symbol **BEGI expression**

It creates a new location counter, or redefines the value of any location counter before the counter has been used. BEGI gives the new or redefined location counter the value of the **expression**, but has no effect on the current location counter.

BEGI cannot redefine the value of any location counter that has been used for location assignment.

Any symbol used as a variable field expression must have been previously defined.

Table 15-5. Standard DAS 8A Location Counters

Counter	Initial Value	Description
COMN	002000	Controls assignment of memory within an interface area common to two or more programs
IAOR	000200	Controls assignment of memory to indirect pointers
LTOR	001000	Controls assignment of memory to literals
SYOR	000000	Controls assignment of memory to all system parameters
blank	004000	Used initially and normally by the assembler for memory assignments until/unless overridden by the use of ORG directive

USE (DAS 8A)

This directive has the format

blank USE xxxx

where **xxxx** is a blank, COMN, SYOR, or a user-created location counter label.

The USE directive uses location counter **xxxx** to assign locations to data and instructions (except literals and indirect pointers).

If **xxxx** is PREV, the previously used location counter is recalled with the restriction that only the last-used counter can be so recalled.

Data Definition Directives

These directives control the sign and assignment of data words. In the descriptions, **item** refers to a data item, which can be an expression or a direct or indirect address.

DATA (DAS 8A, DAS MR)

This directive has the format

symbol **DATA item,***item(s)*

It generates data words with the values specified by the **items** in the variable field. DATA assigns the *symbol*, if used, to the memory address of the first generated word. In the absence of a symbol, an unlabeled block of data is generated.

When a single alpha constant is used in the variable, DAS MR left-justifies it in the field and fills the remaining positions with blanks, and DAS 8A right-justifies it, filling the remaining positions with zeros.

PZE (DAS 8A, DAS MR)

This directive has the format

symbol **PZE item,***item(s)*

It is similar to DATA except that the sign bit of the generated data word is always zero (positive).

MZE (DAS 8A, DAS MR)

This directive has the format

symbol **MZE item,***item(s)*

It is similar to DATA except that the sign bit of the generated data word is always one (negative).

FORM (DAS 8A, DAS MR)

This directive has the format

symbol **FORM term,***term(s)*

where the **terms** are absolute terms or expressions.

FORM specifies the format of a bit configuration of a data word. The *symbol*, if used, is the name of the format. The **terms** specify the length in bits of each field in the generated data word, where the sum of their values is from one to the number of bits in the computer word.

FORM is ignored if there are any errors in the variable field, except that an error is flagged when a **term** cannot be represented in the number of bits specified when FORM is applied (by placing its name in the operation field of a symbolic source statement) to another statment. FORM can be redefined.

Example:

```
BYTE    FORM    8,8
BCD     FORM    4;4,4,4
PTAB    FORM    1,2,3,4
```

would, given the FORM definition

```
ABC     FORM    6,2,8
```

and the FORM reference

```
        ABC     2*3,1,'A'
```

generate the binary data word

0 001 100 111 000 001

Memory Reservation Directives

These directives control the reservation of memory addresses and areas.

BSS (DAS 8A, DAS MR)

This directive has the format

symbol **BSS expression**

It reserves a block of memory addresses by increasing the value of the current location counter the amount indicated by the **expression**. The *symbol*, if used, is assigned the value of the counter prior to such an increase, thus referencing the starting address of the reserved block.

The location counter always points to the next available word.

If the variable field expression value is zero, the *symbol* is assigned the next available address.

BES (DAS 8A, DAS MR)

This directive has the format

symbol **BES expression**

It is similar to BSS, except that if there is a symbol it is assigned to the address one less than the incremented location counter. If the variable field expression is zero, the symbol is assigned the last available address.

DUP (DAS 8A, DAS MR)

This directive has the formats

blank DUP n
blank DUP n,m

It duplicates source statements following its use. The first format duplicates the next source statement the number of times specified by **n**. The second format duplicates the next source statement (the number of which is specified by **m**) the number of times specified by **n**, where **m**, ≤ 3 and **n** $\leq 32{,}767$. If **n** or **m** is zero, it is treated as if it were a one.

Conditional Assembly Directives

These directives assemble portions of the program according to the conditions specified in the variable fields.

IFT (DAS 8A, DAS MR)

This directive has the format

blank IFT expression,*expression(s)*

It assembles the next symbolic source statement only if the first **expression** is less than the second, and the second is less than or equal to the third.

Examples:

```
        IFT     a
```

for a ≠ 0.

```
        IFT     a,,b
```

for a≠ b.

```
        IFT     a,b,b
```

for a< b.

```
        IFT     0,a,b
```

for 0< a≤ b.

IFF (DAS 8A, DAS MR)

This directive has the format

blank IFF expression,*expression(s)*

It is similar to IFT (IFT = true), except that IFF (IFF = false) is the logical complement of IFT.

Examples:

```
        IFF     a
```

for a = 0.

```
        IFF     a,,b
```

for a = b.

```
        IFF     a,b,b
```

for a ≥ b.

```
        IFF     0,a,b
```

for 0 ≥ a > b.

GOTO (DAS 8A, DAS MR)

This directive has the formats

blank GOTO symbol
blank GOTO symbol,
blank GOTO integer
blank GOTO integer,

It skips more than one instruction and usually follows an IFF or IFT directive. All source statements between the GOTO and the statement containing the **symbol** in its label field are skipped, and the instruction so labeled executed next. GOTO cannot return to an earlier point in the program.

If the first and third GOTO formats are used, the skipped instructions are listed. If the second and fourth formats (containing a comma after the variable field element) are used, they are not listed. This listing can also be suppressed by a SMRY directive.

CONT (DAS 8A, DAS MR)

This directive has the format

symbol CONT blank

It provides a target for a previous GOTO directive. The **symbol** is not entered in the assembler's symbol table.

NULL (DAS 8A, DAS MR)

This directive has the format

symbol NULL blank

It provides a target for a previous GOTO directive with the **symbol** entered in the symbol table. NULL has the same effect as a BSS directive with a blank variable field.

Assembler Control Directives

These directives signal the end or continuance of an assembly.

MORE (DAS 8A)

This directive has the format

blank MORE blank

It halts the assembly process to allow additional source statements to be put in the input device. Assembly resumes when the RUN or START switch on the computer control panel is pressed. MORE is never listed.

END (DAS 8A, DAS MR)

This directive has the format

blank END *expression*

It is the last source statement in the program. The *expression* is the execution address of the program after it has been loaded into the computer. A blank in the expression field yields an execution address of 000000.

Subroutine Control Directives

These directives create closed subroutines and control their use.

ENTR (DAS 8A, DAS MR)

This directive has the format

symbol ENTR blank

where the **symbol** is the name of the subroutine called. ENTR generates a linkage word of zero in the object program.

RETU* (DAS 8A, DAS MR)

This directive has the format

symbol **RETU*** **expression**

It returns from a closed subroutine, generating an unconditional jump to the address indicated by the value of the expression.

CALL (DAS 8A, DAS MR)

This directive has the format

symbol **CALL** **name**,*parameter*,*error*

where

name	is the symbolic name of a subroutine
parameter	is an optional list of parameters comprising valid data items
error	is an optional list of error returns comprising valid data items

CALL causes the program to jump to the closed subroutine specified by **name**. Where a *symbol* is used in the label field, it is entered in the symbol table and assigned the value of the current location counter.

Example:

```
CALL    FUNC,X,Y + 1,(ERR),(GOOF)*
```

produces a machine code identical to that obtained with

```
JMPM        FUNC
.
.
.
DATA        X,Y,+1,(ERR),(GOOF)*
.
.
.
```

List and Punch Control Directives

These directives, which are operative only during the second pass of the assembler (that producing the object program and listings), control listing and punching during program assembly.

LIST (DAS 8A)

This directive has the format

blank LIST blank

It causes the assembler to produce a program listing. The assembler normally outputs a list of the source statements. The LIST directive is used to bring the assembler back to this condition when any of the following directives change the listing status.

NLIS (DAS 8A)

This directive has the format

blank NLIS blank

It suppresses further listing of the program.

SMRY (DAS 8A, DAS MR)

This directive has the format

blank SMRY blank

It suppresses the listing of source statements that have been skipped under control of the conditional assembly directives.

DETL (DAS 8A, DAS MR)

This directive has the format

blank DETL blank

It removes the effect of SMRY, i.e., causes listing of all source statements, including those skipped by conditional assembly directives.

PUNC (DAS 8A)

This directive has the format

blank PUNC blank

It causes the assembler to produce a paper tape punched with the object program. The assembler normally outputs such a tape. PUNC returns the assembler to this condition when the following directive changes the punching status:

NPUN (DAS 8A)

This directive has the format

blank NPUN blank

It suppresses further production of paper tape punched with the object program.

SPAC (DAS 8A, DAS MR)

This directive has the format

blank SPAC blank

It causes the listing device to skip a line. SPAC is not listed.

EJEC (DAS 8A, DAS MR)

This directive has the format

blank EJEC blank

It causes the listing device to move to the next top of form. EJEC is not listed.

READ (DAS 8A)

This directive has the format

blank READ *number*

where

number	is the number of characters (20 to 80) from each source statement to be processed by the assembler

Normally, the assembler processes 80 characters per statement with 026 keypunch codes. If *number* is outside the range 20 to 80, the assembler resets the number of characters to 80 and outputs error message ***SZ.**

Program Linkage Directives

These directives establish and control links among programs that have been assembled separately but are to be loaded and executed together.

NAME (DAS 8A, DAS MR)

This directive has the format

blank NAME symbol,...,*symbol*

It establishes linkage definition points among separately assembled programs. Each **symbol(s)** can then be referenced by other programs. Each **symbol** also appears in the label field of a symbolic source statement in the body of the program. Undefined NAME symbols cause error messages to be output.

Examples:

```
NAME      A
NAME      A,B
NAME      EX,WHY,ZEE
```

EXT (DAS 8A, DAS MR)

This directive has the format

symbol **EXT** *symbol,...,symbol*

In linking separately assembled programs, it declares each *symbol* not defined within the current program. Each *symbol*, in both the label and variable field, is output to the relocatable loader with the address of the last reference to the symbol.

If a symbol is not defined within the current program and not declared in an EXT directive, it is considered undefined and causes an error message output. If a symbol is declared in EXT but not referenced within the current program, it is output to the loader for loading, but no linkage to this program is established. If a symbol is both defined in the program and declared to be external, the EXT declaration is ignored.

Examples:

```
        EXT     AY
BEG     EXT     BE,SEE
        EXT     DEE,EE,FF,GEE
```

COMN (DAS 8A, DAS MR)

This directive has the format

symbol **COMN item**

where **item** is an absolute item or expression.

COMN defines an area in blank common for use at execution time. This allows an assembler program to reference the same blank common area as a FORTRAN program. The common area is cumulative for each use of COMN, i.e, the first COMN defines the base area of the blank common, the second COMN defines an area to be added to the already established base, etc.

Examples:

```
AAA     COMN    3
        COMN    6*2
BBB     COMN    9
```

MOS I/O Control Directives

As a free-standing program or under MOS, DAS MR accepts the MOS control directives listed below and explained in the Software Handbook (98 A 9952 20x) in the Master Operating System section.

Directive	Description
RBIN	Read binary record
RALF	Read alphanumeric record
RBCD	Read binary-coded decimal (BCD) record
WBIN	Write binary record
WALF	Write alphanumeric record
WBCD	Write BCD record
WEOF	Write end of file
REW	Rewind
SKFF	Skip files forward
SKFR	Skip files reverse
SKRF	Skip records forward
SKRR	Skip records reverse
FUNC	Function
STAT	Status
ION	I/O driver reference number

VORTEX I/O Control Directives

DAS MR accepts the VORTEX control directives that are listed below and explained in the VORTEX Reference Manual (document number 98 A 9952 10x).

Directive	Description
OPEN	Open file
CLOSE	Close file
READ	Read one record
WRITE	Write one record
REW	Rewind
WEOF	Write end of file
SREC	Skip one record
FUNC	Function
STAT	Status
DCB	Generate data control block
FCB	Generate file control block

Macro Definition Directives

These directives begin and end macro definitions. The macro is the assembly equivalent of the execution subroutine. It is defined once and can then be called from the program. The macro is an algorithmic statement of a process that can vary according to the arguments supplied. It is assembled with the resultant data inserted into the program at each point of reference, whereas the subroutine executed during execution time appears but once in a program. Its definition comprises the statements between MAC and EMAC.

MAC (DAS MR)

This directive has the format

symbol MAC blank

It introduces a macro definition. The **symbol** is the name of the macro.

EMAC (DAS MR)

This directive has the format

blank EMAC blank

It terminates the definition of a macro.

A macro is called by the appearance of its name in the operation field of a symbolic source statement. The variable field of this statement contains expression(s) P(1), P(2),...P(n), then processed with the values in the table being substituted for the respective values of the expressions in the source statement variable field. For example, if the variable field of the symbolic source statement contains

2,B,9 + 8, = 63

then within the generated macro P(1) = 2, P(2) is the value of B, P(3) = 021, and P(4) is the address of the value 63. All terms and expressions within the macro-referencing symbolic source statement parameter list are evaluated prior to calling the macro.

If the label field of such a source statement contains a symbol, the symbol is assigned the value and relocatability of the location counter at the time the macro is called but before data generation.

A macro definition can contain references to machine instruction mnemonics or to assembler directives other than DUP. Macros can be nested within macros to a depth limited only by the available memory at assembly time.

Example: Define the macro.

```
SBR   MAC
      SEN   0200 P(1),* +3
      JMP   *-2
      EMAC
```

Call the macro.

```
      SBR   031
```

Expand the macro.

```
      SEN   0231,* +3
      JMP   *-2
```

P(0) can also be accessed by a normal call. P(0) is the first entry in the table formed by the assembler and contains the number of entries in that table. Figure 15-1 shows the output listing obtained by calling P(0).

```
                                    1       A       MAC
                                    2               DATA        P(0)
                                    3               EMAC
000001          000000A             4               A
000002          000001A             5               A           1
000003          000002A             6               A           1,2
000004          000003A             7               A           1,2,3
000005          000004A             8               A           1,2,3,4
000006          000005A             9               A           1,2,3,4,5
                                    10              END
```

Figure 15-1. P(0) Output Listing

Symbol And Expression Modes

Each symbol or expression has one of the following modes assigned by the assembler:

a. External (E)

b. Common (C)

c. Relative (R)

d. Absolute (A)

The mode of an expression is determined by the mode of the symbols in the expression.

The mode of a symbol is determined by the following rules:

a. If the symbol is in an EXT directive, the mode is E.

b. If the symbol is defined by a COMN directive, the mode is C.

c. If the symbol is a symbol in a program, or if * is the current location counter value, the mode is R.

d. If the symbol is a number (numerical constant), the mode is A.

e. If the symbol is defined by an EQU, SET, or similar directive, the mode of the symbol is that of the variable field expression in the directive.

The mode of an expression is determined by the following rules:

a. If the expression contains any mode E or C symbol, the expression is mode E.

b. If the expression contains only mode A symbols, the expression is mode A.

c. If the expression contains mode and R symbols, the mode of the expression is R if there is an odd number of mode R symbols. Otherwise, the mode of the expression is A.

The following restrictions apply only to DAS MR and to FORTRAN-compatible output assembly with DAS 8A.

a. No expression can contain symbols of both modes E and C.

b. A mode E expression comprises a single mode E symbol.

c. No mode E, C, or R expression can multiply or divide a mode E or C symbol.

```
EEEE     EXT                    Defines mode E
CCCC     COMN     6             Defines mode C
RTN      ENTR                   Defines a symbol (RTN) as a mode R
TBL      BSS      50            TBL is mode R
ABL      BSS      'A'+5         ABL is mode R
LENG     EQU      *-TBL         LENG is mode A (defines area length)
         CALL     EEEE,TBL,LENG
         LDA      *+6           Legal, one-word relative forward
         LDA      CCCC+6        Illegal, one-word not R or A
         LDXI     CCCC+6        Legal, two-word instruction
         LDA      0,1           Legal, loads CCCC+6 in A register
         .
         .
         .
         DATA     EEEE+4        Illegal, value not zero
         DATA     CCCC+4        Legal
         DATA     CCCC+LENG     Legal
         DATA     TBL+LENG 5    Legal, mode is R
```

Figure 15-2. Manipulation of Expression and Symbol Modes

d. No expression can add or subtract a mode C and a mode R symbol, or a mode E and a mode R symbol.

e. No expression can add two or more mode E, C, or R symbols.

f. A mode A symbol can be added to or subtracted from a mode C or R symbol.

Figure 15-2 illustrates the above rules.

Relocatability Rules

A relocatable program (DAS 8A, DAS MR) is one that has been assembled with its instruction and directive locations assigned in such a manner that it can be loaded and executed anywhere in memory. When such a program is loaded, the beginning memory address is specified, and a value (known as the relocation bias) is added to the addresses of subsequent relocatable instructions. The programs are usually assembled with a zero relocation bias on the first instruction.

The location counter contains the (relative) address of the instruction or directive currently being executed. The location counter is absolute when it contains the actual address of the instruction, and relocatable when it contains the relative address (the current address of the start of the program).

Symbols can be absolute or relocatable. Expressions, since they contain symbols, can be absolute or relocatable. Constants are always absolute.

Figure 15-3 shows, for each arithmetic operation, whether the result is absolute (abso), relocatable (relo), or illegal.

	A = abso B = abso	A = abso B = relo	A = relo B = abso	A = relo B = relo
A + B	abso	relo	relo	illegal
A– B	abso	illegal	relo	abso
A*B	abso	illegal	illegal	illegal
A/B	abso	illegal	illegal	illegal

Figure 15-3. Arithmetic Operation Results

The relocatable loader can load a program in any area of memory and modify the addresses as it loads so that the resulting program executes correctly. Programs can contain absolute addresses, relocatable addresses, or both. At the beginning of each instruction or data word generated by the assembler, it can be set by the ORG directive. On encountering an ORG directive, the assembler makes the location counter absolute if the corresponding expression is absolute, or relocatable if the corresponding expressions is relocatable.

If a symbol is equated to the location counter, it is relocatable if the location counter is relocatable. Otherwise, the symbol is absolute.

Assembler Input Media

Punched Card Format

Punched cards used as input to the DAS assemblers contain four fields corresponding to the instruction and directive fields:

a. The **label field** is in columns 1 through 6. Its use is governed by the requirements of the instruction or directive.

b. The **operation field** is in columns 8 through 14. It contains the instruction or directive mnemonic. Indirect addressing is specified by an asterisk following the mnemonic.

c. The **variable field** begins in column 16 and ends with the first blank that is not part of a character string. Its use depends on the instruction or directive. If two or more subfields are present, they are separated by commas.

d. The **comment field** fills the remainder of the card. If the variable field is blank, the comment field begins in column 17.

An asterisk in column 1 indicates that the entire card contains a comment.

Note that columns 7 and 15 are always unpunched (blank).

Paper Tape Format

Paper tape used as input to the DAS assemblers contains source statements of up to 80 characters each (not including the carriage return and line feed characters). Each punched statement contains four fields corresponding to the instruction and directive fields. The label, operation, and variable fields are separated by commas, and the comment field starts after the first variable field blank that is not part of a character string. Each statement is terminated by a carriage return (CR) followed by a line feed (LF).

a. **Label field** use is governed by the requirements of the instruction or directive. It is terminated with a comma. If this field is not used, a comma appears as the first character of the source statement.

b. The **operation field** contains the instruction or directive mnemonic. An asterisk following the mnemonic specifies indirect addressing. This field begins immediately following the label field terminator and is terminated by a comma.

c. The **variable field** can be blank, or contain one or more subfields separated by commas. It must immediately follow the instruction field terminator (,). Subfields can be voided by using adjacent commas. This field is terminated by a blank that is not part of a character string, or with a CR or LF.

d. The **comment field** fills the remainder of the statement (from the terminating blank of the variable field to the next CR or LF).

If the first nonblank character of a source statement is an asterisk, the entire statement is a comment.

Assembler Output Listing

DAS produces a source/object listing of the assembled program, as well as a paper tape containing the object program in reloadable format.

The listing can be obtained in whole or in part as the program is being assembled. The source (symbolic) program and the object (absolute) program are listed side by side on the listing device. This device is either a Teletype or a line printer.

The listing is output according to the specifications given by the list and punch control directives in the assembly (DAS 8A, DAS MR).

Error analysis during assembly causes the error messages described below to be output on the line following the point of detection.

Figure 15-4 illustrates the format of the output listing. A line count appears only on DAS MR listings. The addressing modes are: FORTRAN common reference = C, externally defined = E, indirect pointer = I, and absolute or relative = R.

Address	Code	Mode	Line Count	Symbolic Source Statement
014000				ORG 014000
014000	000000		ABS	ENTR
014001	001002			JAP* ABS
014002	114000	R		
014003	005211			CPA
014004	001000			JMP* ABS
014005	114000	R		
	000000			END

Figure 15-4. Output Listing Format

Error Messages

The assembler checks source statement syntax during both pass 1 and 2. Detectable errors are listed during pass 1. During pass 2, the following information is listed:

a. Error code

b. Location counter value

c. Object code when the instruction is assembled

This information is suppressed by NLIS directives and list-suppression commas in GOTO directives.

The error message appears in the listing line following the statement found to be in error. Each line can hold up to four error messages.

Table 15-6 lists the DAS error codes and their meanings.

Table 15-6. DAS Error Codes

Code	Meaning
***AD**	Error in an address expression
***DC**	Decimal character in an octal constant
***DD**	Illegal redefinition of a symbol or the location counter
***E**	Incorrectly formed statement
***EX**	Illegally constructed expression
***FA**	Floating-point number contains a format error
***IL**	First nonblank character of a source statement is invalid (the statement is not processed)
***NR**	No memory space available for additional entries in assembler tables
***NS**	No symbol in the label field of a SET, EQU, MAC, or FORM directive or no symbol in the label or variable field of an OPSY directive, or no symbol in the variable field of a NAME directive
***OP**	Undefined operation field (two No Operation (NOP) instructions are generated in the object program; the remainder of the statement is not processed), or illegal nesting of DUP or MAC directives or DUP of a macro call

Table 15-6. DAS Error Codes *(continued)*

Code	Meaning
*QQ	Illegal use of prime (')
*R	Relocatable item where an absolute item should be defined
*SE	Synchronization error: symbol value in pass 2 is different from that found in pass 1
*SY	Undefined symbol in an expression
*SZ	Expression value too large for a subfield, or a DUP directive specifies that more than three statements are to be assembled
*TF	Undefined or illegal indexing specification
*UC	Undefined character in an arithmetic expression
*UD	Undefined symbol in the variable field of a USE directive
*XR	Address out of range for an indexing specification
* =	Illegal use of a literal

Operating The Assembler

DAS 8A Operations

Load the assembler program supplied by Varian into memory using the binary load/dump program (BLD II, section 16). Execute it by entering a positive, nonzero value in the A register during loading, or by clearing all registers, pressing (SYSTEM) RESET and entering the RUN state. (Set RUN indicator on and press START.)

During execution, the program first determines the amount of memory required. It then stores in address 000003 a value one less than the lower limit of BLD II. This is the highest address that the assembler can use without destroying part of BLD II.

DAS 8A comprises two sections: The I/O section allows the specification of I/O devices for assembler input and output. The second section is the assembler itself.

I/O Section Definitions

The I/O section of DAS 8A, using the Teletype printer, makes three requests for definitions of I/O devices:

```
ENTER DEVICE NAME FOR xx
```

where xx is one of the I/O function names: SI (source input), LO (list output), or BO (binary output), respectively.

Respond to each request in turn by typing, on the Teletype keyboard, the name of the

desired device, followed by a carriage return (CR). Table 15-7 lists the acceptable device names in response to each request. If the default assignment is desired, merely press CR.

If an incorrect device name is typed, the message

```
DEVICE NAME NOT VALID
```

is output and the request repeated.

To terminate the output of any line to the Teletype, press RUBOUT. This error correction feature can be used any time during I/O device specification.

When I/O assignments are complete, the I/O section uses BLD II to load the assembler section into memory.

To restart the I/O section before the assembler section is loaded, set STEP indicator on, clear all registers, press (SYSTEM) RESET, set RUN indicator on and press START.

Assembler Section Definitions

When BLD II relinquishes control to the assembler section, the computer halts with 000001 in the program counter (P register). For an assembler pass 1, set SENSE switch 1; for pass 2, reset SENSE switch 1 and set SENSE switches 2 and 3.

If pass 1 is selected, ready the SI device with the source input media and set RUN indicator on and press START.

For pass 2, ready the SI device with the source input media, ready the BO and LO devices, set RUN indicator on and press START.

The END directive terminates both passes 1 and 2. Pass 1 terminates with 000001 in the P register and 0177777 in the A register. Pass 2 produces the binary object loader text and program listing and terminates when END is encountered with the same register values as pass 1. A MORE directive causes the computer to stop and wait until the SI unit is prepared with the

Table 15-7. Acceptable I/O Devices

Assembly Function	Device	Default Assignment
SI (source input)	Teletype paper tape read: TR Teletype keyboard: TY High-speed paper tape reader: PR Card reader: CR Magnetic tape: MTnn	TR
LO (list output)	Teletype printer: TY Line printer: LP2 (70-6701)	TY
BO (binary output)	Teletype paper tape punch: TP High-speed paper tape punch: PP Card punch: CP Magnetic tape: MTnn	TP

additional source input media, and the RUN state is entered. MORE is indicated by 0170017 in the A register.

The program listing can be suppressed during pass 2 by resetting SENSE switch 2, and the binary output, resetting SENSE switch 3. Error messages cannot be suppressed and are output on the LO device as the error is detected during pass 2.

Synchronization errors (table 15-6) halt the assembley with 000777 in the A register. To continue the assembly, set RUN indicator and press START. The assembler resets the location counter value to that assigned on pass 1, prints error message ***SE**, and continues the assembly.

Pass 2 can be restarted or repeated for extra copies of the assembled program without repeating pass 1.

At the completion of pass 2, the assembler can accept another assembly using the same I/O devices. For other I/O devices, reload the assembler program, starting with the I/O section.

To restart the assembler, set STEP indicator on, clear all registers, press (SYSTEM) RESET, set RUN indicator on and press START. The assembler halts with 000001 in the P register and is ready to accept another assembly.

Using Magnetic Tape

The DAS 8A assembler can communicate with any one of the magnetic tape transports on a controller. Up to four transports may be connected to each of the tape controllers. A configuration may have one to four magnetic tape controllers.

The magnetic tape transport number and controller device address is specified in the device name specification of the I/O Control Section based upon the following table:

Device Name	Address (in octal)	Transport Number
MT00	010	1
MT01	010	2
MT02	010	3
MT03	010	4
MT10	011	1
MT11	011	2
MT12	011	3
MT13	011	4
MT20	012	1
MT21	012	2
MT22	012	3
MT23	012	4
MT30	013	1
MT31	013	2
MT32	013	3
MT33	013	4

DAS MR Operations

Since DAS MR operates under MOS or VORTEX and uses the MOS or VORTEX I/O control system, the I/O devices can be defined as required (refer to MOS manual, document number 98 A 9952 09x, or the VORTEX manual, 98 A 9952 10x).

DAS MR inputs the symbolic source statements from the processor input (PI) logical unit in alphanumeric mode, and outputs them in the same mode on the processor output (PO) logical unit. When DAS MR detects the END directive, it terminates pass 1, returns to the beginning of the source program, and begins pass 2. During

pass 2, the source statements are the input from the system scratch (SS) logical unit, a listing is output on the LO unit, and the binary object program is output on the BO unit. Note that PO and SS must be the same magnetic tape, drum, or disc unit.

For an assembly without a program listing, input the following directive to the MOS executives when requesting the assembly:

```
/ASSEMBLE        N
```

For a binary object program, input

```
/ASSEMBLE        B
```

If the memory map portion (symbol table, external names, and entry names) is not wanted, input

```
/ASSEMBLE        M
```

To read the same physical symbolic source statements for both assembly passes, input

```
/ASSIGN          PO=DUM, SS=PI
/ASSEMBLE
```

The processor output listing serves as a copy of the program; it can be input for another assembly.

With an operating system, the DAS MR user gains the facilities provided in either MOS or VORTEX. The features of MOS are described in a later section, in this handbook.

When stand-alone DAS MR is used under control of the stand-alone FORTRAN IV loader, the following logical unit assignments apply:

Name	Number	Default
PI	3	Card reader
LO	4	Line printer (-77)
BO	2	High-speed punch
SS	8	Magnetic tape (010)
GO	6	Dummy
PO	9	Magnetic tape (011)

The default assignments may be changed when loading.

To use DAS MR under control of the stand-alone FORTRAN IV loader, the loader is loaded by the Binary Load/Dump program using the HALT option (A register set to zero). The loader control words at locations 5, 6, and 7 are then set to 0210 before execution of the loader program. The loader then processes I/O specifications, loads DAS MR and its drivers, and starts execution of DAS MR.

Assembly continues through the source modules and halts when an end-of-file (externally identical to MOS) or a record with a / (slash) character in the first position in the record read. The slash terminator forces output of an end-of-file to the BO device.

Loading and Executing DAS MR

The DAS MR assembler is loaded and executed as follows:

1. Load the stand-alone loader using the binary load/dump program (BLD II). Set the A register to zero before loading to prevent execution of the stand-alone loader. At completion of loading, the execution address of the stand-alone loader will be in the X register, 13260.

2. Make the following modifications to core:

Location	New Contents
5	0210
6	0210
7	0210

3. Execute the stand-alone loader by setting the P register to the execution address determined in step 1 and pressing RUN.

4. When executed, the stand-alone loader will print "LN" on the Teletype. At this time, peripheral device assignments may be altered by entering the one-digit number of the old logical unit followed by the two-digit number of the substitute unit. DAS MR uses the following logical units:

Logical Unit Number	Logical Unit Name	Default Device Assignment
3	PI	Card reader
4	LO	Line printer
2	BO	Paper tape punch
6	GO	Dummy
8	SS	Magnetic tape* 00
9	PO	Magnetic tape** 10

*Device Address 010
**Device Address 011

As an example of device reassignment:

LN
300400201806900

Would reassign:

PI = Teletype Keyboard
LO = Teletype Printer
BO = Teletype Paper Tape Punch
SS = Teletype Keyboard
PO = Dummy

5. For a complete list of peripheral assignments, see the references. Following device reassignments, the stand-alone loader will print "IN" on the Teletype. At this time, the operator should ready the DAS MR object on the input device and respond by typing the proper designation on the Teletype:

P = Paper Tape Reader
T = Teletype Paper Tape Reader
0, 1, 2, 3 = Magnetic Tape Controller 0, 1, 2, or 3 respectively

To enable print out of a load map, the operator must type "M" immediately following the device designator. Following the typed characters, the operator must type a CR (carriage return) to initiate loading of the DAS MR object.

6. After DAS MR is loaded, peripheral devices for logical units 3, 4, 2, 6, 8, and 9 must be loaded from the Run-Time I/O tape. This is accomplished by placing the Run-Time I/O tape on the input device and repeating step 5.

7. After the Run-Time I/O is loaded, the I/O control program must be loaded from the Run-Time utility tape. This is accomplished by placing the Run-Time utility tape on the input device and repeating step 5.

8. When all externals have been satisfied, the loader will halt with the P register = 3. To execute DAS MR, the operator should press RUN.

   Upon execution, DAS MR will input source statements from logical unit (PI), output source for pass to logical unit (PO), input pass source from logical unit (SS), output binary object to logical unit (BO), and output listing to logical unit (LO).

   Source input to DAS MR terminates upon input of either an EOF or a source record containing a slash (/) as the first character. A slash record will cause an end-of-file to be output to the BO device.

Stand-Alone FORTRAN/DAS MR

Libraries

There are eight libraries for the stand-alone FORTRAN/DAS MR system.

1. Complex Math Functions (FORTRAN Coded)

   This library consists of programs collected, without modification, from the MOS. In order, they are:

$9E	$AC
CCOS	CMPLX
CSIN	$8K
CLOG	$8L
CEXP	$8M
CSQRT	$8N
CABS	$ZD
CONJG	AIMAG
$AK	$OC
$AL	REAL
$AM	$8F
$AN	$8S

2. Double Precision Math Functions (FORTRAN Coded)

   This library consists of programs collected, without modification, from the MOS. In order, they are:

$XE	DMINI
$YE	DSIGN
$ZE	$YK
DATAN2	$YL
DLOGIO	$YM
DMOD	$YN
DINT	DBLE
DABS	$XC
DMAXI	

3. Single Precision Math Functions (FORTRAN Coded)

   This library consists of programs collected, without modification, from the MOS. In order, they are:

TANH	SNGL
ATAN2	MAX0
ALOG10	MAX1
AMOD	MIN0
AINT	MIN1
AMAX0	MOD
AMAXI	INT
AMIN0	IDIM
AMIN	IFIX
DIM	$JC
FLOAT	

4. Double Precision Arithmetic (DAS Coded)

   This library consists of programs collected from the MOS. The only modifications made were the deleting or adding of control cards to define the object code for 16- or 18-bit machine. In order, they are:

DSINCOS
DATAN
DEXP
DLOG
IF
POLY
CHEB
DSQRT
$DFR
IDINT
DMULT
DDIVIDE
DADDSUB
DNORMAL
DLOADAC
DSTOREAC
RLOADAC
SINGLE
DOUBLE
DBLECOMP

5. Single Precision Arithmetic (DAS Coded): Hardware Multiply/Divide

This library consists of programs collected from the MOS. The only modifications made were the deleting or adding of control cards to define the object code for 16- or 18-bit machine. In order, they are:

$HE
$PE
$QE
ALOG
EXP
ATAN
SQRT-H
SINCOS
FMULDIV
FADDSUB
SEPMANTI
FNORMAL
XDDIV-H
XDMULT-H
XDADD
XDSUB
XECOMP
$FLOAT
$IFIX
IABS
ABS
ISIGN
SIGN
$HN-H
$HM-H
XMUL
XDIV
I$FA

6. Single Precision Arithmetic (DAS Coded): Software Multiply/Divide

This library consists of programs collected from the MOS. The only modifications made were the deleting or adding of control cards to define the object code for 16- or 18-bit machine. In order, they are:

$HE
$PE
$QE
ALOG
EXP
ATAN
SQRT-S
SINCOS
FMULDIV
FADDSUB
SEPMANTI
FNORMAL
XDDIV-S
XDMULT-S
XDADD
XDSUB
XDCOMP
$FLOAT
$IFIX
IABS
ABS
ISIGN
SIGN
$HN-S
$HM-S
XMUL
XDIV
I$FA

7. Run-Time I/O (DAS Coded)

This library consists of programs collected from the MOS. Control cards were added or deleted to define the object code for 16- or 18-bit machine.

Two additional modifications were made to the MOS routines: the Tele-type paper tape reader and punch drivers were merged into a single driver, $0H/$01; and the entry name of the driver for the line printer was changed to $0R. In order, they are:

FORTIO
$00
$04
$08
$0C
$0G
$0H/$01
$00
$0M
CRIE
$0Q($0R)
$0Q
$0P
$0S
CPAE
MT$0
MT$1
MT$2
MT$3
MTAE
KNT$
RDC$
WRT$
STR$
SWR$
BL$P
FCH$
TCK$
$TC01
$HC37
HCK$
DIM$
LAS$
IOA$
I00K
$BICD

8. Run-Time Utilities (DAS Coded)

This library, except for $BUF, consists of MOS programs, some modified and some not. In the following list, an asterisk (*) flags the programs which have more extensive modifications than selecting the 16- or 18-bit word size. In order, they are:

$DO
$CG
$3S
$SE
FORTUTIL
$EE
RSCB3*
RSCBIMTB*
$BUF

SECTION 16 - BINARY LOAD/DUMP PROGRAM

The **Varian 73 Binary Load/Dump Program (BLD II)** prepares the Varian 73 computer for the loading of object programs from a high-speed or Teletype paper tape reader. It also allows the punching of the binary contents of memory on paper tape in a reloadable format.

BLD II is loaded using the bootstrap loader routine, which specifies the input reader. Once loaded, BLD II automatically relocates itself into the upper part of the highest 4K memory increment, unless the operator specifies another 4K increment. BLD II also dynamically adapts itself to load object program tapes from the input device specified in the bootstrap loader routine, and performs a check-sum of object program records.

After BLD II has been loaded into memory, it need not be reloaded for the entering of subsequent object programs.

Initially, BLD II occupies addresses 007000 through 007755 of the first 8K memory increment, where it does not interfere with the bootstrap loader routine occupying addresses 007756 through 007776. Immediately after loading, BLD II relocates to occupy addresses 0x7400 through 0x7755, where x denotes the highest 8K of memory.

x =	Memory Increment
1	8K
3	16K
5	24K
7	32K

Entry to BLD II to load object program tapes is always 0x7600, and entry to punch binary object tapes of memory contents is 0x7404.

Loading the Bootstrap Routine

Under normal conditions the bootstrap loader routine would be loaded automatically as follows:

a. With the POWER switch in the ON position, place the computer in the run mode by pressing the STEP/RUN switch (RUN indicator blinking).

b. Insert the BLD II tape in the reader with the first binary frame at the read station.

c. Press the BOOT switch (RUN indicator is now on). This transfers the bootstrap program from the processor's control store to the computer memory and executes loading of the BLD II program.

For maintenance purposes it may be desirable to load the bootstrap routine manually.

Table 16-1 lists the manual bootstrap loader routines. If the high-speed paper tape reader is to be used for subsequent program loading, select the column headed High-Speed Reader Code; for the Teletype paper tape reader, select the column headed Teletype Reader Code.

To load the bootstrap loader routine:

a. Ensure that computer power is turned on and that the system is initialized.

b. Load the starting memory address of the bootstrap loader (007756) into the P register.

c. Press MEM switch momentarily.

d. Clear the console display (Press DISPL CLR).

e. Select the first bootstrap loader instruction from the appropriate column in table 16-1, and load it into the console display.

f. Press ENTER to load the display contents into the address specified by the P register, which is incremented by one after the instruction is loaded.

g. Clear the display (Press DISPL CLR).

h. Repeat steps d, e, f, and g for each bootstrap loader instruction.

To verify bootstrap loading:

a. Initialize the system by pressing (SYSTEM) RESET.

b. Load 007756 into the P register.

c. Select the memory for display by pressing MEM and press DISPL.

Table 16-1. Bootstrap Loader Routines

Address	High-Speed Reader Code	Teletype Reader Code	Symbolic Coding		
007756	102637	102601	READ	CIB	RDR
007757	004011	004011		ASLB	NBIT-7
007760	004041	004041		LRLB	1
007761	004446	004446		LLRL	6
007762	001020	001020		JBZ	SEL
007763	007772	007772		(Memory address)	
007764	055000	055000		STA	0,1
007765	001010	001010		JAZ	LHLT+1
007766	007000*	007000*		(Memory address)	
007767	005144	005144		IXR	
007770	005101	005101	ENTR	INCR	1
007771	100537	102601		EXC**	RDON
007772	101537	101201	SEL	SEN	IBFR,READ
007773	007756	007756		(Memory address)	
007774	001000	001000		JMP	*-2
007775	007772	007772		(Memory address)	

NOTE

The bootstrap loader routine is always loaded into the specified address of the first 8K memory increment, regardless of available memory.

* Replace this code with 007600 if the test executive of MAINTAIN III (refer to document number 98 A 9952 07x) is to be loaded and executed.

** CIB instruction if TTY bootstrap.

The contents of the memory addresses are displayed sequentially each time the DISPL switch is pressed. If an error is found, load the correct instruction code into memory. **Note that the P register error address is always the error address plus one.**

BLD II, and subsequent object programs, can now be loaded into memory

Loading the BLD II Program

After the bootstrap loader routine has been successfully loaded into memory:

a. Clear the instruction register.

b. Load 007770 into the P register.

c. Load 007000 into the X register.

d. Set the SENSE switch(es) for the desired program option (table 16-2).

e. Turn on the paper tape reader specified by the bootstrap loader routine.

f. Position the BLD II program tape in the reader with the first data frame after the position-8-only punches (figure 16-1) under the high-speed reader head or under the reading station of the Teletype reader.

g. To load tape, press RUN, then START. Loading is complete when the computer changes to step mode.

Table 16-2. BLD II SENSE Switch Options

SENSE Switch	When Set =
1	Allows selection of any 8K memory increment in which BLD II is to operate, or specification of a nonstandard device address for the high-speed paper tape punch. After BLD II is loaded, the computer halts with 07014 in the P register. To specify a 8K memory increment, load one of the following in the A register: **A Register — Memory Increment** 000001 — First 8K 000003 — Second 8K 000005 — Third 8K 000007 — Fourth 8K The standard high-speed paper tape punch device address is 037. To specify a nonstandard device address, load it into the B register.

Table 16-2. BLD II SENSE Switch Options *(continued)*

SENSE Switch	When Set =
	Result: Pressing START initiates the relocation at BLD II from the first 8K memory increment and implements the punch address. The computer halts with zeros in the A, B, and X registers and 0x7600 in the P register, where X = the specified increment as described above. Object program tapes can then be loaded.
2	Adjusts the program for Teletype paper tape punch output. (For use when input is from high-speed reader, but a high-speed punch is not available.) **Result:** BLD II and the object program can be loaded and executed without further operator intervention.
3	Allows splicing an object program to the BLD II program tape. **NOTE** If no SENSE switches are set, the BLD II program is loaded and relocates automatically to the highest 8K memory increment. The computer then halts with the entry address for reading object program tapes in the P register (0x7600) and zeros in the A, B, and X registers.

If SENSE switch 1 was set:

a. Reset SENSE switch 1.

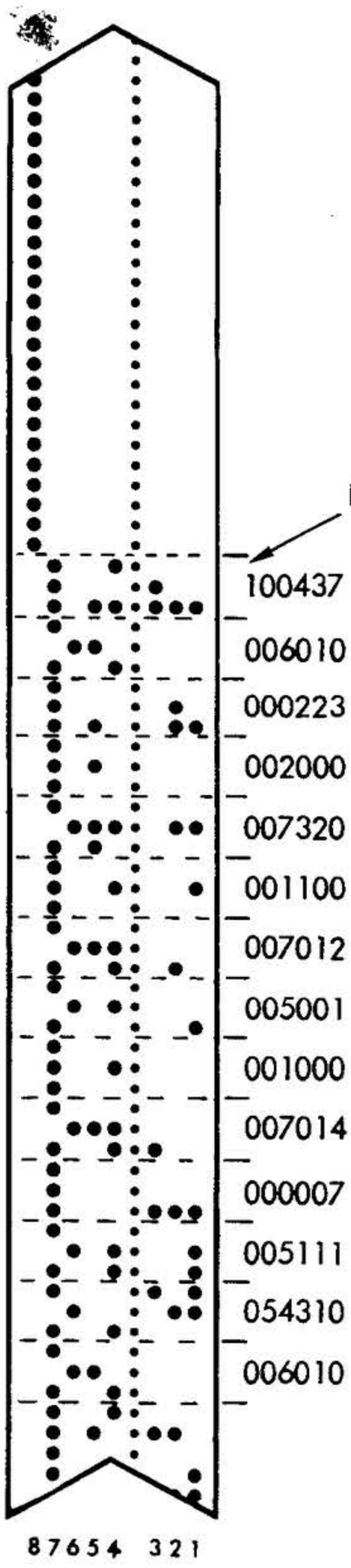
b. Clear the A register.

c. Load the appropriate values, as defined in table 16-2, in the A and/or B registers.

d. Press START.

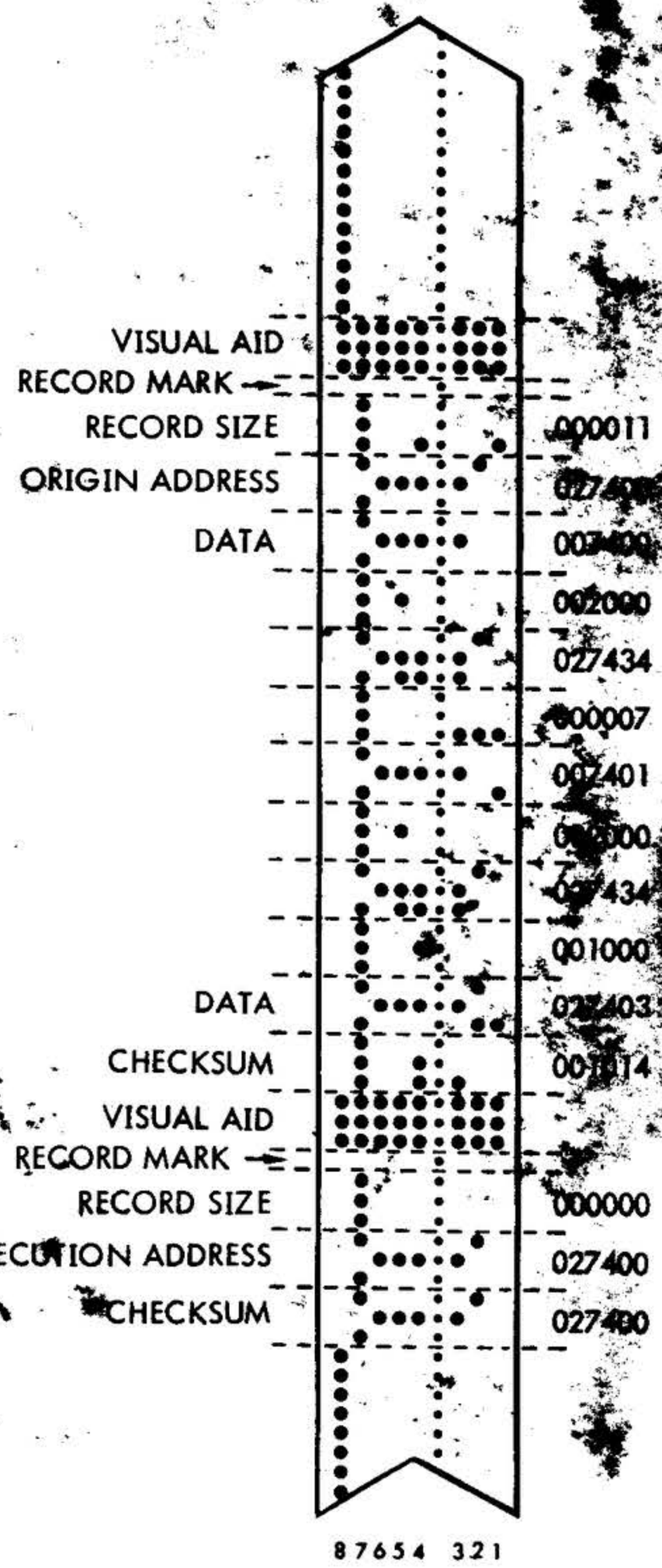
When BLD II loading is complete, the computer halts with 0x7600 in the P register unless SENSE switch 3 was set (table 16-2), in which case the computer implements loading and execution of the spliced object program.

Remove the BLD II program tape from the reader after loading, and reset SENSE switch 2, if applicable.



VTI1-1188A

Figure 16-1. BLD II Tape Format (Bootstrap-Loadable)



VTI1-1189

Figure 16-2. Object Program Tape Format

Loading an Object Program

Object programs can be loaded from the bootstrap-routine-specified device immediately after BLD II. For all subsequent loadings, make sure that the P register is set to 0x7600.

Verification

To ensure that an object program tape contains no errors before it is loaded into memory, BLD II has a check-sum error-checking option. To use this option:

a. Turn on the bootstrap-routine-specified reader.

b. Position the object program tape in the reader with leader at the reading station (figure 16-2).

c. Load minus value (0100000) into the A register.

d. Clear the instruction register.

e. Set RUN indicator on and press START.

No errors are indicated by the computer halting with:

P register = 0x7600
A register = 0100000
B register = 000000
X register = execution address

If a check-sum error occurs, the computer halts with:

P register = 0x7600
A register = 0100000
B register = 0177777
X register = Address of last record.

To retry a check-sum error record, reposition the object program tape at the previous visual aid and press START. If a check-sum error is again read, visually check each character in the record for an error in punching or damaged tape.

Load Program and Halt

To load the object program and halt before execution:

a. Turn on the reader and position the tape in the reading station.

b. Clear the A, B, X, and instruction registers.

c. Load 0x7600 into the P register.

d. Set RUN indicator on and press START.

Correct loading is indicated when the computer halts with:

P register = 0x7600
A register = 000000
B register = 000000
X register = execution address

A check-sum error is indicated by the conditions described for object program tape verification described previously.

Load Program and Execute

Programs can be loaded and immediately executed using the steps described above for the load-and-halt option, except in step b load 000001 (or any positive number) in the A register.

Punching Program Tapes

The BLD II program adapts to the input reader and the output punch devices by interrogating the bootstrap loader routine. Setting SENSE switch 2 (table 16-2) prior to loading BLD II program adjusts the program for Teletype punch output regardless of the bootstrap-routine-specified devices.

To punch reloadable object program tapes after the programs have been loaded into memory, turn on the punch and:

a. Load the beginning address of the area to be punched into the A register.

b. Load the final address to be punched into the B register.

c. Load the first instruction to be executed at load time into the X register.

OR

if noncontiguous memory areas are to be punched, load minus one (177777) into the X register.

d. Load 0x7404 (entry address to BLD II to punch object tapes) into the P-register.

e. Clear the instruction register.

f. Press (system) RESET, set RUN indicator on and press START.

The program punches the object tape and the computer halts with all registers unaltered.

If noncontiguous areas are to be punched, perform steps a through f. Prior to punching the last area, load the first instruction to be executed at load time into the X register.

Punching Memory Contents

To punch a tape of the binary memory contents on the high-speed paper tape punch, SENSE switch 2 must not be set when BLD II is loaded. To punch a tape from memory on the Teletype punch, SENSE switch 2 must be set (if the input reader is a high-speed paper tape reader).

The operator can specify that tapes be punched in binary format for reloading using the BLD II, or that the BLD II program be punched in bootstrap-loadable format.

To punch a tape in binary format, use the procedures described above for punching program tapes.

To punch a bootstrap-loadable tape of BLD II itself:

a. Load 0x7400 into the P register.

b. Clear the A and B registers.

c. Load a nonzero value into the X register.

d. Press (system) RESET, set RUN indicator on and press START.

SECTION 17 - AID II DEBUGGING PROGRAM

The **Varian 73 AID II Debugging Program** is supplied for use with all Varian 73 systems. AID II provides software to facilitate on-line program checkout and correction. By entering AID II commands on the Teletype keyboard, the operator can:

a. Display and alter the contents of registers and any memory address or group (block) of addresses.

b. Transfer (trap) into or out of selected blocks of memory and search for specific conditions.

c. Load, monitor, and alter any program.

As an added feature, data can also be transferred (dumped) from memory to magnetic tape, punched out on paper tape, or printed on the Teletype printer. Object programs can thus be converted from one media to another, simply and directly.

AID II is loaded into computer memory using the binary load/dump program (BLD II, section 16. Once loaded, AID II resides in memory addresses 0x6000 through 0x7377, where x denotes the highest available 8K memory increment.

x =	Memory Increment
1	8K
3	16K
5	24K
7	32K

The programmer is responsible for ensuring that a program to be debugged does not interfere with those areas of memory containing BLD II and AID II.

Loading AID II

To load AID II into memory:

a. Ensure that the bootstrap loader routine and BLD II (section 16) are correctly loaded.

b. Turn on the reader used to load BLD II and position the AID II program tape with leader at the reading station.

c. Clear the B, X, and instruction registers, and load 000001 into the A register.

d. Load 0x7600 into the P register (i.e., the BLD II entry address for loading program tapes; refer to section 16 for the definition of x).

e. Set the RUN indicator on and press START.

Loading is complete when the program outputs a carriage return (CR) and line feed (LF) and rings the Teletype bell.

Programs to be debugged can be loaded either before or after AID II loading.

Register and Memory Modification

With AID II and the program to be debugged entered, the computer in run mode, and the Teletype operating on-line, the Teletype keyboard entries summarized in table 17-1 produce the indicated results.

The pseudoregisters referred to in the following descriptions denote software

buffers that duplicate the actual contents of the computers's operation registers. A command to change register contents, in effect, changes the specified pseudoregister contents, which are then transferred to the corresponding operation register.

Table 17-1. AID II Register/Memory Modification Commands

Command	Operation
A **B** **X**	Displays (prints) the contents of the indicated pseudo register on the Teletype printer. To change the contents, type the desired octal number and a period; otherwise, type only a period.
Cx	Displays (prints) the contents of memory address x on the Teletype printer. To change the contents, type the desired octal number, followed by a period to execute the command or by a comma to request display of the next sequential address contents. Otherwise, type only a period.
Gx.	Loads the contents of the pseudo registers into the respective A, B, and X registers and starts program execution at address x.
Ix,y,z,.	Stores the value of z in all memory addresses starting at address x and ending at address y.
Sx,y,z,m.	Searches through memory starting at address x and ending at address y for the value of z masked by the value of m. A masked-search compares the value of z with each bit corresponding to a one in the m value. Each time the values compare, the address and value are printed on the Teletype printer. If an N is typed instead of a mask value, the program searches for the negative value of z. Omission of m assumes an all-ones mask.
Ty,x.	Transfers execution of an operational program to address x when the program reaches the instruction in address y. This trapping feature permits interrupting a program sequence without internal patching. The program also displays the transfer address and the contents of the A, B, and X pseudo registers, respectively. Note: If location y is not reached, two locations of the program will be destroyed.

Table 17-1. AID II Register/Memory Modification Commands *(continued)*

Command	Operation
Ty,.	Continues trap from last break point.
Vx.	Displays the contents of memory on the Teletype printer beginning at address x, continuing until a RUBOUT character is typed. The display (dump) is printed in columns: the left column is the octal base address, and the contents of eight memory addresses, in ascending order, appear in the next eight columns. The first number in succeeding lines indicates the base address for the next eight memory address contents.

Usage Examples

NOTE

In the following examples, operator inputs are represented in bold type. Other entries are program responses output to the Teletype printer.

Display the contents of a pseudo register:

```
A    142340    .
B    001000    .
X    006003    .
```

Display and change the contents of a pseudo register:

```
A    010454    10406.
B    006016    10406.
X    007413    10406.
```

Display the contents of memory address 002050:

```
C2050        =    102401    .
```

Display and change the contents of memory address 002050, then display the next two addresses:

```
C2050        =    102401    103402,
( 002051 )   =    000067    ,
( 002052 )   =    177777    .
```

Display memory contents starting at address 006000:

```
V6000.
( 006000 )   010454   002000   ...
( 006010 )   005145   004543   ...
( 006020 )   005041   001000   ...
( 006030 )   006217   001000
```

NOTE

When displaying memory contents, eight columns of data actually follow the base address in the first column. Space limitations prohibit an actual representation herein.

(Display terminated by entering RUBOUT.)

PROGRAM

n beginning at address

```
T204,100.
(=000204 )  142340  002000  010405
```

n memory addresses
0210:

```
7777,.
.
```

ldresses 000200 through
ent of 0106213 masked
display addresses that

```
06213,177777.
  = 106213
  = 106213
```

address 000204; start
ddress 000100; and dis-
ess and the A, B, and X
if the trap is reached. If
iginal contents into both

AID II DEBUGGING

Execute the progra
000500:

```
G500.
```

Store 0177777
000200 through 00

```
I200,210,17
I200,210,-1
```

Search memory ac
000240 for a cont
by 0177777 and
compare:

```
S200,240,1
( 000220 )
( 000235 )
```

Trap to memory
execution from ac
play the trap addr
register contents
not, reload the or
trap locations.

Paper Tape Handling

The Teletype paper tape reader and punch can be controlled through AID II to read object program tapes into, and punch program tapes from, computer memory.

With AID II entered, the computer in run mode, and the Teletype and its paper tape system operational, the Teletype keyboard entries summarized in table 17-2 produce the indicated results.

Magnetic Tape Handling

Data can be manipulated from and to magnetic tape through AID II commands.

With AID II entered, the computer in run mode, the Teletype keyboard on-line, and the selected magnetic tape unit operational, the Teletype keyboard entries summarized in table 17-3 produce the indicated results.

Table 17-2. AID II Paper Tape Commands

Command	Operation
Dx,y,z,.	Punches a program tape from the contents of address x through address y, specifying execution address z.
Lm.	Reads an object program paper tape into memory. If the value of m is 1 and no check-sum errors are encountered, the program is executed. If the value of m is 0 and no check-sum errors are encountered, the contents of the A, B, and X registers, respectively, are output on the Teletype printer: A register = 000000, B register = 000000, and X register = execution address.

SECTION 18 - SOURCE PROGRAM EDITOR

Varian's 73 Source Program Editor (EDIT) allows the Varian 73 computer programmer to create and modify symbolic source programs (section 15) on paper tape. Source programs can be loaded directly into computer memory from an on-line Teletype keyboard, listed with identifying line numbers on the Teletype printer, and modified using EDIT commands input from the Teletype keyboard.

Source programs already formatted on paper tape can be loaded into memory, listed, modified with EDIT generating a paper tape of the modified program ready for assembly (section 16).

An added feature of EDIT is its ability to search through the source program and point to a specific character or group of characters, as well as entire lines and groups of lines.

EDIT has two modes of operation: command and text. In command mode, EDIT accepts inputs from the Teletype keyboard specifying the EDIT function and, optionally, line numbers and searching parameters. In text mode, characters typed on the Teletype keyboard or read from paper tape are stored in a text buffer for subsequent manipulation and/or output. The text buffer represents available memory, i.e., those memory addresses not occupied by the bootstrap loader routine, the binary load/dump program (BLD II, section 16, and the EDIT program routines.

In text mode, EDIT runs without an operating system. Both MOS and VORTEX include editing functions which are an alternative in their environments.

EDIT operates in the minimum configuration of a Varian 73 system (8K to 32K of memory) and 33/35 ASR Teletype. However, EDIT determines the size of memory and uses of all available memory for the editing buffer; only the binary loader at the top of memory is served. Use of the high-speed paper tape reader and/or punch for input/output is optional.

Loading EDIT

To load the EDIT program into memory:

a. Ensure that the bootstrap loader routine and BLD II are correctly loaded (section 16).

b. Turn on the reader used to load BLD II and position the EDIT program tape with leader at the reading station.

c. Clear the B, X, and instruction registers.

d. Load 000001 into the A register.

e. Load 0x7600 into the P register (i.e., the BLD II entry address for loading object program tapes; refer to section 15 for the definition of x).

f. Set RUN indicator on and press RUN or START.

Loading is complete when the EDIT program outputs, on the Teletype printer, the message:

```
SOURCE PAPER TAPE PROGRAM

INPUT DEVICE (H OR T)
```

If the high-speed paper tape system is to be used for text input to EDIT, type H on the Teletype keyboard, and type T if the Teletype is the input device. The program then outputs

```
OUTPUT DEVICE (H OR T)
```

Respond as described above for defining the input device. EDIT dynamically adapts to use the specified equipment and enters the command mode, outputting a carriage return (CR) and line feed (LF), followed by an asterisk (*), to the Teletype printer.

Once entered, EDIT can be restarted at any time by clearing all registers and pressing RUN or START.

NOTE

To change input and output devices from those initially specified, EDIT must be re-loaded using the procedures described above.

EDIT Commands

With EDIT loaded, the computer in run mode, and the Teletype operating on-line, the Teletype keyboard entries summarized in table 18-1 produce the indicated results. Pressing the RETURN key terminates and executes all EDIT commands.

Table 18-1. EDIT Commands

Command	Operation
A	Enter text mode and add the following text input from the Teletype keyboard to the contents of the text buffer.
nC	Delete the line specified by n, and replace it with new text.
m,nC	Delete and replace lines m through n.
nD	Delete line n.
m,nD	Delete lines m through n.
F **xxxx**	Search the entire contents of the text buffer for character string xxxx (maximum number of characters, 72). Output the line on which it appears. If the string is not found, return to command mode, and output CR, LF, and *
nF **xxxx**	Go to line n and search it and succeeding lines for character string xxxx (see above).
G	List (output on the Teletype printer) the next sequential line whose first character is alphabetic.

Table 18-1. EDIT Commands *(continued)*

Command	Operation
nG	Go to line n and list the next line whose first character is alphabetic.
I	Insert the following text before the first line in the text buffer.
nI	Insert the following text before line n.
K	Delete the entire contents of the text buffer.
L	List the entire contents of the text buffer, assigning sequential line numbers (decimal), on the Teletype printer.
nL	List line n.
m,nL	List lines m through n.
P	Punch the contents of the text buffer on paper tape using the output device specified at edit loading time.
nP	Punch line n.
m,nP	Punch lines m through n.
R	Read (append) the following text input from the device specified at EDIT loading time to the contents of the text buffer.
S	Search the contents of the text buffer for the character input after RETURN. Output sequential text lines on the Teletype printer until the line in which the character appears is printed. If the character is not found, return to command mode, and output CR, LF, and *.
nS	Go to line n and search for the character input after RETURN (see above).
m,nS	Search lines m through n for the character input after RETURN (see above).
T	Punch approximately 20 inches of leader/trailer on paper tape using the output device specified at EDIT loading time.

Table 18-1. EDIT Commands *(continued)*

NOTES

Line numbers when specified in EDIT commands are decimal integers derived from the output of a listing command. The value of n must be greater than that of m.

Execution of **all** EDIT commands begins when the RETURN key is pressed.

Table 18-2 lists EDIT functions that are controlled by the use of Teletype special-purpose keys. Note that their use differs in the two modes of operation.

Table 18-2. Teletype Key EDIT Functions

Teletype Key	Command Mode	Test Mode
RETURN	Execute the instruction	Load the input line into the text buffer
←	Illegal	Delete one character to the left and output ←
RUBOUT	Cancel the instruction	Delete all the line to the left and output \
CTRL and C (simultaneously)	Remain in instruction mode and output an asterisk (*)	Return to instruction mode and output an asterisk (*)
. (period)	Current line number (used alone or with the minus sign and number, e.g., 1. −8 refers to the eighth line preceding the current line)	Legal text character
/ (slash)	Number of the last line in the text buffer	Legal text character

Table 18-2. Teletype Key EDIT Functions *(continued)*

Teletype Key	Command Mode	Test Mode
=	Used with . and / to obtain their values	Legal text character
ESCAPE (ESC)*	List the next line	Ignored
CTRL and TAB (simultaneously)	Illegal	Interpreted as seven spaces on the Teletype printer output

* On the model 35 Teletype, simultaneously press SHIFT, CTRL, and K.

Usage Example

To illustrate the use of EDIT commands and Teletype key functions, assume we wish to search line 20 for the character A and replace it with the character X. Note that the Teletype keys are shown enclosed in parentheses where they are applicable and that the simultaneous pressing of two or more keys is illustrated as follows: (SHIFT)(CTRL)(K).

a. To ensure that EDIT is in command mode, type

```
(CTRL)(C)
```

b. EDIT responds with a CR, LF, and *. Type

```
20S(RETURN)
```

c. EDIT enters a delay loop and waits for input of the character for which it is to search. Type

```
A(RETURN)
```

d. EDIT goes to line 20 and types it until an A is found:

```
XYZ   LDA
```

then waits for input. Type

```
-X(RETURN)
```

Other editing options available for use in step d are:

a. To delete the line to the left, type RUBOUT.

b. To delete the line to the right, type RETURN.

c. To delete the entire line, type the appropriate deletion command (table 18-1).

d. To delete characters from right to left, type ← once for each character.

Error Messages

EDIT checks all commands input to it for valid parameters and correct formatting. When an error is detected, EDIT:

a. Types a question mark on the Teletype printer.

b. Issues a CR and LF.

c. Types an asterisk.

d. Waits for a valid command.

The following conditions are recognized as errors:

a. Incorrect response to the I/O device queries at loading time.

b. A nonexistent command code.

c. Commands terminated with any character other than RETURN.

d. A starting line number that is greater than an ending line number.

e. Transposition of command parameters.

f. Specifying a line number whose value is greater than the last line in the buffer.

g. A deletion command that does not specify a line number.

h. Pressing the ESCAPE (ESC) key to list the next line in the buffer when the buffer is empty.

When text being loaded into the text buffer from the keyboard exceeds the capacity of the buffer, EDIT outputs the message

```
BUF FULL
```

and enters the command mode. The last line entered at the keyboard is lost. When text being read from tape (R command) exceeds the alloted space, reading stops and the program enters the command mode. In this case no input data is lost (tape reading stops with approximately 1600 characters of text buffer remaining to allow for editing). To save the buffer contents and continue processing:

a. Type a punch (**P**) command (table 18-1) and RETURN.

b. After punching is complete, clear the text buffer using the **K** command.

The following options are also available:

a. List, modify, and punch the buffer contents before clearing the text buffer.

b. Abort the current source program edit and continue processing with a new program.

SECTION 19 - VORTEX/VORTEX II

The **Varian Omnitask Real-Time EXecutive** (**VORTEX**) is a modular software operating system for controlling, scheduling, and monitoring tasks in a real-time multiprogramming environment. VORTEX also provides for background operations such as compilation, assembly, debugging, or execution of tasks not associated with the real-time functions of the system.

The basic features of VORTEX comprise:

- Real-time input/output processing
- Provision for directly connected interrupts
- Interrupt processing
- Multiprogramming of real-time and background tasks
- Priority task scheduling (clock time or interrupt)
- Load and go
- Centralized and device-independent input/output
- Operator communications
- Batch-processing job-control language
- Program overlays
- Background programming aids: FORTRAN compiler, DAS MR assembler, load-module generator, library updating, debugging, and source editor
- Use of background area when required by foreground tasks
- Disc/drum directories and references
- System generator

A rotating-memory device (either disc or drum) serves as storage for the VORTEX operating system components, enabling real-time operations and a multiprogramming environment for solving real-time and nonreal-time problems. Real-time processing is implemented by hardware interrupt controls and software task scheduling. Tasks are scheduled for execution by operator requests, other tasks, device interrupts, or the completion of time intervals.

Background processing (nonreal-time) operations, such as FORTRAN compilations or DAS MR assemblies, are under the control of the job-control processor, itself a VORTEX background task. These background processing operations are performed simultaneously with the real-time foreground tasks until execution of the former is suspended, either by an interrupt or a scheduled task.

All VORTEX tasks are scheduled, activated, and executed by the real-time executive component on a priority basis. Thus, in the VORTEX operating system, each task has a level of priority that determines what will be executed first when two or more tasks come up for execution simultaneously.

The job-control processor component of the VORTEX system manages requests for the scheduling of background tasks.

Upon completion of a task, control returns to the real-time executive. In the case of a background task, the real-time executive schedules the job-control processor to determine if there are any further background tasks for execution.

During its execution, any foreground task can use any real-time executive service, i.e., operations that the task itself cannot perform including those involving linkages with other tasks.

VORTEX requires the following minimum hardware configuration:

a. Varian 73 processor with 8K read/write main memory

b. Direct memory access (DMA)

c. 33/35 ASR Teletype on a priority interrupt module (PIM)

d. Real-time clock (RTC)

e. Memory protection (MP)*

f. Power failure/restart (PF/R)

g. PIM

h. Rotating-memory device on a PIM with either a buffer interlace controller (BIC) or priority memory access (PMA)

i. One of the following on a PIM:
   (1) Card reader with a BIC
   (2) Paper tape system or a paper tape reader
   (3) Magnetic tape unit with a BIC

The system supports and is enhanced by the following optional hardware items:

a. Additional main memory (up to 32K)** and/or rotating memory

b. Automatic bootstrap loader (ABL)

c. Card reader, if one is not included in the minimum system

d. Card punch with BIC and PIM

e. Line printer with BIC and PIM

f. Paper tape punch, if one is not included in the minimum system.

The VORTEX system is described in detail in the VORTEX Reference Manual (document number 98 A 9952 10x). The VORTEX II system is described in the VORTEX II Reference Manual (document number 98 A 9952 24x).

Note:

* VORTEX II uses the memory map feature for memory protection.

** VORTEX II supports additional main memory (up to 256K) and/or rotating memory.

SECTION 20 - MASTER OPERATING SYSTEM

The Varian 73 Master Operating System (MOS) is a batch-processing operating system for Varian 73 system computers. It is a complete integrated software package that operates in a wide range of hardware configurations.

MOS is modular, thus facilitating expansion and making optimum use of memory since only those portions of the system required for a specific operation need be loaded.

Features of MOS include:

- Minimum operator intervention required
- Single tape, drum, or disc as secondary storage device
- Extensive job control language (22 directives)
- Multisource input during loading
- Debugging aids
- File maintenance and editing programs
- Extensive status- and error-reporting

MOS requires, in addition to the processor, a minimum of 8K of memory, a 33/35 ASR Teletype, and either a rotating memory (drum or disc) unit on a buffer interface controller (BIC) or a magnetic tape unit. MOS supports and is enhanced by the addition of the high-speed paper tape and/or punch, line printer, the hardware multiply/divide and extended addressing feature, card reader and/or punch, and additional memory increments.

MOS is described in detail in the Varian Master Operating System Manual (document number 98 A 9952 09x).

MOS is divided into resident (in memory) and nonresident partitions. Figure 20-1 shows the relationship.

The resident partition comprises the resident monitor, absolute loader, I/O assignment tables, system flags and parameters, and dump routine.

The nonresident partition comprises the control programs, support programs, and language processors, all of which need be in memory only when required for processing.

The **control programs** are:

a. Executive

b. System loader

c. I/O control

The *executive program* directs execution of the user's programs. It interprets the system directives and either executes the instructions directly, or calls and initiates the appropriate software. Upon completion of the sequence, the executive resumes control and goes to the next directive.

The *system loader* can load program components unconditionally and/or selec-

tively by program name. It accepts only relocatable object text and permits literal addressing and linking of external programs. When the system loader successfully completes an operation, it returns control to the resident monitor for program execution.

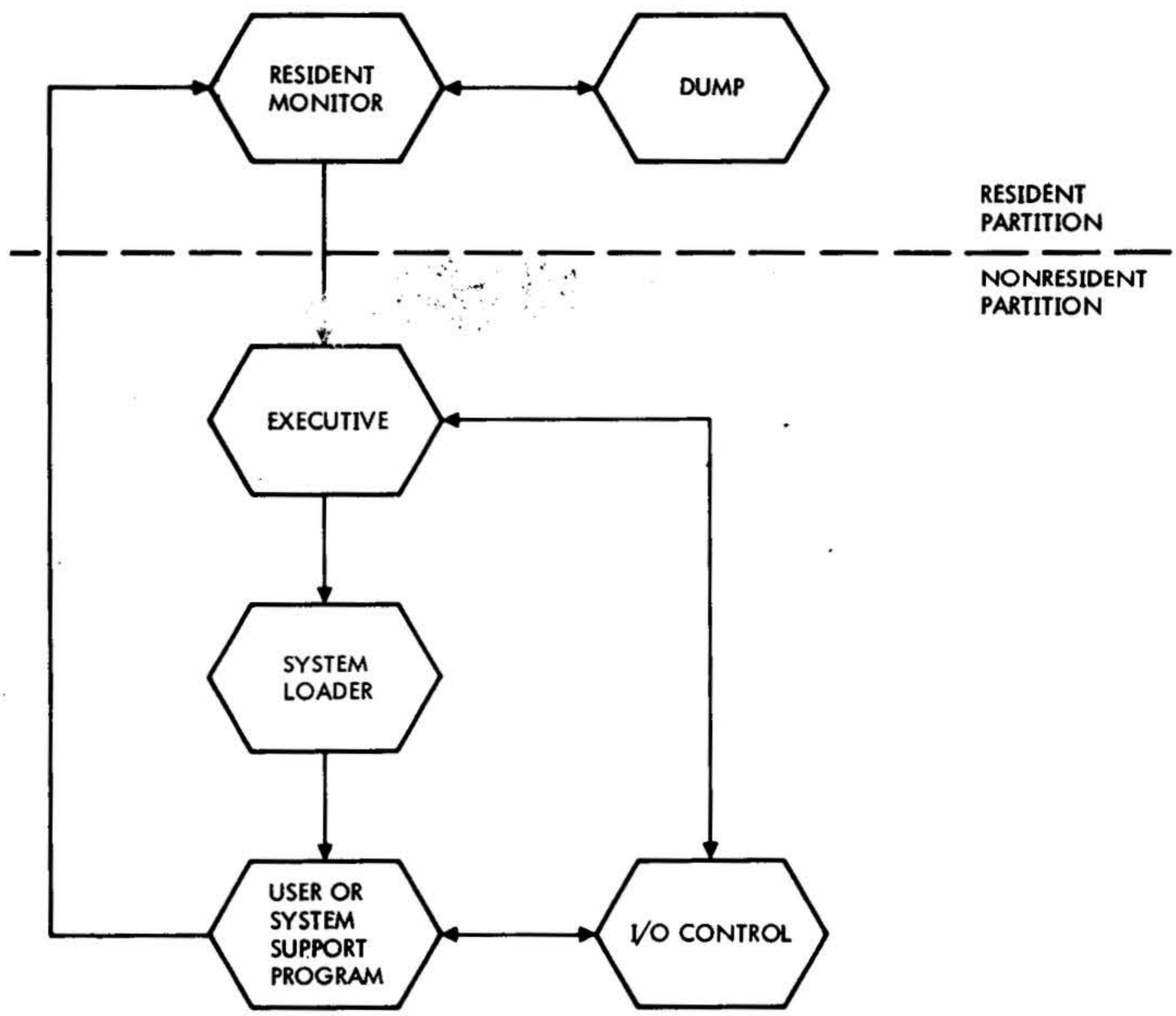
The *I/O control program* processes all I/O requests. I/O control is flexible and device-independent. This program assigns I/O functions to peripherals by logical unit designations instead of addressing the devices as hardware units. Up to 225 logical units can be so assigned, and they can be reassigned at any time. Thus, different peripherals can be substituted for I/O operations without reassembling the program.

MOS **support programs** include:

a. The *debugging program* -- aids the programmer in finding and correcting program errors. Debugging commands permit examining and/or changing a program, in addition to running part or all of it.

b. The *concordance program* -- analyzes the symbols of a DAS MR assembler program, indicating where they are defined and referenced. Any source program in the DAS language can be so analyzed.

c. The *file editing program*, or source editor -- provides for the creation, duplication, and correction of source files, such as symbolic DAS and FORTRAN IV program statements.



VTI1-0926

Figure 20-1. MOS Partitions and Flow

d. The *file maintenance program* -- edits the MOS mathematical and support subroutine library. Program subroutines can be added to or deleted from the library.

e. The *system preparation program* -- is a stand-alone support program. It creates a system file on a magnetic tape or rotating memory unit, tailored to the hardware and software requirements of the MOS installation.

f. The *mathematical and support library*.

The MOS **language processors** are: the DAS MR assembler, and the FORTRAN IV compiler (requires an additional memory increment). Under MOS supervision, the processors automatically process source statements to generate relocatable and compatible object programs.

SECTION 21 - BEST

The **Basic Executive Scheduler and Timekeeper (BEST)** is a real-time monitor for Varian 73 system computers that allows a variable number of core resident routines to operate concurrently within a relative priority system.

Scheduling is based on time of day so that a program can be scheduled to be run at a specific time, after a specific time interval, or at the next opportunity.

The BEST system makes use of periodic interrupts from the real-time clock at an interval determined by the user's requirements. These interrupts trigger a sequential scan of the packet table. Each packet contains the information required to put the routine referenced by the packet into execution; including the time requested, the overflow indicator, the address requested (or interrupted at), and the A, B, and X register contents. This information may be dynamically placed into the packets by a monitor call or a system interrupt; or it may be determined initially at assembly time.

When a clock interrupt occurs, the system searches through the packet table, comparing the scheduled run time of each routine with the system time at the last periodic interrupt. Once the system has determined that a given packet is due to run, it sets that routine into execution and remembers its position within the packet table.

If control is returned before the next periodic interrupt has occurred, the scan will continue from the next packet in the table. If the periodic interrupt occurs before control is passed to the system, the data from the interrupted routine is placed into its packet, and the scan is initiated from the top of the packet table.

BEST provides the following secondary functions: DEBUG, Arithmetic Package, and Data Manipulation Package. BEST serves as a simple and convenient framework for the user's real-time tasks, while requiring only minimum hardware for its support. BEST will run in any 73 system with 8K or more of memory.

SECTION 22 - FORTRAN IV

The **Varian 73 FORTRAN IV Programming System** permits simple solutions of complex mathematical problems and is especially useful for scientific and engineering applications. The name FORTRAN is derived from the primary use of the language: **formula translation.**

Varian's FORTRAN IV system comprises a programming language, a library of subprograms, a compiler, and runtime program. It is available in both stand-alone, master operating system (MOS) and VORTEX configurations. The FORTRAN IV language is compatible with, and encompasses the capabilities of, American National Standards Institute (ANSI) FORTRAN, including its mathematical subroutine provisions.

A major feature of FORTRAN IV is its simplicity. Problems can be stated in simple English words and mathematical terms. Thus, persons having little computer organization experience and minimum programming skills can write effective programs after only a few hours of training.

The FORTRAN IV language consists of a series of source statements divided into physical sections called lines and coded to a precise grammatical format. The compiler analyzes the source program statements and transforms them into an object program suitable for execution on the Varian 73 computers. One-pass processing provides convenient, efficient compilations, and the stand-alone system operates in only 8K of memory. The MOS version requires 12K. FORTRAN IV is also an integral part of VORTEX.

The FORTRAN IV system is described in detail in the Varian FORTRAN IV Reference Manual (document number 98 A 9902 03x).

The writing of effective FORTRAN IV programs is facilitated by the following features of the language:

a. A *scale factor* allows internal and external representations of data to be modified during conversion.

b. *Variable and array attributes* can be explicitly named by statements specifying:

   (1) The number of words assigned to an item.

   (2) A variable as integer, real, double-precision, complex, or logical.

   (3) Array dimensions.

   (4) Data initialization values for variables.

c. *An array* can be preset to specific values.

d. *Subprogram array dimensions* can be specified as variables, and absolutes substituted when the subprogram is called.

e. *An array* can have one, two, or three dimensions.

f. *A variable name* can contain up to six characters.

g. *Mathematical function subprograms* return results via an argument list.

There are two classes of FORTRAN source statements: executable and nonexecutable. Executable statements specify program action. Nonexecutable statements describe program usage, operand characteristics, editing information, statement functions, and data arrangement. Functional groupings of these statements include data specification, arithmetic/logical expressions and assignments, input/output, subprogram definition, and program control.

The one-pass FORTRAN IV compiler translates source programs to relocatable machine-language object programs. Error diagnostics, source listings, and object listings are generated. Input/output and listing options can be selected for each program to be processed.

The FORTRAN IV relocatable loader enters into memory the object programs produced by the compiler and FORTRAN-compatible subprograms produced by the DAS 8A assembler (section 15). Object program media can be either paper or magnetic tape. Memory maps and error diagnostics are output on the Teletype printer.

Standard FORTRAN IV library subprograms include:

a. Runtime input/output and utility.

b. Single- and double-precision and complex mathematical functions.

c. Single- and double-precision mathematical library.

SECTION 23 - RPG IV (Report Program Generator)

The **Varian 73 RPG IV (Report Program Generator) language** is an advanced version of the widely used RPG commercial and general data-processing systems. RPG IV permits the concise coding of powerful programs, simply and efficiently. Thus, users with background other than data-processing can use RPG IV problem-solving techniques without extensive training or practice.

RPG IV programs are far more concise than equivalent programs written in the COBOL language. In typical instances, only one-fourth to one-third of the program steps are required. This results in more efficient processing and reduces the amount of memory required to store the program.

Varian's RPG IV improves on basic RPG in that it incorporates many automatic features and powerful procedural statements. Each RPG IV statement is written free-form, thus simplifying the programmer's task.

RPG IV is particularly adapted to processing data for the output of reports, but has many other applications as well.

The RPG IV system is offered in a stand-alone version, which operates with a Varian 73 processor, 8K (minimum) memory, card reader and punch, and line printer. RPG IV can also be operated under the control of the Varian Master Operating System (MOS, document number 98 A 9952 09x) or VORTEX (document number 98 A 9952 10x).

RPG IV is described in detail in the Varian RPG IV System User's Manual (document number 98 A 9947 03x).

RPG IV programs comprise two sequences of statements. First, there is a sequence of data-defining statements delineating the structures and formats of the data to be processed. This is followed by a sequence of procedural statements; these statements process the data through the structures defined in the first sequence. These two sequences of statements handle tables and records, update files, produce reports, and can deal with any other business-oriented applications.

Four types of **data-defining statements** provide a definition of the data structures to be used by the program. These data structures are records and tables. Records hold intermediate results and data being input from or output to files. Tables contain related, repetitive data items.

Both records and tables are divided into fields. Fields are the elementary variables of any RPG IV program. The computations performed by the program, its logic, and its final output are based on the manipulation of fields and their contents.

The functions of data-defining statements are:

a. A **record statement** -- identifies a record and specifies the conditions under which this record is manipulated.

b. A **record field statement** -- identifies and defines all of the fields in the record. All record field statements pertaining to a given record immediately follow the record statement for that record.

c. A **table statement** -- identifies a table and specifies its size.

d. A **table field statement** -- identifies and defines all of the fields in the entries in a table. All table field statements pertaining to a given table immediately follow the table statement for that table. Each entry in a given table has the same field structure as any other entry in that table.

Procedural statements follow the data-defining statements. They direct the execution of the program as it processes the data previously defined by the data-defining statements.

Procedural statements are executed in the order of their appearance in the program unless a specified condition is not met, or unless the program is directed to another statement by the statement in process.

Each RPG IV program statement can be numbered so that the program has access to it as required. Numbering is optional for statements that do not require other than sequential access.

Another important feature of RPG IV is that comment lines can be included to clarify a program, improve the format of the output listing, or document the program.

The basic component of the Varian RPG IV system is the two-part compiler. This compiler accepts RPG IV source program statements and, in one pass, produces both a listing of the program and an object deck. The listing includes diagnostics and error messages. The object deck, the RPG IV loader, and the RPG runtime support program process the data given in the data-defining portion of the source program.

The principal output of RPG IV is a printed record output on the line printer, and ready for reproduction and/or distribution. In certain applications, the program outputs a portion or all of the processed data on punched cards to be used as input for later data processing. For example, the record of an updated end-of-the-month inventory can be used as start-of-the-month data at the end of the following month.

Specific operating procedures for both the stand-alone and MOS-controlled versions of RPG IV are given in the Varian RPG IV Manual (document number 98 A 9947 03x).

SECTION 24 - BASIC LANGUAGE

The Varian 73 Basic Language is a popular, easy-to-use programming system for a wide variety of business and scientific applications. BASIC has many of the characteristics of ordinary mathematical notation: simple vocabulary and grammar, plus problem-solving steps can be specified completely and precisely.

The simplicity of BASIC, and its conversational operation, permit the inexperienced programmer to write and execute useful computer programs with a minimum of training. The computer responds to all commands entered by the operator, thus reinforcing the learning process. If the operator makes an error, diagnostics report the type of error, and corrections can be made immediately.

BASIC (Beginner's All-Purpose Symbolic Instruction Code) was originally developed at Dartmouth College. Varian's version of BASIC adapts it for use on the Varian 73 computer system.

For the experienced programmer, the Varian version of BASIC includes expanded instructions and capabilities. The advanced features permit wider-range programming applications, while retaining the inherent simplicity of the language.

Only 8K of memory and a Teletype are required for using BASIC in Varian 73 computer systems. Even dedicated computers can perform general computation when they are not being used for other primary functions.

BASIC is described in detail in the Varian BASIC Language Manual (document number 98 A 9952 03x).

The simplicity of programming with BASIC is illustrated by the following features of the language:

a. Each line of the BASIC program begins with a **line number** that identifies the statement, and specifies the order in which the statement is to be processed by the computer. The program can be written in any order since the computer sorts and edits it as specified by the line numbers.

b. Each statement has a word following the line number. This word specifies the **type of statement** and, thus, the operation to be performed by the computer in self-defining terms.

c. BASIC uses only **capital letters** corresponding to the letters of the Teletype keyboard, which is used to input the programs.

d. Each statement is **free-form**. Thus, statement fields and spacing can be disregarded during input.

e. BASIC uses the following five **arithmetic operators**, each indicated by the corresponding symbol:

Addition	+
Subtraction	-
Multiplication	*
Division	/
Exponentiation	↑

the following **arithmetic relationships**

Equal to	=
Not equal to	#
Less than	<
Greater than	>
Not less than	> =
Not greater than	< =

the following **logical operators**:

AND OR NOT

a wide variety of **mathematical and special functions**, and complete **matrix operations**.

BASIC can be operated in two modes: program and calculator. Program mode is defined as entering a set of numbered BASIC statements that are checked for validity and stored, then entering the control command RUN to start execution of the program. The calculator mode causes the computer to respond immediately to a BASIC statement; in this mode, the operator enters a BASIC statement, but not a statement number, and the BASIC system executes it immediately.

BASIC also features:

a. Immediate syntax-checking, statement diagnosis, and error correction.

b. Optional Teletype or high-speed paper tape input/output.

c. Optimum use of memory through operator library selection.

d. Control commands to halt and begin program execution, for input/output selection, and for deleting a program from memory.

SECTION 25 - PERT

Varian PERT is a minicomputer-based system for performing scheduling analyses by the **P**rogram **E**valuation and **R**eview Technique.

The PERT system operates upon the elements of a job schedule by combining them into a network so that their interdependencies and interrelationships are represented in an easily visualized form. Thus, PERT is a visual aid to managers responsible for project control. Using PERT, a project manager is able to measure progress, determine task responsibility, and identify both slack periods and potential bottlenecks in complex projects spanning a year or more in time.

Varian's PERT system provides a unique interactive capability which allows the user to create, update, and reschedule their PERT networks rapidly via a terminal, thus avoiding the costly and time-consuming batch-oriented methods. The PERT technique has been adopted and used by a wide variety of governmental agencies and industries. It is effective whenever complex programs are to be completed within specified limits of time and money. Varian's PERT system makes this powerful tool available for the first time to the minicomputer user, and in addition gives him an interactive capability not found in the larger systems.

PERT can be used on any Varian 73 computer system running under the Master Operating System (MOS) or VORTEX, with FORTRAN. The system must have the following minimum configuration:

a. Processor with 16K of memory

b. 33/35 ASR Teletype

c. One of the following:
   1. Magnetic tape unit
   2. Rotating memory device on a buffer interlace controller (BIC)

PERT is described in detail in the Varian PERT Users Manual (document number 98 A 9952 19x).

SECTION 26 - MATHEMATICAL SUBROUTINES

In support of Varian 73 computer applications programs that require mathematical computation, Varian provides a comprehensive Mathematical Subroutine Library with complete, easily accessible subroutines.

The mathematical subroutines are grouped into four major categories: fixed-point arithmetic, floating-point arithmetic, arithmetic functions (both real and complex), and number and character conversions. The subroutines are called by other programs and fill the mathematical requirements of virtually all computer applications.

The mathematical subroutine library is described in detail in the Varian Subroutine Descriptions Manual (document number 98 A 9902 04x).

Fixed-Point Arithmetic

The fixed-point arithmetic subroutines are for applications that demand a high-speed arithmetic package. They include:

a. Addition, subtraction, multiplication, and division (single- and double-precision)

b. Two's complement (double-precision)

c. Absolute value

d. Transfer of sign

Fixed-point, double-precision addition (**XDAD**) adds the double-precision number whose address is in the calling sequence to the double-precision number in the A and B registers. The low-order halves of the numbers are added first, and, if there is a carry, it is added to the high-order sum.

Fixed-point, double-precision subtraction (**XDSU**) subtracts the double-precision number whose address is in the calling sequence from the double-precision number in the A and B registers.

Fixed-point, double-precision multiplication (**XDMU**) multiplies the double-precision number whose address is in the calling sequence by the double-precision number in the A and B registers. The most-significant two words of the 4-word product are returned in the A and B registers. The least-significant two words are lost.

Fixed-point, double-precision division (**XDDI**) divides the double-precision number in the A and B registers by the double-precision number whose address is in the calling sequence. The 2-word quotient is returned in the A and B registers.

Fixed-point, double-precision two's complement (**XDCO**) takes the two's complement of the double-precision number in the A and B registers. XDCO complements the number, then tests the low-order bits for a carry.

Fixed-point, integer absolute value (**IABS**) takes the absolute value of the signed integer in the A register. If the number is negative, IABS one's complements it, then corrects it to two's complement form.

Fixed-point, integer sign transfer (ISIG) applies the sign of the integer whose address is in the calling sequence to the quantity in the A and B registers.

Floating-Point Arithmetic

The floating-point subroutines provide higher accuracy, more flexibility, and wider number ranges than fixed-point arithmetic. Floating-point subroutines include:

a. Addition, subtraction, multiplication, and division

b. Absolute value

c. Sign copy

d. Mantissa separation

e. Normalization

Floating-point addition ($QK) algebraically adds the floating-point number in the A and B registers to the floating-point number whose address is in the calling sequence.

Floating-point subtraction ($QL) computes the difference of the floating-point minuend in the A and B registers and the floating-point subtrahend whose address is in the calling sequence.

Floating-point multiplication ($QM) multiplies the floating-point number in the A and B registers by the number whose address is in the calling sequence. $QM separates the mantissa and calls XDMU to implement the arithmetic operation.

Floating-point division ($QN) divides the floating-point number in the A and B registers by the number whose address is in the calling sequence. $QN separates the mantissa and calls XDDI to implement the arithmetic operation.

Floating-point, real-number absolute value (ABS) takes the absolute value of the floating-point, real quantity in the A and B registers. If the number is negative, ABS one's complements it and returns the result in the A and B registers.

Sign copy (SIGN) sets the sign of the floating-point number in the A and B registers equal to the sign of the quantity whose address is in the calling sequence.

The mantissa separation subroutines ($FMS, $FSM) separate the floating-point number in the A and B registers and return the mantissa in the A and B registers and the characteristic in the X register.

Normalization ($NML) normalizes the floating-point, double-precision number in the A and B registers. $NML tests the sign, two's complements the number using XDCO, and returns the fixed-point result in the A and B registers and the sign flag in the X register.

Arithmetic Functions

Subroutines are provided for the following arithmetic functions:

a. Logarithm

b. Exponential function

c. Square root

d. Sine

e. Cosine

f. Arctangent

g. Polynomial

h. Exponentiation

Fixed-point, single-precision logarithm (**XLOG**) computes the natural logarithm of the quantity in the A register. XLOG uses a Chebychev polynomial of the fifth degree.

Floating-point, double-precision logarithm (**ALOG**) computes the natural logarithm of the quantity whose address is in the calling sequence, returning the result in the A and B registers.

Fixed-point, single-precision exponential function, positive argument (**XEXP**) computes the exponential of the positive quantity in the A register.

Fixed-point, single-precision exponential function, negative argument (**XEXN**) computes the exponential of the negative quantity in the A register.

Floating-point exponential function (**EXP**) computes the exponential of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision square root (**XSQT**) takes the unrounded square root of the quantity in the A register (if it is nonnegative) and returns the result in the A register.

Floating-point square root (**SQRT**) takes the square root of the floating-point number whose address is in the calling sequence.

Fixed-point, single-precision sine (**XSIN**) computes the sine of the quantity in the A register, returning the result in the A register.

Floating-point sine (**SIN**) computes the sine of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision cosine (**XCOS**) takes the cosine of the quantity in the A register and returns the result in the A register.

Floating-point cosine (**COS**) takes the cosine of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision arctangent (**XATN**) computes the arctangent of the quantity in the A register, returning the result in the A register.

Floating-point arctangent (**ATAN**) computes the arctangent of the floating-point quantity whose address is in the calling sequence.

Fixed-point, single-precision polynomial (**POLY**) supports the fixed-point, single-precision mathematical subroutines that require the evaluation of a polynomial in one variable of any finite degree. The polynomial is evaluated in Horner form:

Fixed-point, integer exponentiation (**$HE**).

Integer/floating-point exponentiation (**$PE**).

Floating-point exponentiation (**$QE**).

Conversions

The number and character conversion subroutines include:

a. Fixed-point/floating-point

b. Binary/decimal

c. EBCDIC/Hollerith

d. EBCDIC/ASCII

Fixed-point, single-precision integer to floating-point conversion (**$QS**) converts the signed integer in the A register to floating-point format.

Floating-point to fixed-point, single-precision integer conversion (**$HS**) converts the floating-point number in the A and B registers to integer format.

Fixed-point, single-precision binary-to-decimal conversion (**XBTD**) converts the absolute value of the integer in the A register to a four-digit decimal-coded integer in the B register.

Fixed-point, single-precision decimal-to-binary conversion (**XDTB**) converts the four-digit, binary-coded-decimal integer in the A register to a pure binary integer in the B register.

EBCDIC-to-Hollerith conversion (**SA01**) converts and eight-bit EBCDIC character in the A register to its equivalent 12-bit Hollerith code, returning the result in the A register.

Hollerith-to-EBCDIC conversion (**SB01**) converts a 12-bit Hollerith code in the A register to its equivalent eight-bit EBCDIC character, returning the result in the A register.

EBCDIC-to-ASCII conversion (**SC01**) converts an eight-bit EBCDIC character in the A register to its equivalent eight-bit ASCII code, returning the result in the A register. This subroutine can be modified to produce seven-bit ASCII codes.

SECTION 27 - MAINTAIN III TEST PROGRAMS

The **Varian 73 MAINTAIN III Test Programs** comprise a system approach to testing and maintaining Varian 73 computers. MAINTAIN III is designed to verify total system operation, and to minimize maintenance time by aiding in the isolation of system malfunctions to a specific area.

The major functional elements of the MAINTAIN III system (illustrated in figure 27-1) are:

- Test executive
- Preliminary and comprehensive CPU and memory test programs
- Computer and I/O option test programs
- Peripheral and I/O interface test programs

MAINTAIN III is described in detail in the MAINTAIN III Manual (document number 98 A 9952 07x).

Test Executive Program

The test executive program controls the MAINTAIN III system. It includes preliminary CPU (instructions) and memory tests, a binary loader, and the test executive.

The preliminary instructions test portion of the test executive program validates basic CPU operation, the preliminary memory test monitors the operation of the first 8K of memory, and the binary loader reads object program data and stores it in memory.

The test executive:

a. Provides directives to control testing activities

b. Loads and executes the other MAINTAIN III test programs.

c. Contains a utility subroutine package of aids for debugging, maintenance, and troubleshooting

d. Includes standard test subroutines (Teletype input/output, program delays, SENSE switch options, etc.)

The operator communicates with the test executive through the Teletype keyboard and printer. Directives and parameters input from the keyboard control execution and monitoring of associated test programs. To accommodate the minimum memory system, the test executive operates with only one test program in memory at a time.

The utility subroutines of the test executive provide the software to:

a. Display and alter memory and register contents

b. Search memory for specific data patterns

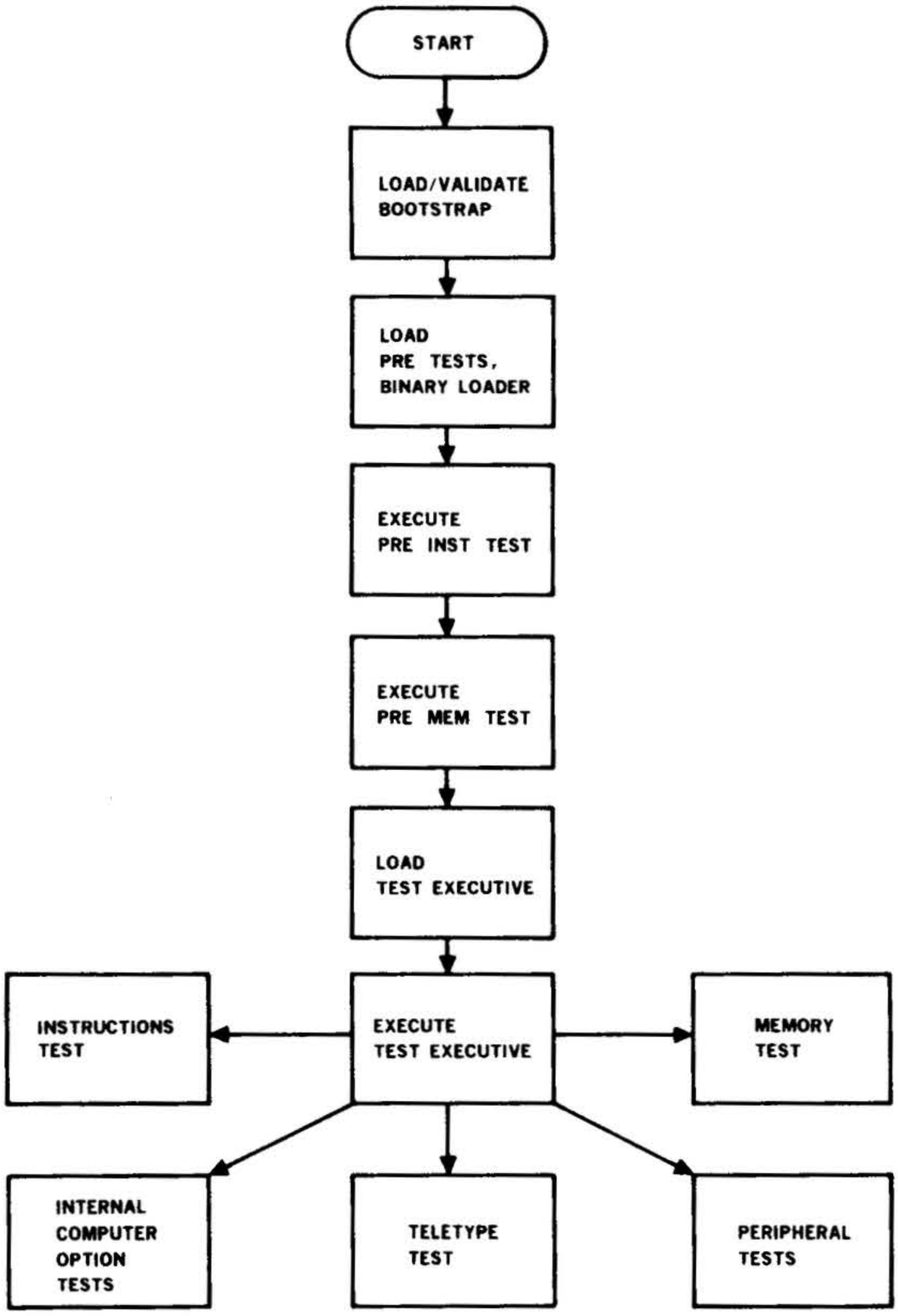
c. Set areas of memory to various data patterns

d. Create object code

e. Interrupt test programs during their execution

Test Programs

MAINTAIN III test programs exercise the computer, options, and associated peripherals with sequences of instructions. If an instruction produces incorrect results, the sequence is halted and error messages indicate the failing instruction or opera-



NOTE: The Test Executive operates with only one test program in memory at a time.

VTI1-0946A

Figure 27-1. MAINTAIN III Test Program System

tion. The operator can then repeat, continue, or halt the program until the fault is isolated. Good maintenance procedures include:

a. Either on a routine preventive basis or when a system malfunction is suspected, run the MAINTAIN III test programs and eliminate from consideration the functional areas that are operating properly to isolate a fault to a specific area.
b. Exercise the area of a suspected fault by executing, repeating, or modifying the applicable test program.
c. Correct the fault by replacing the faulty component or circuit card, and restore the system to normal operation.
d. Verify system operation by rerunning the test program.

The maintenance and reference manuals appropriate to the system installation describe theory of operation, timing, and signal locations and levels for the system components. Also given are system checkout procedures using the computer control panel and recommended test equipment.

MAINTAIN III test programs are normally supplied on punched paper tape; other media such as card decks or magnetic tape are available. Loading and operating procedures are delineated in the MAINTAIN III Manual (document number 98 A 9952 07x).

GLOSSARY

A

accumulator -- A part of the arithmetic unit of a digital computer where numbers are totaled (i.e., accumulated) and temporarily stored.

accuracy -- Freedom from error (not the same as precision).

adder -- A device for forming the sum of two numbers.

address -- 1. A label, name, or number identifying a register, memory location, or unit where information is stored. 2. The operand part of an instruction.

alphanumeric -- Coding system using letters and numbers.

arithmetic unit -- A device (or part of the computer) for performing basic operations including addition, subtraction, multiplication, and division.

ASCII -- Acronym for American Standard Code for Information Interchange.

assemble -- To put a program through the process of assembly by means of an assembly program.

assembly language -- A symbolic language used for programming which must go through an assembly in order to be converted into the machine code required for operation on a computer.

assembly program -- The program which operates on a symbolic language program to produce a machine language program in the process of assembly.

B

base -- The radix of a number notation; the decimal system is to the base 10.

BCD -- Acronym for binary coded decimal.

binary -- A numbering system to the base 2.

binary code -- A code of binary numbers (0 and 1).

binary coded decimal -- A method of using groups of binary digits to represent decimal numbers, with each digit position of a decimal number being allocated four bits.

binary digit -- Either 0 or 1.

bit -- A binary digit.

block -- A group of computer words.

branch -- A point in a program where there is a choice of steps; a program jump.

buffer -- Temporary storage.

bug -- Program (or computer) error or omission.

bus -- A group of conductors used for transmitting signals or power.

byte -- A set of binary digits (8 bits) considered as a unit; one half of a word.

C

card -- A stiff paper, about 7-3/8 by 3-1/4 inches, which is punched or marked with data to be sensed electronically or visually.

card punch -- A machine for punching data on cards.

card reader -- A machine for sensing data contained in punched cards.

carry -- In addition, the overflow from one column to the next higher column.

character -- An elemental mark such as a letter, number, or special symbol.

clear -- To replace data in a storage device with some standard character, e.g. zero or blank.

code -- A group of characters representing data or instructions.

coding -- The translation of a program into actual machine instructions.

collate -- To arrange material (usually from more than one source) in a specific sequence.

column -- A vertical array of characters.

compile -- To assemble subroutines into a program.

complement -- The difference between an integer and it's radix. For example 6 and 4 are complements in decimal (radix equal to 0) notation.

conditional jump -- A point in a program where the computer skips a series of instructions if specified conditions are met.

control -- A device or signal that affects other devices or signals.

convert -- To change from one state to another.

core -- A small donut shaped piece of magnetic material capable of retaining a positive or negative charge indefinitely, and it reverses its charge when a current is passed through it.

core memory -- A computer memory composed of magnetic cores.

counter -- A device for recording or registering a sequence of events.

cycle -- A complete series of steps or events.

D

debugging -- The technique of detecting, diagnosing, and correcting errors (also known as bugs) which may occur in programs or systems (both hardware and software).

decimal -- Pertaining to the base 10 numbering system.

diagnostic routine -- A program on the computer used for troubleshooting.

digit -- A single character or symbol.

double-precision number -- A number that is twice as long as a number normally handled.

dump -- To write the contents of memory in a form suitable for future use.

E

error -- Incorrect procedure, number, or result.

exponent -- The power to which a quantity is raised; e.g. in the expression 24 the exponent is 4.

F

feed holes -- Holes punched in paper tape to enable it to be driven by sprockets.

ferrite core -- See core.

field -- A group of bits (less than one computer word) considered as a unit of information.

fixed point -- A system of notation in which the decimal or binary point is fixed with respect to one end of the numbers used.

flip-flop -- A bistable (two-state) device in which the two possible outputs can be labeled on-off, 0-1, left-right, etc.

floating point -- A system of notation which takes into account the varying location of the decimal or binary point by expressing each number as a sign, coefficient, and base power.

flowchart -- A program or routine expressed block diagram form.

full duplex -- Transmission circuits in which messages may be transmitted in both directions at the same time.

G

gate -- A device that produces an output signal when certain specific conditions are met.

gray code -- A special binary code where successive numbers change by only one digit.

H

half duplex -- Transmission circuits in which messages may be transmitted in both directions but not simultaneously.

hard copy -- A printed copy of machine output in a visually readable form.

hardware -- The physical units making up a computer system - the apparatus as opposed to the programs (software).

hexadecimal -- A notation of numbers to the base of sixteen. The ten decimal digits 0-9 are used and in addition six more digits, A, B, C, D, E, and F, to represent 10-15 respectively.

high order -- The more significant figure or figures in a number expressed in positional notation.

hold -- To retain information after copying.

I

indexing -- A method of address modification.

indirect address -- An address that specifies a storage location that contains either a direct address or another indirect address.

information -- Any form of data, such as an operand or instruction.

inhibit -- To prevent a particular signal for occurring, or to prevent a particular operation from being performed.

input -- The process of transferring data, or program instructions, into memory from some peripheral unit.

instruction -- A form of information that tells the computer what operation to perform, where to get the operand, and what to do with the result.

interrupt -- A break in a program or routine caused by an external source, which requires that control should pass temporarily to another routine.

I/O -- Input/output.

J

jump -- A change in a program in which the next sequential instruction is not executed; instead the computer skips one or more instructions.

K

k -- An abbreviation for Kilo, used to denote a thousand.

L

language -- A complete, organized set of characters together with the rules for their use.

leader -- A length of paper tape that precedes the data recorded on a reel or strip of paper tape that contains feed holes only.

library -- A collection of subroutines, complete programs, etc.

logic -- A formal set of operational rules for computer device or circuit behavior.

loop -- A closed set or ring of instructions.

low order -- The least significant figure or figures in a number expressed in positional notation.

M

Magnetic Core -- A small toroid, with two stable states, used in high-speed memories.

magnetic disc -- A rotating cylinder, the magnetized surface of which stores information.

magnetic tape -- A tape with a magnetic surface on which data can be stored by selective polorization of portions of the surface.

maintenance -- Any activity intended to eliminate faults or to keep hardware or programs in satisfactory working condition, including tests, measurements, replacements, adjustments, and repairs.

matrix -- An array of components formed by the intersection of vertical and horizontal elements.

memory -- A device or place for information storage.

microsecond -- One millionth of a second.

mnemonic -- Relating to a system for remembering codes.

modem -- An acronym for modulator/demodulator. A device which enables data to be transmitted over long distances without error.

multiplexor -- A communications control device which enables a central processor to be connected to a large number of different communications channels, any or all of which may be transferring data to or from the processor.

N

nanosecond -- One thousand-millionth of a second.

normalize -- To standardize, as to adjust numbers to floating-point format.

O

octal digit -- A digit in the octal (base 8) number system.

off-line -- Pertaining to equipment or devices not under control of the central processing unit.

on-line operation -- Computer control of input or output data where the data are processed as soon as available.

op code -- Abbreviation for operation code. The code which specifies the particular operation to be performed.

operand -- A quantity used in or resulting from an operation.

operator -- One who runs programs on a computer.

output -- Information transferred from a central processor to an output device.

P

parity check -- A method for checking the bits in a computer word to assure that none were lost.

peripheral -- Devices used with, but not an integral part of the computer, i.e., line printers, card readers/punches, etc.

pipelining -- The ability to fetch the next instruction in sequence while the current instruction is being executed.

precision -- The exactness of a measurement as opposed to the degree of accuracy or correctness.

program -- An ordered series of steps followed by the computer to solve a specific problem or class of problems.

programmer -- One who develops or writes a program.

R

radix point -- The point in a system of numbers separating integers from fractions.

read -- To acquire information from some form of storage.

register -- A device for the temporary storage of information, such as a computer word.

rotating memory device (RMD) -- A disc storage device in which the data is stored on a rotating disc.

round off -- To remove some of the less significant digits from a number, resulting in a smaller, less accurate number.

routine -- A set of computer instructions in a specific sequence.

S

semiconductor memory -- A computer memory composed of metal oxide semiconductors (MOS).

shift -- To displace a number to the right or left a given number of positions.

software -- A set of computer programs.

sort -- To separate information into two or more classes or groups.

subroutine -- The set of instructions, in machine code, to direct the computer to carry out a well defined mathematical or logical operation; a part of a routine.

V

verifier -- A device for checking against errors.

W

word -- A group of alphanumeric characters with specific meaning, e.g., a computer word.

write -- To output data to some device for storage, printing, or further manipulation.

APPENDIX A - INDEX OF INSTRUCTIONS

Mnemonic	Octal Code	Description	Page
ADD	12xxxx	Add memory to A register	14-8
ADDE	00612x	Add extended	14-8
ADDI	006120	Add immediate	14-8
ANA	15xxxx	AND memory and A register	14-12
ANAE	00615x	AND extended	14-13
ANAI	006150	AND immediate	14-13
AOFA	005511	Add overflow to A register	14-22
AOFB	005522	Add overflow to B register	14-22
AOFX	005544	Add overflow to X register	14-22
ASLA	004200 + n	Arithmetic shift left A register	14-17
ASLB	004000 + n	Arithmetic shift left B register	14-17
ASRA	004300 + n	Arithmetic shift right A register	14-16
ASRB	004100 + n	Arithmetic shift right B register	14-16
BT	0064xx	Bit test	14-33
CIA	1025xx	Clear and input ot A register	14-47
CIAB	1027xx	Clear and input to A and B registers	14-47
CIB	1026xx	Clear and input to B register	14-47
COMP	005xxx	Complement source to destination registers	14-24
CPA	005211	Complement A register	14-22
CPB	005222	Complement B register	14-22
CPX	005244	Complement X register	14-22
DAR	005311	Decrement A register	14-21
DBR	005322	Decrement B register	14-21
DECR	0053xx	Decrement source to destination registers	14-24
DIV	17xxxx	Divide	14-9
DIVE	00617x	Divide extended	14-10
DIVI	006170	Divide immediate	14-10
DXR	005344	Decrement X register	14-21

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
ERA	13xxxx	Exclusive-OR memory and A register	14-12
ERAE	00613x	Exclusive-OR extended	14-12
ERAI	006130	Exclusive-OR immediate	14-12
EXC	100xxx	External control	14-46
EXC2	104xxx	Auxiliary external control	14-46
FAD	105410	Add single precision memory to floating point accumulator	14-50
FADD	105503	Add double precision memory to floating point accumulator	14-50
FDV	105401	Single precision floating point divide	14-52
FDVD	105535	Double precision floating point divide	14-52
FIX	105621	Reformat-floating point accumulator and store integer in memory	14-50
FLD	105420	Load floating point accumulator with single precision number	14-49
FLDD	105522	Load floating point accumulator with double precision number	14-49
FLT	105425	Reformat single precision integer and load into floating point accumulator	14-50
FMU	105416	Single precision floating point multiply	14-57
FMUD	105506	Double precision floating point multiply	14-57
FSB	105450	Single precision floating point subtraction	14-51
FSBD	105543	Double precision floating point subtraction	14-51
FST	105600	Store floating point accumulator in memory in single precision format.	14-49
FSTD	105710	Store floating point accumulator in memory in double precision format.	14-49

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
HLT	000000	Halt	14-45
IAR	005111	Increment A register	14-21
IBR	005122	Increment B register	14-21
IJMP	0067xx	Indexed jump	14-28
IME	1020xx	Input to memory	14-48
INA	1021xx	Input to A register	14-47
INAB	1023xx	Input to A and B registers	14-47
INB	1022xx	Input to B register	14-47
INCR	0051xx	Increment source to destination registers	14-26
INR	04xxxx	Increment memory and replace	14-7
INRE	00604x	Increment memory and replace extended	14-7
INRI	006040	Increment memory and replace immediate	14-7
IXR	005144	Increment X register	14-21
JAN	001004	Jump if A register negative	14-29
JANM	002004	Jump and mark if A register negative	14-35
JANZ	001016	Jump if A register not zero	14-30
JANZM	002016	Jump and mark if A register not zero	14-36
JAP	001002	Jump if A register positive	14-29
JAPM	002002	Jump and mark if A register positive	14-35
JAZ	001010	Jump if A register zero	14-29
JAZM	002010	Jump and mark if A register zero	14-36
JBNZ	001026	Jump if B register not zero	14-30
JBNZM	002026	Jump and mark if B register not zero	14-37
JBZ	001020	Jump if B register zero	14-29
JBZM	002020	Jump and mark if B register zero	14-36
JIF	001xxx	Jump if conditions met	14-32
JIFM	002xxx	Jump and mark if conditions met	14-39
JMP	001000	Jump unconditionally	14-28
JMPM	002000	Jump and mark unconditionally	14-34

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
JOF	001001	Jump if overflow indicator set	14-28
JOFN	001007	Jump if overflow indicator not set	14-28
JOFM	002001	Jump and mark if overflow indicator set	14-35
JOFNM	002007	Jump and mark if overflow indicator not set	14-35
JSR	0065xx	Jump unconditionally and set return in X register	14-32
JS1M	002100	Jump and mark if SENSE switch 1 set	14-37
JS2M	002200	Jump and mark if SENSE switch 2 set	14-37
JS3M	002400	Jump and mark if SENSE switch 3 set	14-38
JS1N	001106	Jump if SENSE switch 1 not set	14-31
JS2N	001206	Jump if SENSE switch 2 not set	14-31
JS3N	001406	Jump if SENSE switch 3 not set	14-32
JS1NM	002106	Jump and mark if SENSE switch 1 not set	14-38
JS2NM	002206	Jump and mark if SENSE switch 2 not set	14-38
JS3NM	002406	Jump and mark if SENSE switch 3 not set	14-38
JSS1	001100	Jump if SENSE switch 1 set	14-30
JSS2	001200	Jump if SENSE switch 2 set	14-31
JSS3	001400	Jump if SENSE switch 3 set	14-31
JXNZ	001046	Jump if X register not zero	14-30
JXNZM	002046	Jump and mark if X register not zero	14-37
JXZ	001040	Jump if X register zero	14-30
JXZM	002040	Jump and mark if X register zero	14-36
LASL	004400+n	Long arithmetic shift left	14-17
LASR	004500+n	Long arithmetic shift right	14-17
LDA	01xxxx	Load A register	14-2
LDAE	00601x	Load A register extended	14-3
LDAI	006010	Load A register immediate	14-5

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
LDB	02xxxx	Load B register	14-2
LDBE	00602x	Load B register extended	14-3
LDBI	006020	Load B register immediate	14-5
LDX	03xxxx	Load X register	14-2
LDXE	00603x	Load X register extended	14-4
LDXI	006030	Load X register immediate	14-5
LLRL	004440 + n	Long logical rotation left	14-14
LLSR	004540 + n	Long logical rotation right	14-14
LRLA	004240 + n	Logical rotation left A register	14-14
LRLB	004040 + n	Logical rotation left B register	14-14
LSRA	004340 + n	Logical shift right A register	14-14
LSRB	004140 + n	Logical shift right B register	14-14
MERG	0050xx	Merge source to destination registers	14-24
MUL	16xxxx	Multiply	14-9
MULE	00616x	Multiply extended	14-9
MULI	006160	Multiply immediate	14-9
NOP	005000	No operation	14-45
OAB	1033xx	Output inclusive-OR of A and B registers	14-47
OAR	1031xx	Output from A register	14-47
OBR	1032xx	Output from B register	14-47
OME	1030xx	Output from memory	14-48
ORA	11xxxx	Inclusive-OR memory and A register	14-11
ORAE	00611x	Inclusive-OR extended	14-11
ORAI	006110	Inclusive-OR immediate	14-11
ROF	007400	Reset overflow indicator	14-45
SEN	101xxx	Program sense	14-46
SOF	007401	Set overflow indicator	14-45
SOFA	005711	Subtract overflow from A register	14-22
SOFB	005722	Subtract overflow from B register	14-22

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
SOFX	005744	Subtract overflow from X register	14-23
SRE	0066xx	Skip if register equal	14-39
STA	05xxxx	Store A register	14-3
STAE	00605x	Store A register extended	14-4
STAI	006050	Store A register immediate	14-5
STB	06xxxx	Store B register	14-3
STBE	00606x	Store B register extended	14-4
STBI	006060	Store B register immediate	14-5
STX	07xxxx	Store X register	14-3
STXE	00607x	Store X register extended	14-4
STXI	006070	Store X register immediate	14-6
SUB	14xxxx	Subtract memory from A register	14-8
SUBE	00614x	Subtract extended	14-8
SUBI	006140	Subtract immediate	14-9
TAB	005012	Transfer A register to B register	14-19
TAX	005014	Transfer A register to X register	14-19
TBA	005021	Transfer B register to A register	14-19
TBX	005024	Transfer B register to X register	14-19
TSA	007402	Transfer switches to A register	14-20
TXA	005041	Transfer X register to A register	14-20
TXB	005042	Transfer X register to B register	14-20
TZA	005001	Transfer zero to A register	14-20
TZB	005002	Transfer zero to B register	
TZX	005004	Transfer zero to X register	14-20
XAN	003004	Execute if A register negative	14-41
XANZ	003016	Execute if A register not zero	14-42
XAP	003002	Execute if A register positive	14-41
XAZ	003010	Execute if A register zero	14-41

INDEX OF INSTRUCTIONS *(continued)*

Mnemonic	Octal Code	Description	Page
XBNZ	003026	Execute if B register not zero	14-42
XBZ	003020	Execute if B register zero	14-42
XEC	003000	Execute unconditionally	14-40
XIF	003xxx	Execute if conditions met	14-44
XOF	003001	Execute if overflow indicator set	14-41
XOFN	003007	Execute if overflow indicator not set	14-41
XS1	003100	Execute if SENSE switch 1 set	14-43
XS2	003200	Execute if SENSE switch 2 set	14-43
XS3	003400	Execute if SENSE switch 3 set	14-43
XS1N	003106	Execute if SENSE switch 1 not set	14-43
XS2N	003206	Execute if SENSE switch 2 not set	14-44
XS3N	003406	Execute if SENSE switch 3 not set	14-44
XXNZ	003046	Execute if X register not	14-43
XXZ	003040	Execute if X register zero	14-42
ZERO	005007	Zero (clear) registers	14-25

n = shift count

APPENDIX B - NUMBER SYSTEMS

Digital computers use a binary number system based on a count (radix) of two. The binary number system has simpler rules than the familiar decimal (radix of 10) number system, making it ideal for computers. The electronic components that make up a digital computer are inherently binary. A relay is either opened or closed; magnetic materials (tape or core) are magnetized in one direction or another; a vacuum tube or transistor is either fully conducting or nonconducting; an electrical pulse can be transmitted at a given time or it cannot be transmitted.

Binary System

In the decimal system, we think in "tens". For example, the number 35 means: $10 + 10 + 10 + 5 = 35$. Or 35 can be written as: $3(10) + 5(1) = 35$. Or 35 can be written in positional notation as: $3(10^{1}) + 5(10^{0}) = 35$. In the pure binary system, we deal with powers of two rather than powers of 10. The positions of the digits do not have the meaning of units, tens, hundreds, thousands, etc.; instead, these positions signify units, twos, fours, eights, sixteens, etc. The sum of these binary positions gives the same decimal sum.

Decimal values

32	16	8	4	2	1

Binary positional notational weight

2^5	2^4	2^3	2^2	2^1	2^0

Remembering that in the binary system we have only two marks, 0 and 1, we then convert the decimal number 35 to a binary number; reading from right to left, we place a one in the first, second, and sixth positions and zeros in the other three positions.

$$1(2^5) + 0(2^4) + 0(2^3) + (2^2) + 1(2^1) + 1(2^0)$$

The resulting binary number is 100011, which is binary for decimal 35 (32 + 0 + 0 + 0 + 2 + 1 = 35).

Decimal-to-Binary Conversion

Method 1. The positional notation chart used in the example above for converting decimal 35 to a binary number suggests a method for decimal-to-binary conversions. We ask the question, what is the largest number to the power of two that can be contained in the decimal number 35. The answer is 32, or 2^5; we place a one in that position. We next ask which power of two can be contained in the remainder 35 - 32, or 3). Since 2^1 equals 2 and 2^0 equals 1 and both are contained in the remainder 3, we place ones in those positions. Hence, binary 100011 = decimal 35.

Figure B-1 is an example of decimal-to-binary conversion using method 1.

Example

Convert decimal 943 to binary:

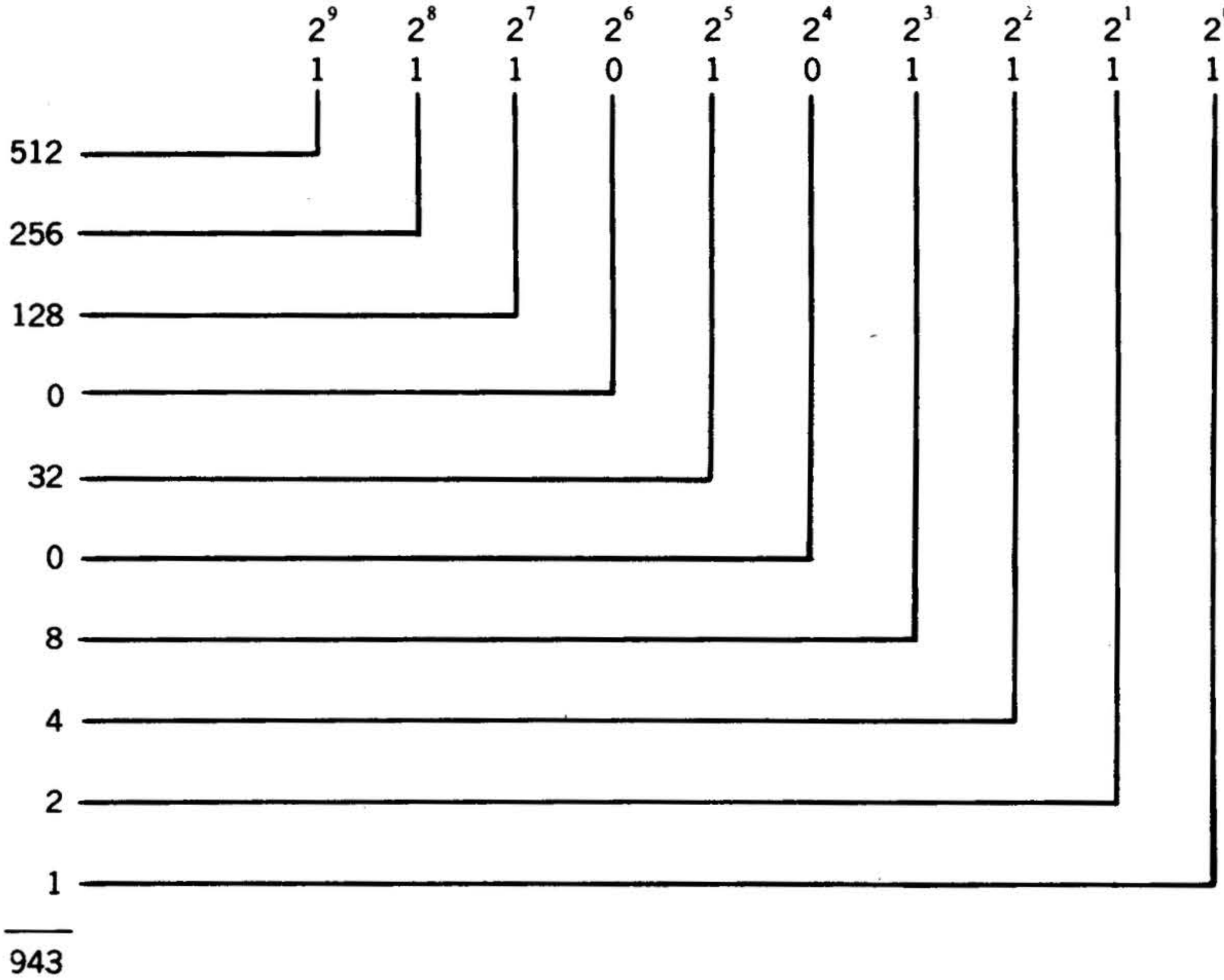


Figure B-1. Converting Decimal to Binary (Method 1)

Method 2. A decimal number can be converted to its binary equivalent by successive division of the number by two. If there is a remainder after the first division, a binary one is placed in the least significant (right-most) binary position. The occurrence or lack of a remainder after each division determines the binary state of each position.

Figure B-2 illustrates decimal-to-binary conversion using method 2.

Example

Convert decimal 135 to binary:

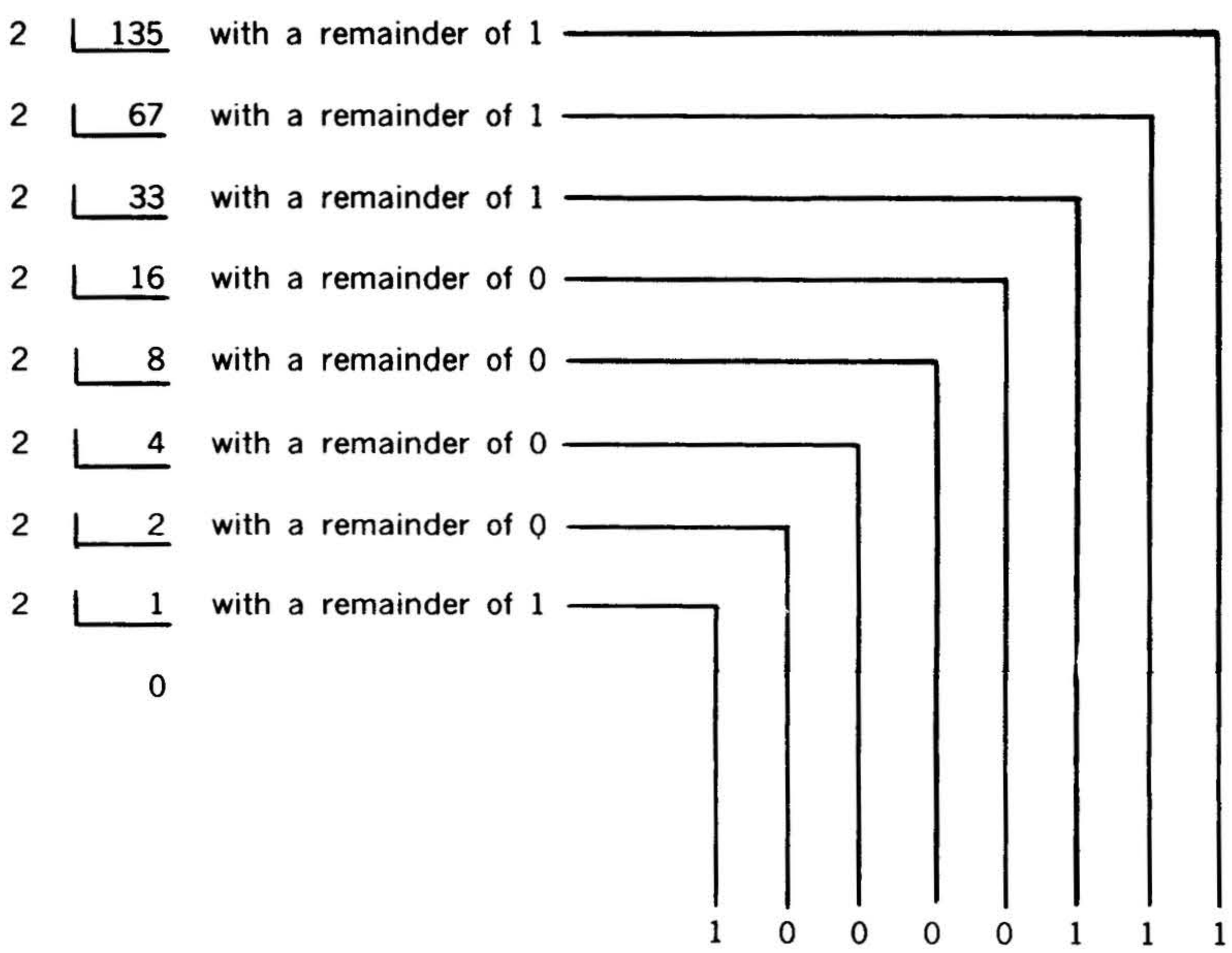


Figure B-2. Converting Decimal to Binary (Method 2)

Binary-to-Decimal Conversion

Binary numbers are converted to their decimal equivalents by multiplying each digit by two (starting with the most significant -- left-most -- digit) and adding the decimal value of the next digit to the right as illustrated in figure B-3.

Convert binary 100001 to decimal:

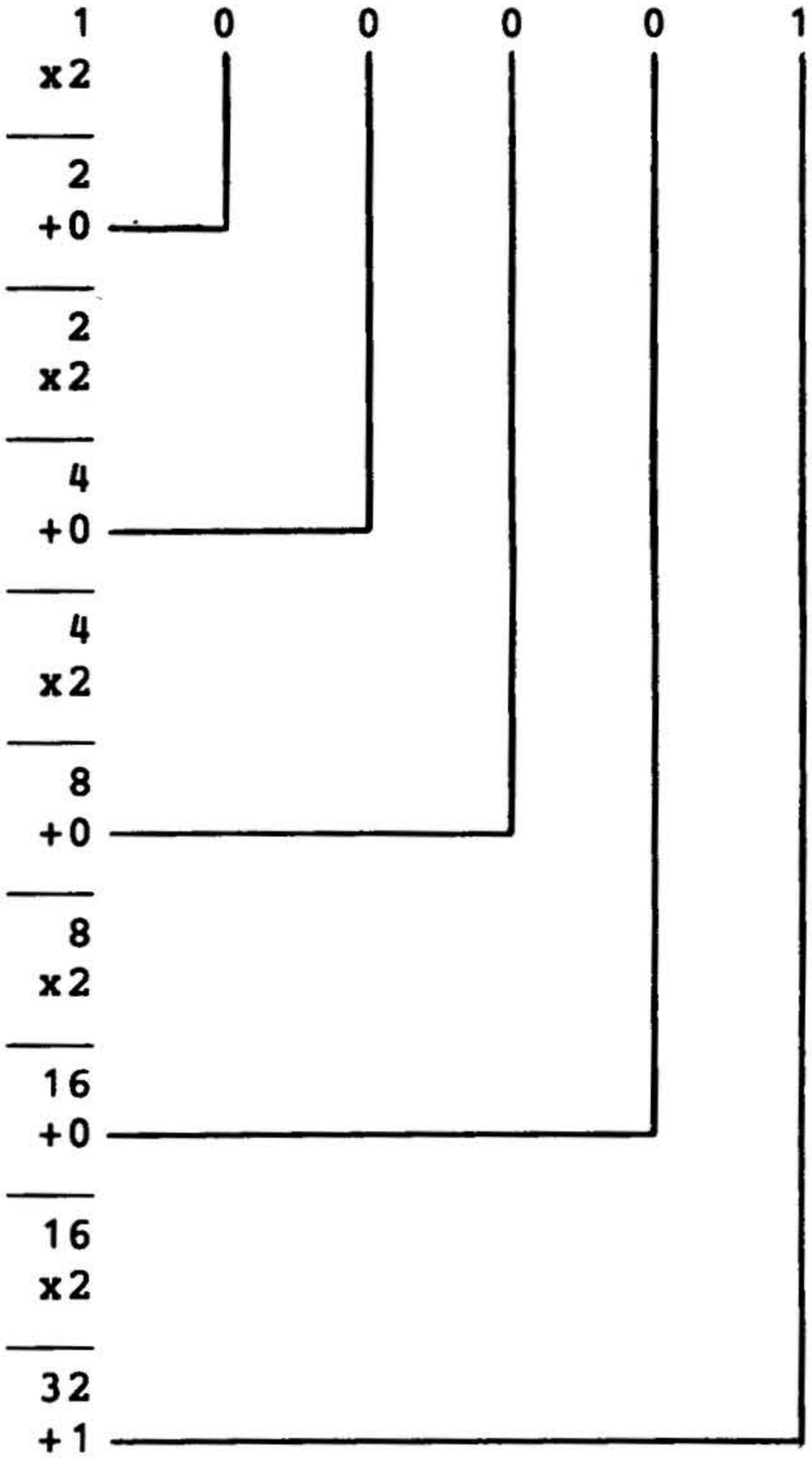


Figure B-3. Binary to Decimal Conversion

Binary Addition

Only four rules apply in binary addition:

0 + 0 = 0	1 + 0 = 1
0 + 1 = 1	1 + 1 = 10

Here 1 plus 1 is 10 (pronounced " one - oh ") because binary 10 is decimal 2. This is the same as saying that decimal 1 plus 1 is 2. Following the above rules, it is possible to add any two binary numbers directly. For example, add decimal 12 and 5:

Decimal	Binary
12	01100
5	101
17	10001

To add 01011 and 00110:

Binary	Decimal
01011	11
00110	6
10001	17

In binary addition, there is the problem of the carry, as when 1 + 1 = 10 -- that is, 0 plus carry 1. This is shown by the following example, where binary 111101 is added to 10110:

(A)	111101
(B)	10110
(C)	101011
(D)	1 1
(E)	1010011

The first step is to add A and B to get the partial sum C. Line D shows the two carries resulting from the 1 + 1 sums. Adding partial sum C and the carries D produces the final sum E, or 1010011.

Binary Subtraction

Four rules apply in binary subtraction:

$$0 - 0 = 0$$
$$1 - 0 = 1$$
$$1 - 1 = 0$$
$$0 - 1 = 1$$

To subtract 1011 from 101101:

```
 101101
-  1011
--------
 100010
```

Note that to subtract 1 from 0, it is necessary to borrow 1, making 1 from 10, or 1.

Complements also provide a means of subtraction. In the decimal system, the ten's complement is the difference between 10 and a given number -- hence, the complement of 7 is 3. The nine's complement is the difference between 9 and a given number, the complement of 7 being 2.

By adding complements, it is possible to subtract. To subtract, using the ten's complement system:

```
  7
 +7      (ten's complement of 3:
----      10-3 = 7)
 14
```

Delete the extra digit (which occurs because of the complement), giving the remainder 4 -- just as in the decimal system 7 - 3 = 4.

Using the nine's complement system:

```
  7
 +6      (nine's complement of 3:
----      9 - 3 = 6)
 13
```

Here the extra 1 is not deleted, but is added to the 3, giving the same answer, 4. This adding of the extra digit, known as end-around carry, is a vital step in computer subtraction.

Since in the binary system there are only two digits, there can be only two complements. To find the one's complement of binary 1, subtract 1 - 1 = 0. To find the one's complement of binary 0, subtract 1 - 0 = 1. Thus, to find the complement of a binary number, change all ones to zeros and all zeros to ones; e.g., the complement of 1011 is 0100.

Thus, binary numbers can be subtracted directly:

```
 1101
-1011
------
 0010
```

And, since the complement of 1011 is 0100, subtraction is also possible by adding complements:

```
 1101
+0100
------
 10001
     1      (end-around carry)
------
 0010
```

Binary Multiplication

Four rules apply in binary multiplication:

0 x 0 = 0	0 x 1 = 0
1 x 0 = 0	1 x 1 = 1

No carries are considered in multiplication. Each digit of the multiplier is examined; when a one is found, the multiplicand is added to the result. When a zero is found in the multiplier, zeros are added to the result. The multiplicand is shifted left one digit for each multiplier digit.

Binary multiplication is thus a series of shifts and additions, as in the decimal system. For example, to multiply 100101 by 101:

```
  100101
     101
  ------
  100101
  00000     (shift left, no add)
100101      (shift left and add)
------
10111001    (sum)
```

For every 1 in the multiplier (101), the multiplicand (100101) is moved one place to the left and added. For every 0 in 101, there is one shift but no addition.

Binary Division

By applying the concepts of binary addition, subtraction, and multiplication, we can divide binary numbers. The divisor is subtracted from the dividend, and a 1 is placed in the quotient. If the divisor cannot be subtracted, a 0 is placed in the quotient.

To divide 101 into 1101010:

```
          10101
       ---------
101  ) 1101010
       101
       ---
        110
        101
        ---
          1    (remainder)
```

Binary-Coded Decimal (BCD) System

This system for representing decimal numbers expresses each decimal digit by a four-digit code called a word) written in binary notation:

BCD		Decimal
0000	=	0
0001	=	1
0010	=	2
0011	=	3
0100	=	4
0101	=	5
0110	=	6
0111	=	7
1000	=	8
1001	=	9
0001 0000	=	10

Thus, decimal 1971 would be expressed in BCD as:

0001	1001	0111	0001
1	9	7	1

Octal System

The octal system of assigning numerical values to binary forms is useful as a shorthand method of writing pure binary numbers. The octal system deals with groups of three binary positions; each group is considered a single digit. This means that, in any octal digit, there is a possibility of eight different binary configurations:

Binary		Octal
000	=	0
001	=	1
010	=	2
011	=	3
100	=	4
101	=	5
110	=	6
111	=	7

Given a series of binary digits, the first three to the left of the binary point are represented by the decimal notation 1, 2, 3.....7 x 8^0 , the next three digits in order are represented decimally by 1, 2, 3.....7 x 8^1 . As can be seen, each group of three binary bits represents some number (from 0 to 7) multiplied by a positional power of eight.

A binary number can be converted without using octal notation; however, the process requires the addition of seven quantities, instead of the three quantities in octal notation. To avoid confusion, octal numbers are designated with a leading zero, e.g., 0173.

Octal-to-Decimal Conversion

Octal representation can be converted to its decimal equivalent by multiplying each digit by eight (starting with the most significant -- left-most -- digit) and adding the decimal value of the next digit to the right as illustrated below.

Convert 0207 to decimal:

```
2      0      7
  x8
-----
 16
 +0
-----
128
 +7
-----
135
```

Figure B-4 shows the relationship of a binary number to its octal and decimal equivalents.

Binary Groups	001	111	011
Octal Notation	1	7	3
Octal Equivalents	$(1x8^2)$	$(7x8^1)$	$(3x8^0)$
Decimal Equivalents	64 +	56 +	3 = 123

Figure B-4. Relationship of Binary, Octal, and Decimal

Decimal-to-Octal Conversion

A decimal number can be converted to its octal equivalent by dividing the decimal number by eight and developing the octal number from the remainder as illustrated in figure B-5.

Convert decimal 135 to octal:

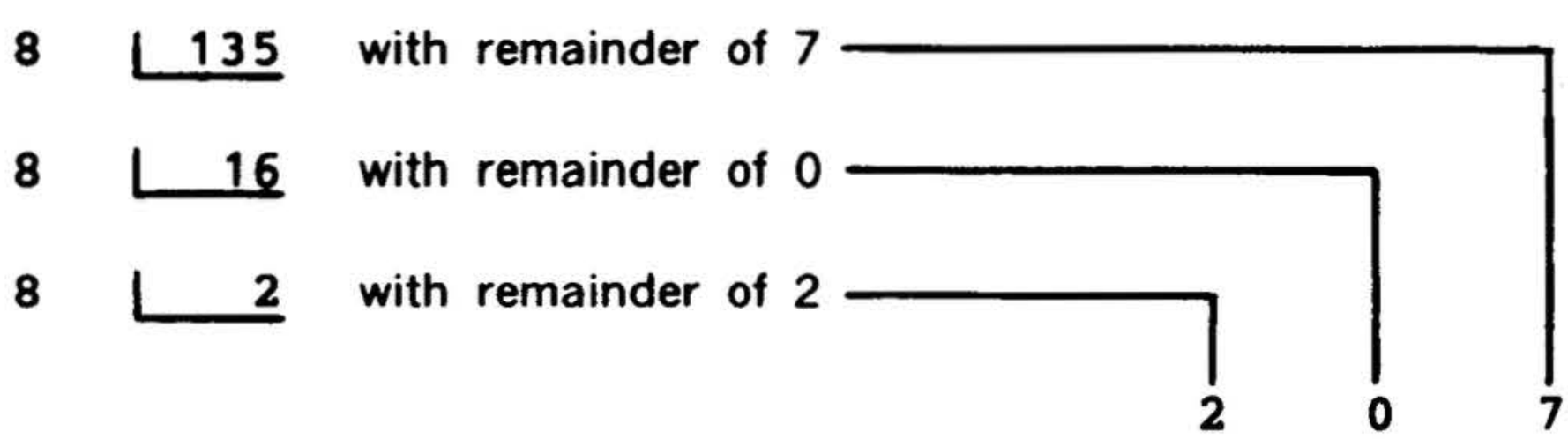


Figure B-5. Decimal to Octal Conversion

Hexadecimal System

Hexadecimal data are expressed to the radix (base) 16 and are related to the decimal numbers as follows:

Decimal	Hexadecimal	Binary
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Thus, hexadecimal numbers proceed from 0 through F (0 through 15 decimal), 10 through 1F (16 through 32 decimal), 20 through 2F (33 through 48 decimal), etc. To avoid confustion, hexadecimal numbers are designated with a leading dollar sign, e.g., $0F3C.

Hexadecimal-to-Decimal Conversion

A hexadecimal number can be converted to its decimal equivalent by expanding each position. Figure B-6 illustrates hexadecimal-to-decimal conversion.

$$
\begin{aligned}
\$55F &= (5 \times 16^2) + (5 \times 16^1) + (15 \times 16^0) \\
&= (5 \times 256) + (5 \times 16) + (15 \times 1) \\
&= 1280 + 80 + 15 \\
&= 1375
\end{aligned}
$$

Figure B-6. Hexadecimal-to-Decimal Conversion

APPENDIX C - POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125

APPENDIX D - OCTAL/DECIMAL INTEGER CONVERSIONS

0000 to 0777 (Octal)	0000 to 0511 (Decimal

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000 to 1777 (Octal)	0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

OCTAL/DECIMAL INTEGER CONVERSIONS

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1509	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

2000 to 2777 (Octal)	1024 to 1535 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

3000 to 3777 (Octal)	1536 to 2047 (Decimal)

OCTAL/DECIMAL INTEGER CONVERSIONS

4000 to 4777 (Octal)	2048 to 2559 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000 to 5777 Octal)	2560 to 3071 (Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

OCTAL/DECIMAL INTEGER CONVERSIONS

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

6000 to 6777 (Octal)	3072 to 3583 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	4022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

7000 to 7777 (Octal)	3584 to 4095 (Decimal)

OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000000	.000000	.000100	.000244	.000200	.000488	.000300	.000732
.000001	.000003	.000101	.000247	.000201	.000492	.000301	.000736
.000002	.000007	.000102	.000251	.000202	.000495	.000302	.000740
.000003	.000011	.000103	.000255	.000203	.000499	.000303	.000743
.000004	.000015	.000104	.000259	.000204	.000503	.000304	.000747
.000005	.000019	.000105	.000263	.000205	.000507	.000305	.000751
.000006	.000022	.000106	.000267	.000206	.000511	.000306	.000755
.000007	.000026	.000107	.000270	.000207	.000514	.000307	.000759
.000010	.000030	.000110	.000274	.000210	.000518	.000310	.000762
.000011	.000034	.000111	.000278	.000211	.000522	.000311	.000766
.000012	.000038	.000112	.000282	.000212	.000526	.000312	.000770
.000013	.000041	.000113	.000286	.000213	.000530	.000313	.000774
.000014	.000045	.000114	.000289	.000214	.000534	.000314	.000778
.000015	.000049	.000115	.000293	.000215	.000537	.000315	.000782
.000016	.000053	.000116	.000297	.000216	.000541	.000316	.000785
.000017	.000057	.000117	.000301	.000217	.000545	.000317	.000789
.000020	.000061	.000120	.000305	.000220	.000549	.000320	.000793
.000021	.000064	.000121	.000308	.000221	.000553	.000321	.000797
.000022	.000068	.000122	.000312	.000222	.000556	.000322	.000801
.000023	.000072	.000123	.000316	.000223	.000560	.000323	.000805
.000024	.000076	.000124	.000320	.000224	.000564	.000324	.000808
.000025	.000080	.000125	.000324	.000225	.000568	.000325	.000812
.000026	.000083	.000126	.000328	.000226	.000572	.000326	.000816
.000027	.000087	.000127	.000331	.000227	.000576	.000327	.000820
.000030	.000091	.000130	.000335	.000230	.000579	.000330	.000823
.000031	.000095	.000131	.000339	.000231	.000583	.000331	.000827
.000032	.000099	.000132	.000343	.000232	.000587	.000332	.000831
.000033	.000102	.000133	.000347	.000233	.000591	.000333	.000835
.000034	.000106	.000134	.000350	.000234	.000595	.000334	.000839
.000035	.000110	.000135	.000354	.000235	.000598	.000335	.000843
.000036	.000114	.000136	.000358	.000236	.000602	.000336	.000846
.000037	.000118	.000137	.000362	.000237	.000606	.000337	.000850
.000040	.000122	.000140	.000366	.000240	.000610	.000340	.000854
.000041	.000125	.000141	.000370	.000241	.000614	.000341	.000858
.000042	.000129	.000142	.000373	.000242	.000617	.000342	.000862
.000043	.000133	.000143	.000377	.000243	.000621	.000343	.000865
.000044	.000137	.000144	.000381	.000244	.000625	.000344	.000869
.000045	.000141	.000145	.000385	.000245	.000629	.000345	.000873
.000046	.000144	.000146	.000389	.000246	.000633	.000346	.000877
.000047	.000148	.000147	.000392	.000247	.000637	.000347	.000881
.000050	.000152	.000150	.000396	.000250	.000640	.000350	.000885
.000051	.000156	.000151	.000400	.000251	.000644	.000351	.000888
.000052	.000160	.000152	.000404	.000252	.000648	.000352	.000892
.000053	.000164	.000153	.000408	.000253	.000652	.000353	.000896
.000054	.000167	.000154	.000411	.000254	.000656	.000354	.000900
.000055	.000171	.000155	.000415	.000255	.000659	.000355	.000904
.000056	.000175	.000156	.000419	.000256	.000663	.000356	.000907
.000057	.000179	.000157	.000423	.000257	.000667	.000357	.000911
.000060	.000183	.000160	.000427	.000260	.000671	.000360	.000915
.000061	.000186	.000161	.000431	.000261	.000675	.000361	.000919
.000062	.000190	.000162	.000434	.000262	.000679	.000362	.000923
.000063	.000194	.000163	.000438	.000263	.000682	.000363	.000926
.000064	.000198	.000164	.000442	.000264	.000686	.000364	.000930
.000065	.000202	.000165	.000446	.000265	.000690	.000365	.000934
.000066	.000205	.000166	.000450	.000266	.000694	.000366	.000938
.000067	.000209	.000167	.000453	.000267	.000698	.000367	.000942
.000070	.000213	.000170	.000457	.000270	.000701	.000370	.000946
.000071	.000217	.000171	.000461	.000271	.000705	.000371	.000949
.000072	.000221	.000172	.000465	.000272	.000709	.000372	.000953
.000073	.000225	.000173	.000469	.000273	.000713	.000373	.000957
.000074	.000228	.000174	.000473	.000274	.000717	.000374	.000961
.000075	.000232	.000175	.000476	.000275	.000720	.000375	.000965
.000076	.000236	.000176	.000480	.000276	.000724	.000376	.000968
.000077	.000240	.000177	.000484	.000277	.000728	.000377	.000972

OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000400	.000976	.000500	.001220	.000600	.001464	.000700	.001708
.000401	.000980	.000501	.001224	.000601	.001468	.000701	.001712
.000402	.000984	.000502	.001228	.000602	.001472	.000702	.001716
.000403	.000988	.000503	.001232	.000603	.001476	.000703	.001720
.000404	.000991	.000504	.001235	.000604	.001480	.000704	.001724
.000405	.000995	.000505	.001239	.000605	.001483	.000705	.001728
.000406	.000999	.000506	.001243	.000606	.001487	.000706	.001731
.000407	.001003	.000507	.001247	.000607	.001491	.000707	.001735
.000410	.001007	.000510	.001251	.000610	.001495	.000710	.001739
.000411	.001010	.000511	.001255	.000611	.001499	.000711	.001743
.000412	.001014	.000512	.001258	.000612	.001502	.000712	.001747
.000413	.001018	.000513	.001262	.000613	.001506	.000713	.001750
.000414	.001022	.000514	.001266	.000614	.001510	.000714	.001754
.000415	.001026	.000515	.001270	.000615	.001514	.000715	.001758
.000416	.001029	.000516	.001274	.000616	.001518	.000716	.001762
.000417	.001033	.000517	.001277	.000617	.001522	.000717	.001766
.000420	.001037	.000520	.001281	.000620	.001525	.000720	.001770
.000421	.001041	.000521	.001285	.000621	.001529	.000721	.001773
.000422	.001045	.000522	.001289	.000622	.001533	.000722	.001777
.000423	.001049	.000523	.001293	.000623	.001537	.000723	.001781
.000424	.001052	.000524	.001296	.000624	.001541	.000724	.001785
.000425	.001056	.000525	.001300	.000625	.001544	.000725	.001789
.000426	.001060	.000526	.001304	.000626	.001548	.000726	.001792
.000427	.001064	.000527	.001308	.000627	.001552	.000727	.001796
.000430	.001068	.000530	.001312	.000630	.001556	.000730	.001800
.000431	.001071	.000531	.001316	.000631	.001560	.000731	.001804
.000432	.001075	.000532	.001319	.000632	.001564	.000732	.001808
.000433	.001079	.000533	.001323	.000633	.001567	.000733	.001811
.000434	.001083	.000534	.001327	.000634	.001571	.000734	.001815
.000435	.001087	.000535	.001331	.000635	.001575	.000735	.001819
.000436	.001091	.000536	.001335	.000636	.001579	.000736	.001823
.000437	.001094	.000537	.001338	.000637	.001583	.000737	.001827
.000440	.001098	.000540	.001342	.000640	.001586	,000740	.001831
.000441	.001102	.000541	.001346	.000641	.001590	.000741	.001834
.000442	.001106	.000542	.001350	.000642	.001594	.000742	.001838
.000443	.001110	.000543	.001354	.000643	.001598	.000743	.001842
.000444	.001113	.000544	.001358	.000644	.001602	.000744	.001846
.000445	.001117	.000545	.001361	.000645	.001605	.000745	.001850
.000446	.001121	.000546	.001365	.000646	.001609	.000746	.001853
.000447	.001125	.000547	.001369	.000647	.001613	.000747	.001857
.000450	.001129	.000550	.001373	.000650	.001617	.000750	.001861
.000451	.001132	.000551	.001377	.000651	.001621	.000751	.001865
.000452	.001136	.000552	.001380	.000652	.001625	.000752	.001869
.000453	.001140	.000553	.001384	.000653	.001628	.000753	.001873
.000454	.001144	.000554	.001388	.000654	.001632	.000754	.001876
.000455	.001148	.000555	.001392	.000655	.001636	.000755	.001880
.000456	.001152	.000556	.001396	.000656	.001640	.000756	.001884
.000457	.001155	.000557	.001399	.000657	.001644	.000757	.001888
.000460	.001159	.000560	.001403	.000660	.001647	.000760	.001892
.000461	.001163	.000561	.001407	.000661	.001651	.000761	.001895
.000462	.001167	.000562	.001411	.000662	.001655	.000762	.001899
.000463	.001171	.000563	.001415	.000663	.001659	.000763	.001903
.000464	.001174	.000564	.001419	.000664	.001663	.000764	.001907
.000465	.001178	.000565	.001422	.000665	.001667	.000765	.001911
.000466	.001182	.000566	.001426	.000666	.001670	.000766	.001914
.000467	.001186	.000567	.001430	.000667	.001674	.000767	.001918
.000470	.001190	.000570	.001434	.000670	.001678	.000770	.001922
.000471	.001194	.000571	.001438	.000671	.001682	.000771	.001926
.000472	.001197	.000572	.001441	.000672	.001686	.000772	.001930
.000473	.001201	.000573	.001445	.000673	.001689	.000773	.001934
.000474	.001205	.000574	.001449	.000674	.001693	.000774	.001937
.000475	.001209	.000575	.001453	.000675	.001697	.000775	.001941
.000476	.001213	.000576	.001457	.000676	.001701	.000776	.001945
.000477	.001216	.000577	.001461	.000677	.001705	.000777	.001949

APPENDIX E - OCTAL/DECIMAL FRACTION CONVERSIONS

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000	.000000	.100	.125000	.200	.250000	.300	.375000
.001	.001953	.101	.126953	.201	.251953	.301	.376953
.002	.003906	.102	.128906	.202	.253906	.302	.378906
.003	.005859	.103	.130859	.203	.255859	.303	.380859
.004	.007812	.104	.132812	.204	.257812	.304	.382812
.005	.009765	.105	.134765	.205	.259765	.305	.384765
.006	.011718	.106	.136718	.206	.261718	.306	.386718
.007	.013671	.107	.138671	.207	.263671	.307	.388671
.010	.015625	.110	.140625	.210	.265625	.310	.390625
.011	.017578	.111	.142578	.211	.267578	.311	.392578
.012	.019531	.112	.144531	.212	.269531	.312	.394531
.013	.021484	.113	.146484	.213	.271484	.313	.396484
.014	.023437	.114	.148437	.214	.273437	.314	.398437
.015	.025390	.115	.150390	.215	.275390	.315	.400390
.016	.027343	.116	.152343	.216	.277343	.316	.402343
.017	.029296	.117	.154296	.217	.279296	.317	.404296
.020	.031250	.120	.156250	.220	.281250	.320	.406250
.021	.033203	.121	.158203	.221	.283203	.321	.408203
.022	.035156	.122	.160156	.222	.285156	.322	.410156
.023	.037109	.123	.162109	.223	.287109	.323	.412109
.024	.039062	.124	.164062	.224	.289062	.324	.414062
.025	.041015	.125	.166015	.225	.291015	.325	.416015
.026	.042968	.126	.167968	.226	.292968	.326	.417968
.027	.044921	.127	.169921	.227	.294921	.327	.419921
.030	.046875	.130	.171875	.230	.296875	.330	.421875
.031	.048828	.131	.173828	.231	.298828	.331	.423828
.032	.050781	.132	.175781	.232	.300781	.332	.425781
.033	.052734	.133	.177734	.233	.302734	.333	.427734
.034	.054687	.134	.179687	.234	.304687	.334	.429687
.035	.056640	.135	.181640	.235	.306640	.335	.431640
.036	.058593	.136	.183593	.236	.308593	.336	.433593
.037	.060546	.137	.185546	.237	.310546	.337	.435546
.040	.062500	.140	.187500	.240	.312500	.340	.437500
.041	.064453	.141	.189453	.241	.314453	.341	.439453
.042	.066406	.142	.191406	.242	.316406	.342	.441406
.043	.068359	.143	.193359	.243	.318359	.343	.443359
.044	.070312	.144	.195312	.244	.320312	.344	.445312
.045	.072265	.145	.197265	.245	.322265	.345	.447265
.046	.074218	.146	.199218	.246	.324218	.346	.449218
.047	.076171	.147	.201171	.247	.326171	.347	.451171
.050	.078125	.150	.203125	.250	.328125	.350	.453125
.051	.080078	.151	.205078	.251	.330078	.351	.455078
.052	.082031	.152	.207031	.252	.332031	.352	.457031
.053	.083984	.153	.208984	.253	.333984	.353	.458984
.054	.085937	.154	.210937	.254	.335937	.354	.460937
.055	.087890	.155	.212890	.255	.337890	.355	.462890
.056	.089843	.156	.214843	.256	.339843	.356	.464843
.057	.091796	.157	.216796	.257	.341796	.357	.466796
.060	.093750	.160	.218750	.260	.343750	.360	.468750
.061	.095703	.161	.220703	.261	.345703	.361	.470703
.062	.097656	.162	.222656	.262	.347656	.362	.472656
.063	.099609	.163	.224609	.263	.349609	.363	.474609
.064	.101562	.164	.226562	.264	.351562	.364	.476562
.065	.103515	.165	.228515	.265	.353515	.365	.478515
.066	.105468	.166	.230468	.266	.355468	.366	.480468
.067	.107421	.167	.232421	.267	.357421	.367	.482421
.070	.109375	.170	.234375	.270	.359375	.370	.484375
.071	.111328	.171	.236328	.271	.361328	.371	.486328
.072	.113281	.172	.238281	.272	.363281	.372	.488281
.073	.115234	.173	.240234	.273	.365234	.373	.490234
.074	.117187	.174	.242187	.274	.367187	.374	.492187
.075	.119140	.175	.244140	.275	.369140	.375	.494140
.076	.121093	.176	.246093	.276	.371093	.376	.496093
.077	.123046	.177	.248046	.277	.373046	.377	.498046

APPENDIX F - V70 SERIES ASCII CHARACTER CODES

Octal	Decimal	Character	029	026	Description
200	128	NUL			Null
201	129	SOH			Start of Heading
202	130	STX			Start of Text
203	131	ETX			End of Text
204	132	EOT			End of Transmission
205	133	ENQ			Enquiry
206	134	ACK			Acknowledge
207	135	BEL			Bell
210	136	BS			Backspace
211	137	HT			Horizontal Tab
212	138	LF			Line Feed
213	139	VT			Vertical Tab
214	140	FF			Form Feed
215	141	CR			Carriage Return
216	142	SO			Shift Out
217	143	SI			Shift In
220	144	DLE			Data Link Escape
221	145	DC1			Device Control 1

V70 SERIES ASCII CHARACTER CODES

Octal	Decimal	Character	029	026	Description
222	146	DC2			Device Control 2
223	147	DC3			Device Control 3
224	148	DC4			Device Control 4
225	149	NAK			Negative Acknowledge
226	150	SYN			Synchronous File
227	151	ETB			End of Transmission Block
230	152	CAN			Cancel
231	153	EM			End of Medium
232	154	SUB			Substitute
233	155	ESC			Escape
234	156	FS			File Separator
235	157	GS			Group Separator
236	158	RS			Record Separator
237	159	US			Unit Separator
240	160	SP	(blank)	(blank)	Space
241	161	!	11/2/8	11/2/8	Exclamation Point
242	162	"	7/8	0/5/8	Quotation Mark
243	163	#	3/8	0/7/8	Pound Sign
244	164	$	11/3/8	11/3/8	Dollar Sign

Octal	Decimal	Character	029	026	Description
245	165	%	0/4/8	11/7/8	Percent Sign
246	166	&	12	12/7/8	Ampersand
247	167	'	5/8	4/8	Apostrophe
250	168	(	12/5/8	0/4/8	Left Paren
251	169	)	11/5/8	12/4/8	Right Paren
252	170	*	11/4/8	11/4/8	Asterisk
253	171	+	12/6/8	12	Plus Sign
254	172	,	0/3/8	0/3/8	Comma
255	173	—	11	11	Minus Sign
256	174	.	12/3/8	12/3/8	Period
257	175	/	0/1	0/1	Slash
260	176	0	0	0	
261	177	1	1	1	
262	178	2	2	2	
263	179	3	3	3	
264	180	4	4	4	
265	181	5	5	5	
266	182	6	6	6	
267	183	7	7	7	

Octal	Decimal	Character	029	026	Description
270	184	8	8	8	
271	185	9	9	9	
272	186	:	2/8	5/8	Colon
273	187	;	11/6/8	11/6/8	Semi-Colon
274	188	<	12/4/8	12/6/8	Less Than
275	189	=	6/8	3/8	Equal Sign
276	190	>	0/6/8	6/8	Greater Than
277	191	?	0/7/8	12/2/8	Question Mark
300	192	@	4/8	0/2/8	At
301	193	A	12/1	12/1	
302	194	B	12/2	12/2	
303	195	C	12/3	12/3	
304	196	D	12/4	12/4	
305	197	E	12/5	12/5	
306	198	F	12/6	12/6	
307	199	G	12/7	12/7	
310	200	H	12/8	12/8	
311	201	I	12/9	12/9	
312	202	J	11/1	11/1	

Octal	Decimal	Character	029	026	Description
313	203	K	11/2	11/2	
314	204	L	11/3	11/3	
315	205	M	11/4	11/4	
316	206	N	11/5	11/5	
317	207	O	11/6	11/6	
320	208	P	11/7	11/7	
321	209	Q	11/8	11/8	
322	210	R	11/9	11/9	
323	211	S	0/2	0/2	
324	212	T	0/3	0/3	
325	213	U	0/4	0/4	
326	214	V	0/5	0/5	
327	215	W	0/6	0/6	
330	216	Xq	0/7	0/7	
331	217	Y	0/8	0/8	
332	218	Z	0/9	0/9	
333	219	[	12/2/8	12/5/8	Left Bracket
334	220	\	11/7/8	0/6/8	Backslash
335	221	]	0/2/8	11/5/8	Right Bracket

V70 SERIES ASCII CHARACTER CODES

Octal	Decimal	Character	029	026	Description
336	222	↑ or ∧	12/7/8	7/8	Vertical Arrow
337	223	←or –	0/5/8	2/8	Horizontal Arrow
340	224	`			Accent Grave
341	225	a			
342	226	b			
343	227	c			
344	228	d			
345	229	e			
346	230	f			
347	231	g			
350	232	h			
351	233	i			
352	234	j			
353	235	k			
354	236	l			
355	237	m			
356	238	n			
357	239	o			
360	240	p			

Octal	Decimal	Character	029	026	Description
361	241	q			
362	242	r			
363	243	s			
364	244	t			
365	245	u			
366	246	v			
367	247	w			
370	248	x			
371	249	y			
372	250	z			
373	251	{			Left Brace
374	252	|			Vertical Line
375	253	}			Right Brace
376	254	~			Sine Curve
377	255	DEL			Delete, Rub Out

APPENDIX G - INSTRUCTION EXECUTION TIME

Instruction execution time is dependent upon a number of complex variables. These varibles include semiconductor or core memory cycle time, instruction type, addressing mode and sequence of instructions executed. Nominal execution times are specified in table G-1. In actual practice some deviation from these specified times is to be expected.

Table G-1. Timing In Nanoseconds

	330 ns Mem.	660 ns Mem.	990 ns Mem.
Load & Store Instructions			
LDA	660	1320	1980
LDAI	660	1320	1980
LDAE	990	1980	2970
LDB	660	1320	1980
LDBI	660	1320	1980
LDBE	990	1980	2970
LDX	660	1320	1980
LDXI	660	1320	1980
LDXE	990	1980	2970
STA	660	1320	1980
STAI	660	1320	2970
STAE	990	1980	2970
STB	660	1320	1980
STBI	660	1320	2970
STBE	990	1980	2970
STX	660	1320	1980
STXI	660	1320	2970
STXE	990	1980	2970
TSA	3300-3795	3341-3836	3300
Arithmetic Instructions			
INR	990	1980	2970
INRI	990	1980	2970
INRE	1320	2640	3960
ADD	660	1320	1980
ADDI	660	1320	1980

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
ADDE	990	1980	2970
SUB	660	1320	1980
SUBI	660	1320	1980
SUBE	990	1980	2970
MUL	4455-4950	4826-5321	5445-6105
MULE	4785-5280	5486-5981	6600-7260
MULI	4290-4785	4661-5156	4950-5610
DIV	4785-5610	5156-5981	5940-6600
DIVI	4620-5445	4991-5816	5445-6105
DIVE	5115-5940	5816-6641	6930-7590
Logic Instructions			
ORA	660	1320	1980
ORAI	660	1320	1980
ORAE	990	1980	2970
ERA	660	1320	1980
ERAI	660	1320	1980
ERAE	990	1980	2970
ANA	660	1320	1980
ANAI	660	1320	1980
ANAE	990	1980	2970

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
Jump Instructions			
JMP JOF JAP JAN JAZ JBZ JXZ JSS1 JSS2 JSS3 JIF JANZ JBNZ JXNZ JS1N JS2N JS3N JOFN	If no jump occurs, timing is 825; for jump operations, timing is 701 plus 371 for each indirect level.	If no jump occurs, timing is 1320; for jump operations, timing is 1320 plus 660 for each indirect level.	If no jump occurs, timing is 1980; for jump operations, timing is 1980 plus 990 for each indirect level.
IJMP	1155	1650	1980
JSR	701	1320	1980
BT	1155*	1650*	1980*
	1320**	1815**	2145**
SRE	1395*	2021*	2970*
	1650**	2186**	3135**

* Times are for conditions met.
** Times are for conditions not met.

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
Jump-And-Mark Instructions JMPM JOFM JANM JAPM JAZM JBZM JXZM JS1M JS2M JS3M JIFM JOFNM JANZM JBNZM JXNZM JS1NM JS2NM JS3NM	If no jump occurs, timing is 825; for jump operations, timing is 1196 plus 371 for each indirect level.	If no jump occurs, timing is 1320; for jump operations, timing is 1980 plus 660 for each indirect level.	If no jump occurs, timing is 1980; for jump operations, timing is 2970 plus 990 for each indirect level.

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
Execution Instructions			
XEC, XOF, XAP, XAN, XAZ, XBZ, XXZ, XS1, XS2, XS3, XIF	If no execution occurs, timing is 495; for executing operations, timing is 701 plus 371 for each indirect level.	If no execution occurs, timing is 1320; for executing operations, timing is 1320 plus 660 for each indirect level.	If no execution occurs, timing is 1980; for executing operations, timing is 1980 plus 990 for each indirect level.
XOFN, XANZ, XBNZ, XXNZ, XS1N, XS2N, XS3N			
Control Instructions			
HLT	N/A	N/A	N/A
NOP	330	660	990
SOF	1155	1155	1155
ROF	1155	1155	1155
Shift Instructions			
LSRA, LSRB, LRLA, LRLB	495+165n*	536+165n*	990 for n=0–3* 990+165n for n=4–31*
LASR, LASL	990+165n* for n=1–31	990+165n* for n=1–31	990+165n* for n=1–31

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
LLSR, LLRL	990+165(n−1)* for n=1−31	990+165(n−1)* for n=1−31	990+165(n−1)* for n=1−31
ASRA, ASLA	495+165n*	536+165n*	990 for n=1−3* 990+165n for n=4−15*
ASRB, ASLB	495+165n*	495+165n* for n=1−15	990 for n=1−3* 990+165n for n=4−15*

*n = number of bit positions shifted.

Register-Change Instructions			
IAR	330	660	990
IBR	330	660	990
IXR	330	660	990
DAR	330	660	990
DBR	330	660	990
DXR	330	660	990
CPA	330	660	990
CPB	330	660	990
CPX	330	660	990
TAB	330	660	990
TAX	330	660	990
TBA	330	660	990
TBX	330	660	990
TXA	330	660	990
TXB	330	660	990
TZA	330	660	990
TZB	330	660	990
TZX	330	660	990
AOFA	330	660	990
AOFB	330	660	990
AOFX	330	660	990
SOFA	330	660	990
SOFB	330	660	990
SOFX	330	660	990

Table G-1. Timing In Nanoseconds *(continued)*

	330 ns Mem.	660 ns Mem.	990 ns Mem.
MERGE	330-825	660-1320	990-1980
INCR	330-825	660-1320	990-1980
DECR	330-825	660-1320	990-1980
COMPL	330-825	660-1320	990-1980
ZERO	330-825	660-1320	990-1980
I/O Instructions			
SEN	2145-2805	2475-3135	2805-3465
EXC	1815-2310	1815-2310	1815-2310
ECX2, CIA, CIB, CIAB, INA, INB	2310-2805	2310-2805	2310-2805
INAB	2475-2970	2475-2970	2475-2970
OAR, OBR, OAB	2310-2805	2310-2805	2310-2805
IME	2805-3300	3465-3960	4125-4620
OME	3135-3630	3795-4290	4620-5115

Microinstructions

With the optional writable control store, 64-bit microinstructions are executed in 190 nanoseconds.