Digital Equipment Corporation
Maynard, Massachusetts

PDP-15 Systems

Volume 1

Maintenance Manual

Ron Moorhead

DEC-15-H2BB-D

PDP-15 SYSTEMS
MAINTENANCE MANUAL
VOLUME 1

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

1st Printing October 1970
2nd Printing December 1970

The material in this manual is for information purposes and is subject to change without notice.

CONTENTS

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

CONTENTS (Cont)

ILLUSTRATIONS

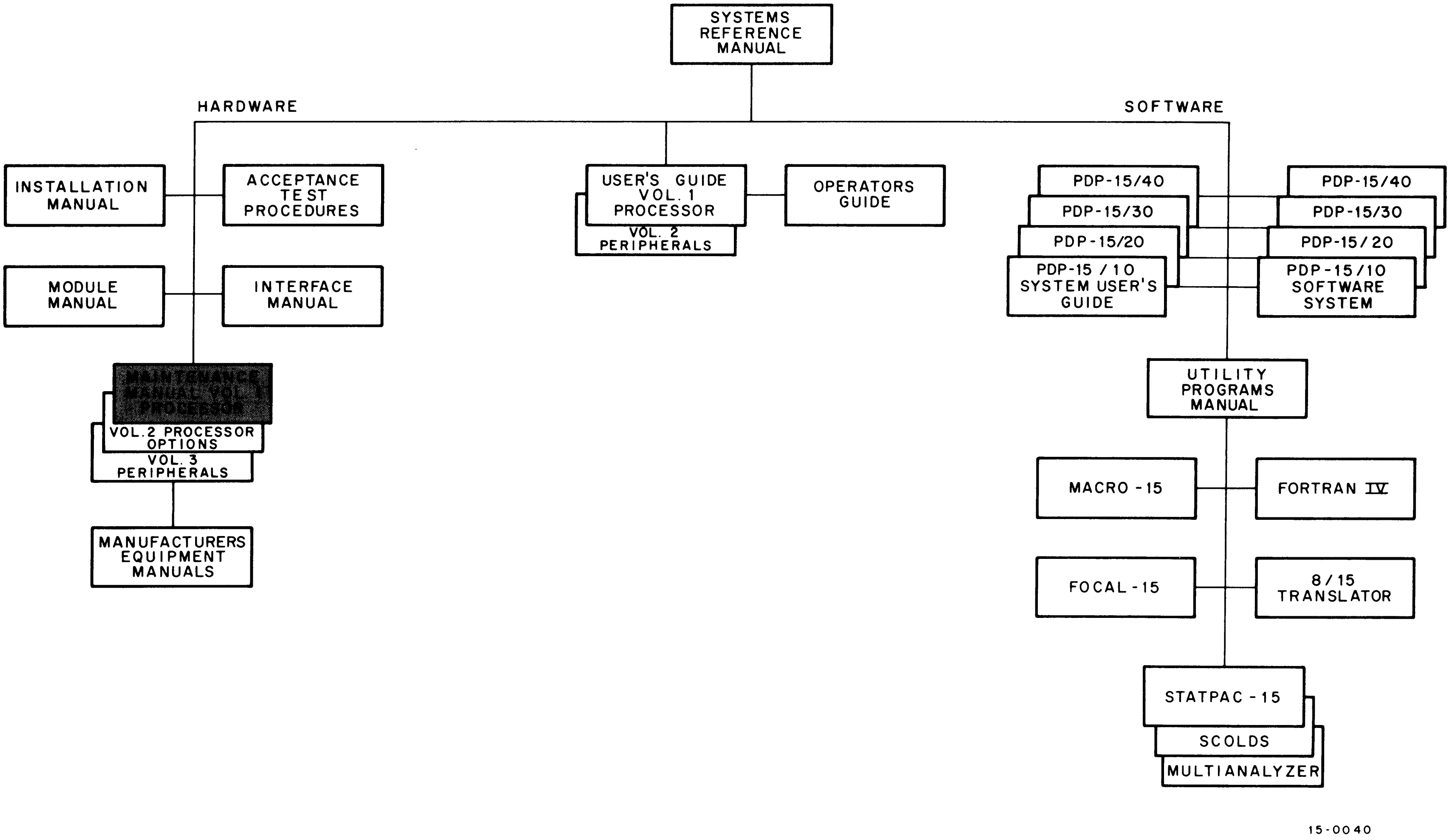
ILLUSTRATIONS (Cont)

ILLUSTRATIONS (Cont)

TABLES

TABLES (Cont)

PDP-15 FAMILY OF MANUALS



SYSTEMS REFERENCE MANUAL – Provides overview of PDP-15 hardware and software systems and options, instruction repertoire, expansion features, and descriptions of system peripherals. (DEC-15-BRZB-D)

USER'S HANDBOOK VOLUME 1, PROCESSOR – Principal guide to system hardware includes system and subsystem features, functional descriptions, machine-language programming considerations, instruction repertoire, and system expansion data. (DEC-15-H2DB-D)

VOLUME 2, PERIPHERALS – Features functional descriptions and programming considerations of peripheral devices. (DEC-15-H2DB-D)

OPERATOR'S GUIDE – Lists procedural data, including operator maintenance, for using the operator's console and the peripheral devices associated with PDP-15 Systems. (DEC-15-H2CB-D)

PDP-15/10 SYSTEM USER'S GUIDE – Features COMPACT and Basic I/O Monitor operating procedures. (DEC-15-GG1A-D)

PDP-15/20 SYSTEM USER'S GUIDE – Lists Advanced Monitor System operating procedures. (DEC-15-MG2B-D)

BACKGROUND/FOREGROUND MONITOR SYSTEM USER'S GUIDE – Lists operating procedures for the DECtape and disk-oriented Background/Foreground monitors. (DEC-15-MG3A-D)

PDP-15/10 SOFTWARE SYSTEM – Describes COMPACT software system and Basic I/O Monitor System. (DEC-15-GR1A-D)

PDP-15/20/30/40 ADVANCED MONITOR SOFTWARE SYSTEM – Describes Advanced Monitor System; programs include system monitor language, utility, and application types; operation, core organization, and input/output operations within the monitor environment are discussed. (DEC-15-MR2A-D)

PDP-15/30/40 BACKGROUND/FOREGROUND MONITOR SOFTWARE SYSTEM – Describes Background/Foreground Software System including the associated language, utility, and applications program. (DEC-15-MR3A-D)

RSX USER MANUAL – Describes the disk-oriented real time system executive language and applications.

MAINTENANCE MANUAL VOLUME 1, PROCESSOR – Provides block diagram and functional theory of operation of the processor logic; lists preventive and corrective maintenance data. (DEC-15-H2BB-D)

VOLUME 2, ENGINEERING DRAWINGS – Provides engineering drawings and signal glossary for the basic processor and options. (DEC-15-H2BB-D)

INSTALLATION MANUAL – Provides power specifications, environmental considerations, cabling, and other information pertinent to installing PDP-15 Systems. (DEC-15-H2AB-D)

ACCEPTANCE TEST PROCEDURES – Lists step-by-step procedures designed to insure optimum PDP-15 Systems operation.

PDP-15 MODULE MANUAL – Provides characteristics, specifications, timing and functional descriptions of modules used in PDP-15 Systems. (DEC-15-H2EA-D)

INTERFACE MANUAL – Provides information for interfacing devices to a PDP-15 System. (DEC-15-H0AB-D)

UTILITY PROGRAMS MANUAL – Provides utility programs common to PDP-15 Monitor systems. (DEC-15-YWZA-D)

MACRO-15 – Provides MACRO assembly language for the PDP-15. (DEC-15-AMZA-D)

FORTRAN IV – Describes PDP-15 version of the FORTRAN IV compiler language. (DEC-15-KFZA-D)

FOCAL-15 – Describes an algebraic interactive compiler level language developed by Digital Equipment Corporation. (DEC-15-KJZB-D)

CHAPTER 1
GENERAL DESCRIPTION

1.1 INTRODUCTION

This chapter identifies and describes the modular parts of the PDP-15 system hardware, and shows how they are incorporated into the system. Figure 1-1 shows the physical location of these parts in the PDP-15 system configuration. Figure 1-2 is a block diagram of the inter-connection of the elements of the system which are covered by this manual. Other options are covered by separate manuals.

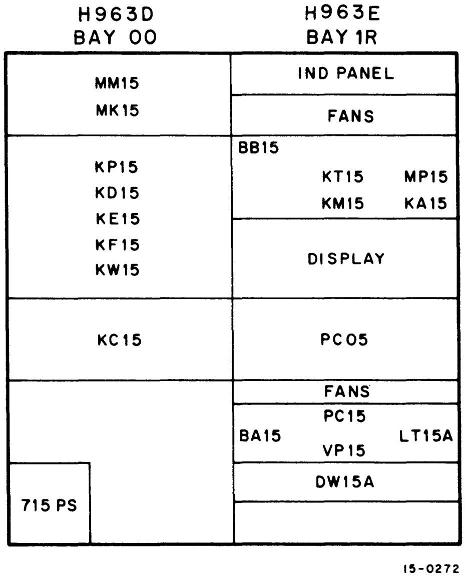


Figure 1-1 PDP-15 Configuration

1.2 SYSTEM DESCRIPTION

The PDP-15 System consists of a KP15 Central Processor, KD15 I/O Processor, KC15 Console, MM15 Memory, associated cabinets, hardware, power supplies, and any of a large number of options.

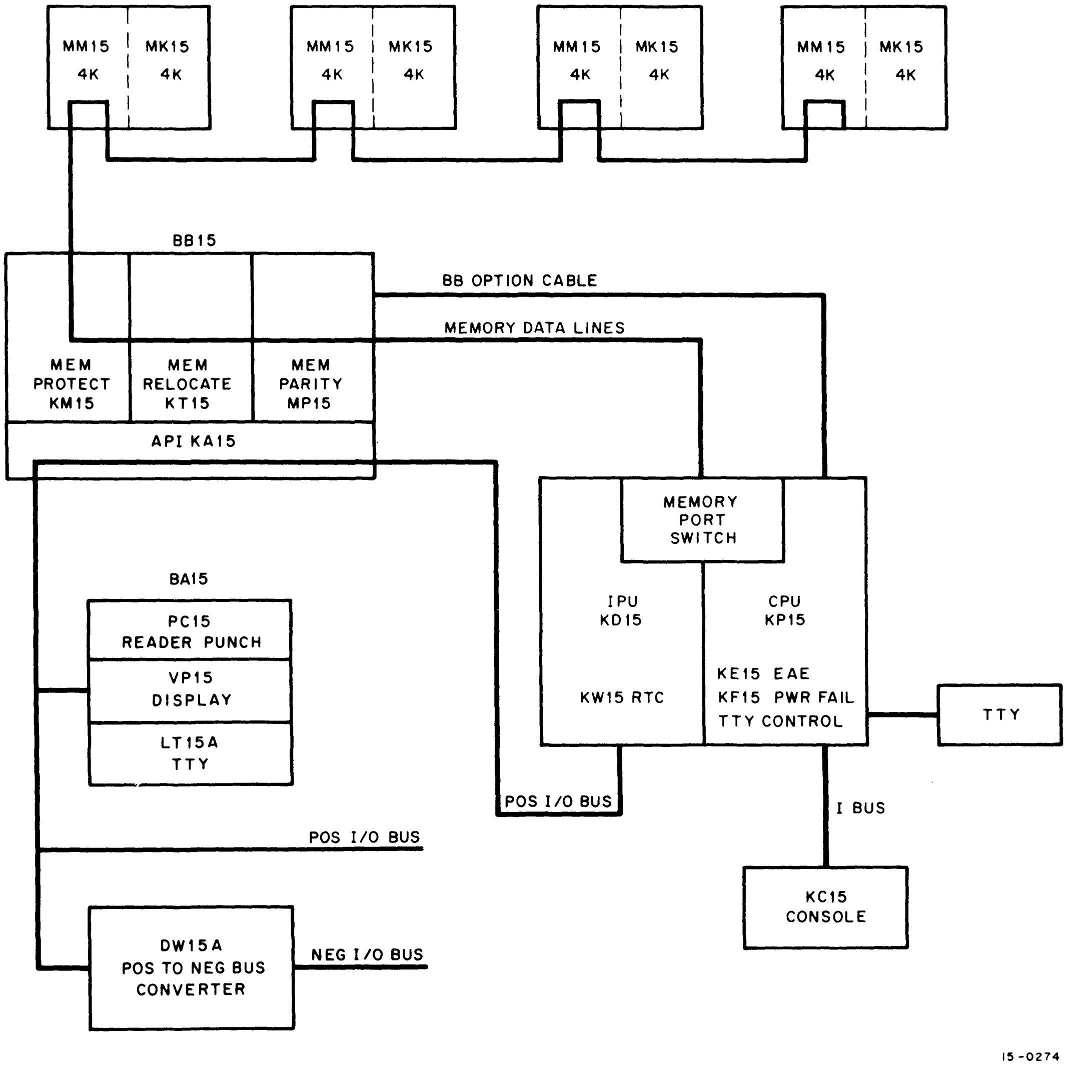


Figure 1-2 PDP-15 System Block Diagram

Hardware configurations for which special software systems have been developed are designated as follows:

a. PDP-15/10 Compact System — consists of a 4K core memory, Central Processor, I/O Processor, and ASR33 Teletype®. This is the basic PDP-15 System (see Figure 1-3).

b. PDP-15/20 Advanced Monitor System — consists of the basic PDP-15 with 8K of core memory, DECtape control and 2 DECtape transports, high-speed paper-tape reader/punch, extended arithmetic element, and KSR 35 Teletype (see Figure 1-4).

® Teletype is a registered trademark of Teletype Corporation.

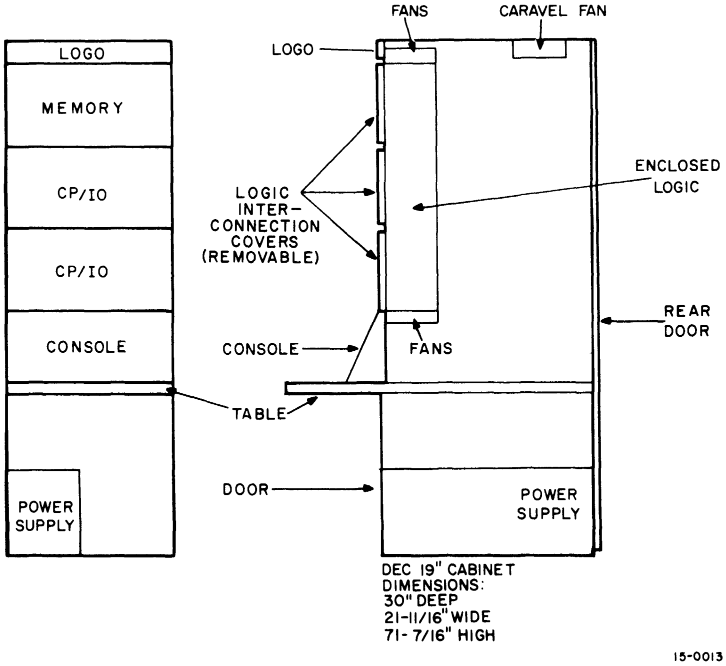
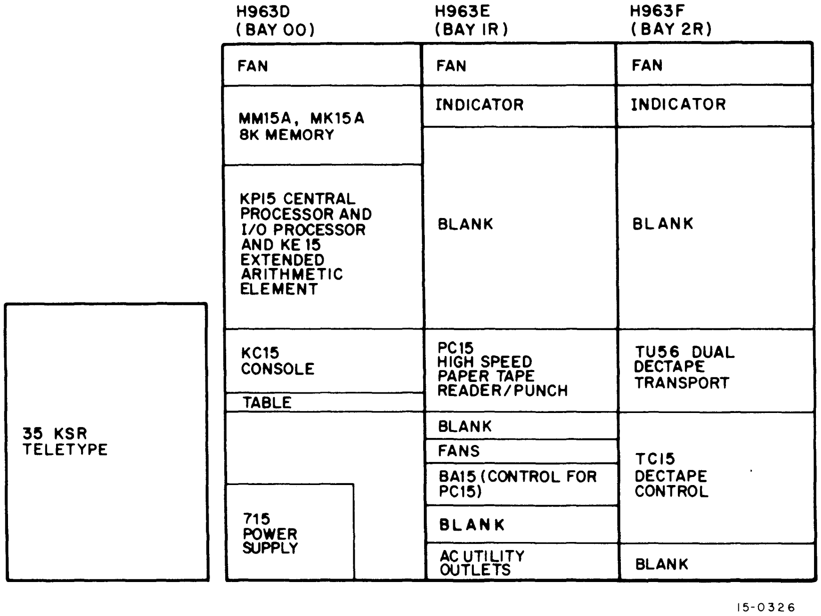


Figure 1-3 PDP-15/10 System Configuration Diagram



Fiqure 1-4 PDP-15/20 System Configuration Diagram

c. PDP-15/30 Background-Foreground System — consists of a PDP-15/20 with a 16K core memory, four DECtapes, a real-time clock, automatic priority interrupt, a second on-line Teletype and memory protect (see Figure 1-5).

d. PDP-15/35 Disk Oriented Real-Time Executive System (RSX) — consists of a PDP-15/20 with a 16K core memory, two DECtape transports, automatic priority interrupt option, real-time clock, a DECdisk file with 262K words of storage and memory protect.

e. PDP-15/40 Disk Oriented Background-Foreground System — consists of a PDP-15/30 with a 24K core memory, two DECtapes, DECdisk Control, two random access DECdisk files with a capacity of 524K words of storage and memory protect (see Figure 1-6).

1.3 CENTRAL PROCESSOR

1.3.1 KP15 Central Processor

The KP15 Central Processor functions as the main component of the computer by carrying on bi-directional communication with both the memory and I/O Processor. Provided with the capability to perform all required arithmetic and logical operations, the central processor controls and executes stored programs. It accomplishes this with an extensive complement of registers, control lines, and logic.

1.3.2 Teletype Control

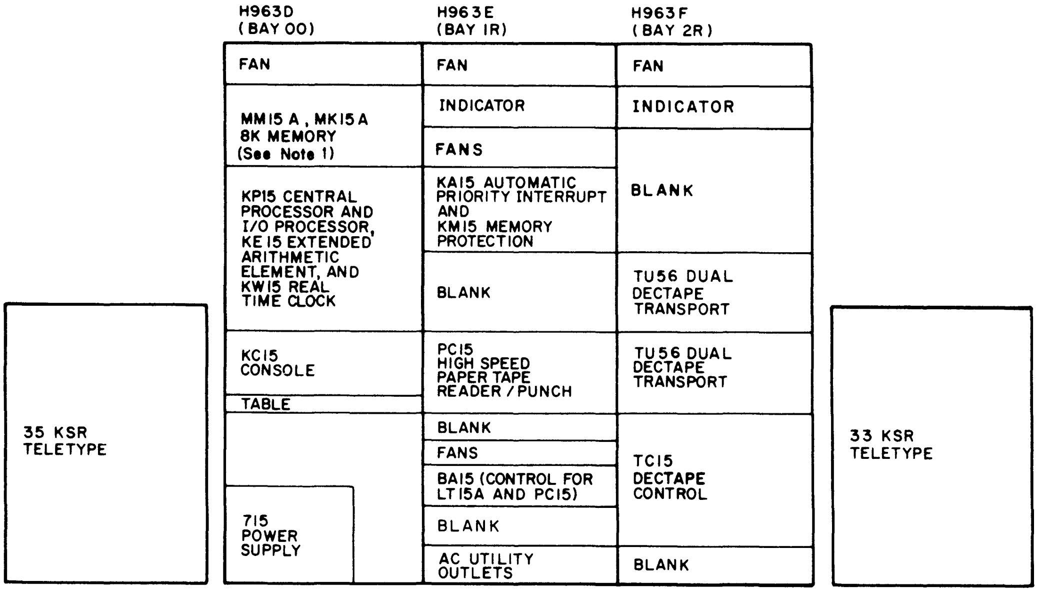
This provides the control for the console Teletype which may be an ASR or KSR Type 33 or 35 Teletype. Logic is provided for the hardware readin of tapes from the Teletype reader in systems without high-speed tape facilities. Characters may be read from the keyboard in either half-duplex mode (characters echoed onto the printer) or full-duplex mode.

1.3.3 KC15 Console

The KC15 Control Console provides facilities for operator initiation of programs, monitoring of central processor (CPU) and I/O Processor (IPU) registers, starting program execution and the manual examination and modification of memory contents.

1.3.4 KE15 Extended Arithmetic Element (optional)

The optional KE15 Extended Arithmetic Element (EAE) facilitates high-speed arithmetic operations and register manipulations. The EAE adds an 18-bit multiplier quotient register (MQ) to the system as well as a 6-bit step counter register (SC). Worst case multiplication time is 7.4 μs; division time is 7.65 μs.

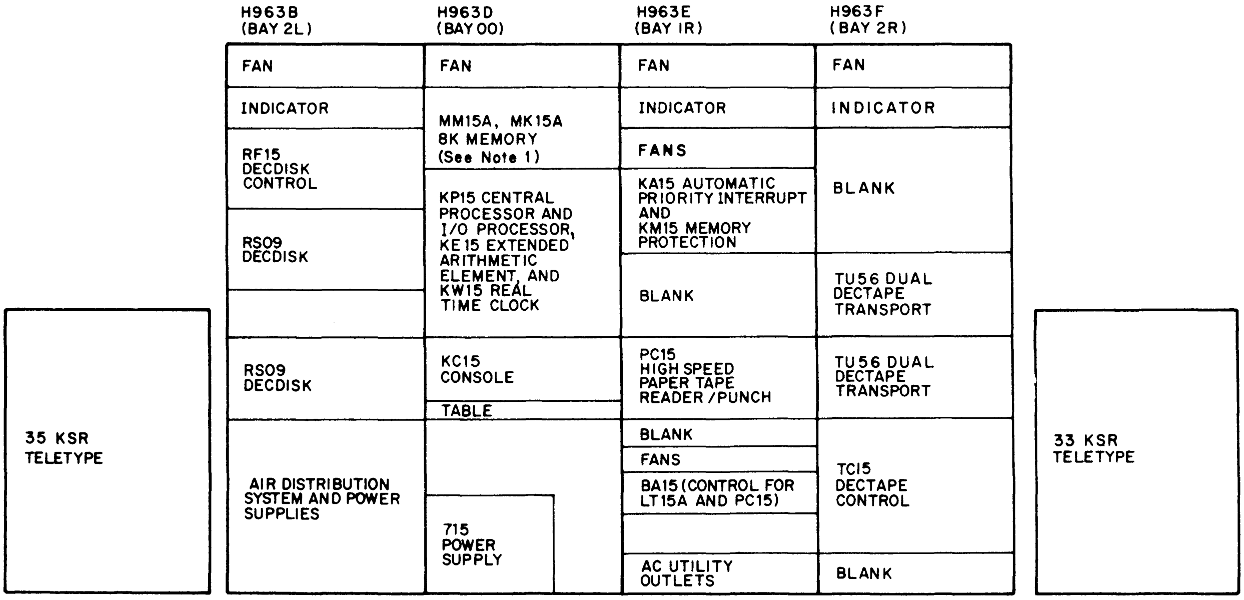


NOTES

1. An identical 8K memory is mounted on the rear door of cabinet H963D. There is space on this door for mounting two more 8K modules.

15-0327

Figure 1-5 PDP-15/30 System Configuration Diagram



NOTES

1. Two more 8K memory modules are mounted on the rear door of cabinet H963D. There is space on this door for mounting one more 8K module.

15-0328

Figure 1-6 PDP-15/40 System Configuration Diagram

1.3.5 KF15 Power Fail (optional)

The KF15 Power Fail option offers maximum protection to programs during power failure and recovery of power after failure. It enables the PDP-15 system to store active registers in memory before power diminishes to a point beyond which data will be lost. It also enables the PDP-15, upon the restoration of power, to restore these registers and continue with the program that was previously in progress.

1.4 I/O PROCESSOR

1.4.1 KD15 I/O Processor

The KD15 Processor (IPU) is an autonomous subsystem of PDP-15 which supervises and synchronizes all IOT and Data Channel (DCH) transfers between the devices and the PDP-15 central processor and memory. The I/O Processor contains the arithmetic and control logic hardware to supervise all I/O device activity. The IPU is, however, a passive system in that it responds to requests for activity from devices or the CPU rather than initiating activity.

1.4.2 KW15 Real-Time Clock (optional)

The KW15 Real-Time Clock option gives the user a time reference capability. The real-time clock produces clock pulses at a rate of 1 every 16.7 ms for 60 Hz systems; these systems increment a core location which can be preset and monitored under program control.

1.5 MEMORY

1.5.1 MM15 Memory

The MM15 Memory is the primary storage area for the computer instructions and data. Memory is organized into pages which are paired into memory banks. Each MM15 has 4K 18-bit binary words of high-speed random access magnetic core storage. Each bank is a unit of 8K words. The Central Processor and I/O Processor have provisions to address up to 128K words of core memory with the aid of a MX15 Memory Multiplexer. Any word in memory may be addressed by either the Central Processor or the I/O Processor.

1.5.2 MK15 4K Memory Expander (optional)

This option expands the MM15 memory control from 4K to 8K 18-bit words. It consists of the core memory stack and the necessary read/write circuits and must be used with an MM15.

1.5.3 MP15 Memory Parity (optional)

The MP15 Memory Parity option enables the PDP-15 to continuously check information being read from core memory and determine whether information has been erroneously picked up or dropped. It does this by monitoring all information as it is being sent to the memory for storage, and writing a parity bit. When this word is read from memory, the word and the bit are checked to determine whether any information has been changed, and a parity error flag is raised if a change is detected.

1.6 BA15 PERIPHERAL OPTION EXPANDER

This option houses power and I/O bus interface logic for the PC15, LT15A, and VP15 which are described below; only one BA15 is required per system.

1.6.1 PC15 High-Speed Paper-Tape Reader/Punch (optional)

This option includes a PC05 Reader/Punch and the control to interface it to the PDP-15. Characters can be read from paper tape at a maximum rate of 300 characters per second or punched at a rate of 50 characters per second. When this option is installed, the PC15, instead of the TTY, is used for hardware READIN.

1.6.2 LT15A Single Teletype Control (optional)

This option provides the logic to receive and transmit information through a KSR33 or KSR35 Teletype. This provides a second Teletype capability to the PDP-15 system.

1.6.3 VP15 Display Control (optional)

This option interfaces the PDP-15 to various display devices by providing the digital-to-analog converters and the control logic for the x and y positioning as well as the intensification of the VP15A Storage Tube Display, VP15B Oscilloscope Display, or VP15C X-Y Oscilloscope Display.

1.7 BB15 INTERNAL OPTION EXPANDER

This expander houses the power, I/O bus interface, memory bus interface and various control signals from the processor for the operation of the KA15 Automatic Priority Interrupt, KM15 Memory Protect, KT15 Memory Protect and Relocate, and MP15 Memory Parity options. Only one BB15 is required per system.

1.7.1 KA15 Automatic Priority Interrupt (API - optional)

The API option adds 8 additional levels of priority to the PDP-15. The upper 4 levels are assigned to I/O devices and are initiated by flags (interrupt requests) from these devices. The lower four levels are programming levels and are initiated by software requests. High data rate or critical devices are assigned to high priority levels and can interrupt slower device service routines. The API option holds the slower device interrupt request until it can be serviced. The cause of the interrupt is also directly identified eliminating the need for service routines and flag search routines.

1.7.2 KM15 Memory Protect (optional)

The KM15 memory protect provides the PDP-15 core memory with protected memory locations that cannot be accessed by the user. It includes a boundary register and associated control logic to establish the lower limit of the user's program. It has the facility to trap IOT, HALT OAS and chained execute instructions and the addressing of a non-existent memory bank.

1.7.3 KT15 Memory Protect and Relocate (optional)

The memory protect and relocate option is similar to the memory protect option. The KM15 must be used with the KT15. However, it contains a relocation register as well as a core allocation register. The relocation register provides the lower limit of the user program and relocates the user upward from the real machine location by the quantity contained in the relocation register. The core allocation register indicates the last 256 word increment available to the user. Other features of the memory protect option are also included in this option.

1.8 SYSTEM INTERCONNECTIONS

1.8.1 CPU and IPU-to-Memory Bus

The central processor and I/O processor each asynchronously access memory over the same memory data lines (MDL). The priority structure concerning which processor's request is sent to memory is determined by the memory port switch. The I/O processor is given first priority. The memory data lines consist of 18 bi-directional lines over which address and then data information is transmitted to and from memory. Various control signals are on this bus.

1.8.2 IPU-to-I/O Devices (I/O Bus)

The I/O processor communicates with all devices over a common I/O bus which contains bi-directional data lines; address lines; enable, request, and grant lines for API and data channel; and others such as program interrupt and skip request.

1.8.3 Console-to-CPU (I Bus)

The indicator bus contains bi-directional lines to transmit information to the lights on the console and switch information from the console to the central processor. Several control lines are also located on the cable.

1.8.4 BB Option Panel to Central Processor

Because the BB15 Option Panel contains several internal options which deal with the operation of the central processor, a special cable is required so that the option may utilize these internal central processor signals.

1.9 SYSTEM SPECIFICATIONS

Functional Characteristics

Word Length	18 bits
Cycle time	Refer to Table 1-1
Core memory capacity	4096 words, expandable to 131,072 words in 4K increments
Core memory access	
Page mode	
Direct	4096 words
Indirect	32,768 words
Indexed	131,072 words
Bank mode	
Direct	8,192 words
Indirect	32,768 words
Computation rate	625,000 additions per second
ASR33 Teletype	10 characters per second
Program-controlled I/O capacity	Up to 256 devices including prewired CPU and IPU IOTs
Data channel capacity	Up to 8 device controllers

Operating Characteristics (H963D Cabinet; CP, I/O and Memory)

Power requirements	115 ± 15%, 50 or 60 cps ± 2%, single phase, 18-30A or 230V ± 15%, 50 or 60 cps ± 2%, single phase, 18-30A
Power consumption	4 KW max
Power supply outputs after local regulation	+5, -6, -24 Vdc

Logic levels	0-.4V = logic 0	2.4-5V = logic 1
Test temperature range	50°-120°F	
Relative humidity	10-95%	
Heat dissipation	13,650 BTU/hr.	

Dimensions

Cabinet height	71-7/16 in.
Cabinet width	21-11/16 in.
Cabinet depth	30 in.
Shelf widths	19 in.
Shelf depth	19-5/16 in.
Door clearance (rear)	18-7/32 in.
Cabinet weight (loaded)	750 lbs.
Teletype height	8-3/8 in.
Teletype width	22 in.
Teletype depth	18-1/2 in.
Teletype weight	44 lbs.

Table 1-1
PDP-15 Central Processor Cycle Times, Basic and Expanded Configurations*

Configuration	Not In User Mode				In User Mode			
	Read		Write		Read		Write	
	Max	Typical	Max	Typical	Max	Typical	Max	Typical
Basic	800	800	800	800	800	800	800	800
KM15 Memory Protect	830	800	830	800	830	800	975	920
KM15 Memory Protect and KT15 Memory Protect/Relocate	965	880	965	880	1165	1080	1165	1080
MP15 Memory Parity	1100	1050	1100	1050	1100	1050	1100	1050
MP15 Memory Parity and KM15 Memory Protect	1130		1130		1130		1255	
MP15 Memory Parity, KM15 Memory Protect and KT15 Memory Protect/Relocate	1155		1155		1355		1355	

*All cycle times are listed in nanoseconds.

CHAPTER 2
MEMORY

2.1 INTRODUCTION

The basic PDP-15 computer has a 4096 word, 18-bit memory. It occupies the top four racks of the PDP-15 mainframe (racks A through D). The principal elements comprising rack A are control logic, bus drivers, and INPUT slots for the memory bus cables. The memory address and memory buffer registers, x- and y-select and driver controls, and OUTPUT slots for the memory bus cables comprise rack B. The stack matrix, sense amps and inhibit drivers, and slots for the indicator bus, make up racks C and D.

This basic memory may be expanded to 8192 words by adding a 4K stack, a set of X- and Y-select and driver modules, and a set of inhibit and sense amp modules. The appropriate slots are available in the MM15 and the addition is the option called MK15.

Each 4K unit of memory is called a page. Two 4K pages form an 8K bank. The physical capacity of the mainframe is 32K (32,768) words. One 8K bank of memory is located above the mainframe front panel. The three remaining 8K banks are located on the rear door. This 32K unit is called a block of memory.

2.2 CORE ADDRESSING MATRIX

2.2.1 Stack Dimensions

The MM15 memory stack is a three dimension-three wire matrix (3D-3 wire). The term three dimension refers to an X-current, a Y-current, and an inhibit current. The X- and Y-currents are used to switch the direction of the flux of a core; the inhibit current is used during the writing time to prevent the writing of bits where 0 bits are desired.

One matrix control network is used for each 4096 core words. The memory address selects which word of the 4096 words is to be read or written into or both. The drive current pulses are generated by G223 Read/Write Drivers. G222 Memory Selectors steer the direction of these currents.

When these currents flow through the core region they pass through 18 planes; the X-current pulse through one set of wires and Y-current pulse through another set of wires. In each plane, the X and Y current pulses intersect at one of the possible 4096 cores (bits). This happens in all 18 planes producing an 18-bit word selection. A nineteenth plane is added for parity options.

The inhibit current pulse occurs only during a write portion of the memory cycle. The inhibit current controls the bit data to be written into the address selected. This control is accomplished by stringing a wire through all the cores of each individual plane to nullify the effect of the Y current at the core intersection point. In addition to using these wires for the inhibit current during a write portion of a memory cycle, they are used for sensing the data during a read portion of a memory cycle.

2.2.2 X-Y Matrix Construction

Each of the 4096 word locations provided by a 64 x 64 X-Y matrix may be referenced by a 12-bit memory address. This memory address is separated into two sections: bits 06-11 for the Y-select and bits 12-17 for the X-select. It is decoded by the G222 modules. The total combination results in a 4096 word selection. The current pulse, generated from the G223s, provides the current needed to change the flux in the core. There is a G223 for each X and Y matrix.

Figure 2-1 shows the current path produced by the combination of the modules and stack. Although this figure represents only one matrix of either X or Y, both matrices run at the same time. The solid arrows show a read current path and the dashed arrows show a write current path through the same selected network. In this figure one can visualize four different core line paths by having pre-selected the gates. Block schematics MM06 through MM09 (Volume 2) show the X and Y matrices contained in a bank of memory. The I Loop is a jumpered wire on the back panel wiring for observation of current waveforms on an oscilloscope. There is a loop for each matrix.

2.2.3 Inhibit/Sensing Construction

The construction of the sense/inhibit network consists of one wire threaded through all the cores in each plane. The circuitry, for the sensing of a bit during a readtime and inhibiting during a writetime, is contained in the G100 Sense Amplifier and Inhibit Driver modules.

The G100 module contains four sense amplifiers and four inhibit drivers. Five of these modules are used in the PDP-15 for each 4K memory stack. (Refer to block schematics MM10 through MM15.) Each inhibit driver consists of a two-input NAND gate and a high-speed current switch. One driver is used for each bit plane of the memory array. An inhibit signal is received by all inhibit drivers only during a write operation (see Figure 2-2).

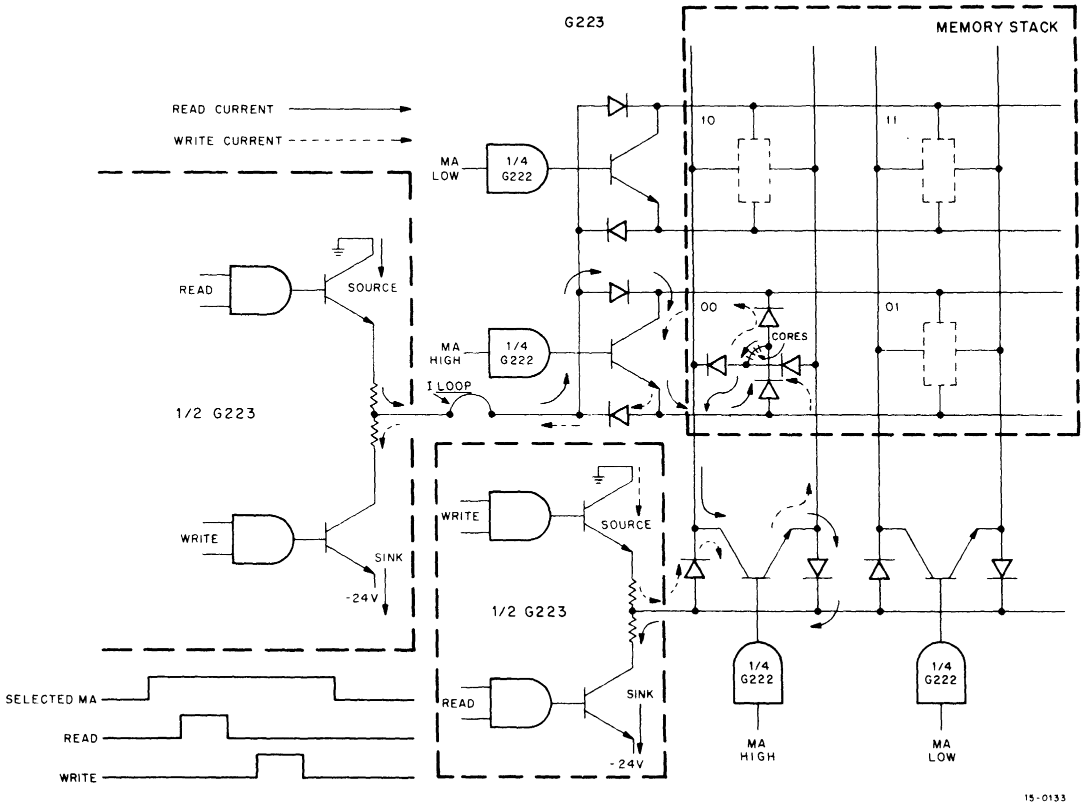


Figure 2-1 Current Path Diagram

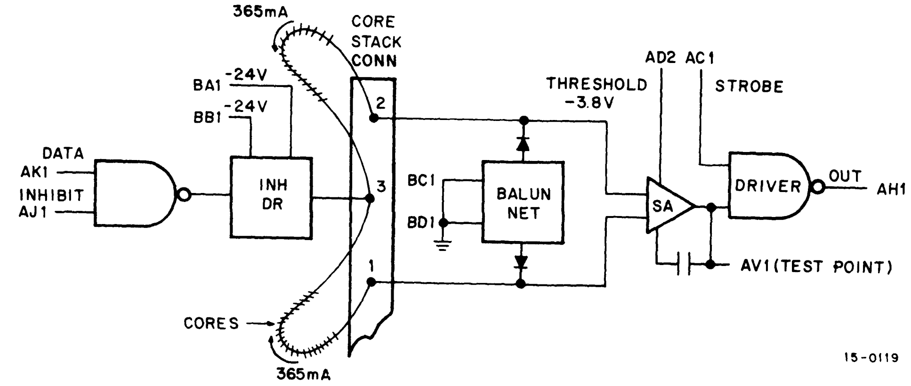


Figure 2-2 Sense Amplifier/Inhibit Driver, Simplified Diagram

Each driver also receives a signal indicating the state of the corresponding bit in the MB. Inhibit drivers that receive a signal indicating a 0 state in the MB bit are gated on and cause inhibit current to be applied to the associated bit plane of the memory array. Each inhibit driver employs a discharge network to speed up inhibit current cutoff. The output of the inhibit driver is connected to the middle of one core sensing string, which represents one bit plane of the memory array. The balun network at the front end of the sense amplifier ensures equal current at all times through both sides of the core string.

In addition to the balun network, the sense amplifier consists of a differential amplifier and output driver. One sense amplifier is used for each bit plane of the memory array. During a read operation only the signal induced on the sense winding of a core plane by a core-changing state is received by the differential amplifier. The differential amplifier has a nominal threshold of 17 mV. Output pulses of standard amplitude and duration are supplied by the output driver when the sense amplifier reads a logic 1 from the associated core, and is strobed by a standard positive going pulse at AC1. Propagation delay from the input to the sense amplifier to the buffered output is 25 ns (maximum) and from strobe input to buffered output is 15 ns (maximum). These output pulses directly set the MB register.

Typical waveforms of the inhibit current and the test point of the sense amplifier during a readtime are shown in Figure 2-3. The only way to look at an inhibit current is to place a current probe around one of the twisted pairs of wires coming out of the stack at the plug end.

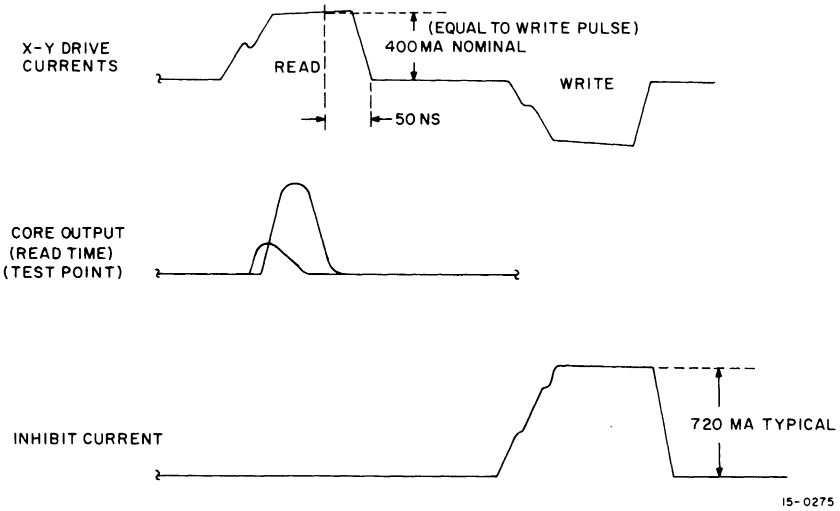


Figure 2-3 Typical Inhibit Current Waveforms

2.3 CONTROL LOGIC ARCHITECTURE

The control logic is designed so that each bank can be coupled to one bus network called the memory bus. Each bank responds when address lines 03 and 04 equal the setting of the manual switches on the bank select card. These two switches can be preset by the operator (refer to Volume 2, drawing MM01).

Maintenance hint: Any computer with more than one bank of memory may interchange one addressing bank number with another bank. All that is necessary is to manipulate the toggle switches on each bank of memory. No two should be set to the same number. Volume 2, drawing MM01 shows the logic representation of the switches.

An exchange of signals between the memory and the device requesting access to the memory provides asynchronous operation. The device has to provide five control signals, a 17-bit address (15 bits for 32K addressing and two more bits for the multiplexer option when addressing up to 128K), 18 bits of data, and one more bit for parity, when a parity option is included. In return the memory accepts the signals and notifies the device with a combination of four control signals and 18 bits of data plus one bit for parity checking. Table 2-1 identifies the bus signals and data lines on the memory bus. All signal lines are considered activated when at ground level. All data lines are considered a one when at ground level.

Table 2-1
Memory Bus Signals and Data Lines

Abbreviation	Signal & Direction	Definition
MDL00-17 (MA)	Memory Data Lines, 00 through 17 (Memory Address to memory)	Sends a seventeen bit word to memory, addressing up to 128K locations.
(DATA)	(Data bidirectional)	Data consisting of 18 bits per word going in both directions on the memory data lines (MDL).
MWR MRD	Memory Write Memory Read (to (to memory)	Two levels used by the memory to identify what mode of operation the device has selected. There are three: 1 Read/Restore (MRD ground) (MWR high) 2 Clear/Write (MRD high) (MWR ground) 3 Read/Pause/Write (MRD ground) (MWR ground)
M REQ	Memory Request (to memory)	The initial signal that begins the memory cycle.
M BUSY B	Memory Busy on the Bus (from Memory bank to other Memory banks)	This signal inhibits a memory request from being accepted by any memory bank on the same bus while any one of the memory banks is busy.

Table 2-1 (Cont)
Memory Bus Signals and Data Lines

Abbreviation	Signal & Direction	Definition
ADR ACK B	Address Acknowledge on the Bus (to device)	Notifies the device of acceptance of the Memory Request, the Memory Address, and the mode of operation (MRD and MWR signals).
RD RST B	Read/Restart on the bus (to device)	Notifies the device that the memory data is on the bus for the device to strobe into its buffer. This signal occurs only during a Read/Restore cycle or Read/Pause/Write cycle.
DATA ACK	Data Acknowledge (to memory)	Notifies the memory that it may take the memory data off the bus. This signal is only needed during a Read/Restore cycle or a Read/Pause/ Write cycle.
MRLS	Memory Release (to memory)	Notifies the memory it has accepted the data from memory during a Read/Restore cycle. During a Clear/Write cycle or a Read/Pause/Write cycle, this signal tells the memory the device has placed data on the bus.
MRLS ACK B	Memory Release Acknowledge on the Bus (to device)	Notifies the device that the memory accepted the data from the device and the memory cycle is terminating.
M PAR	Memory Parity (bi-directional)	This signal line carries a 19th bit of data to be stored into or readout of memory. This bit is used to check data errors when a memory parity option is implemented into the system.
Refer to Figure 2-4 for the device-memory control signal flow.		

Only one memory is allowed to be activated at a time. In order to guarantee that only one bank will be accessed, a Memory Busy level (refer to Volume 2, MM01 and MM19) is placed on the bus, inhibiting any other Memory Request from being accepted. The internal timing of this signal, as well as all other signals, is described in Paragraph 2.4.

2.4 CONTROL LOGIC FLOW

Refer to Figure 2-4. The memory is able to operate in any of three modes: Read/Restore, Clear/ Write, or Read/Pause/Write. The mode of operation to be selected depends entirely upon the needs of the device. This can be seen as the sequence of a memory cycle is defined.

Before memory can be started, Memory Busy must not be true and a 17-bit memory address (MDL 00-17) and MRD, MWR, or both must be activated on the bus.

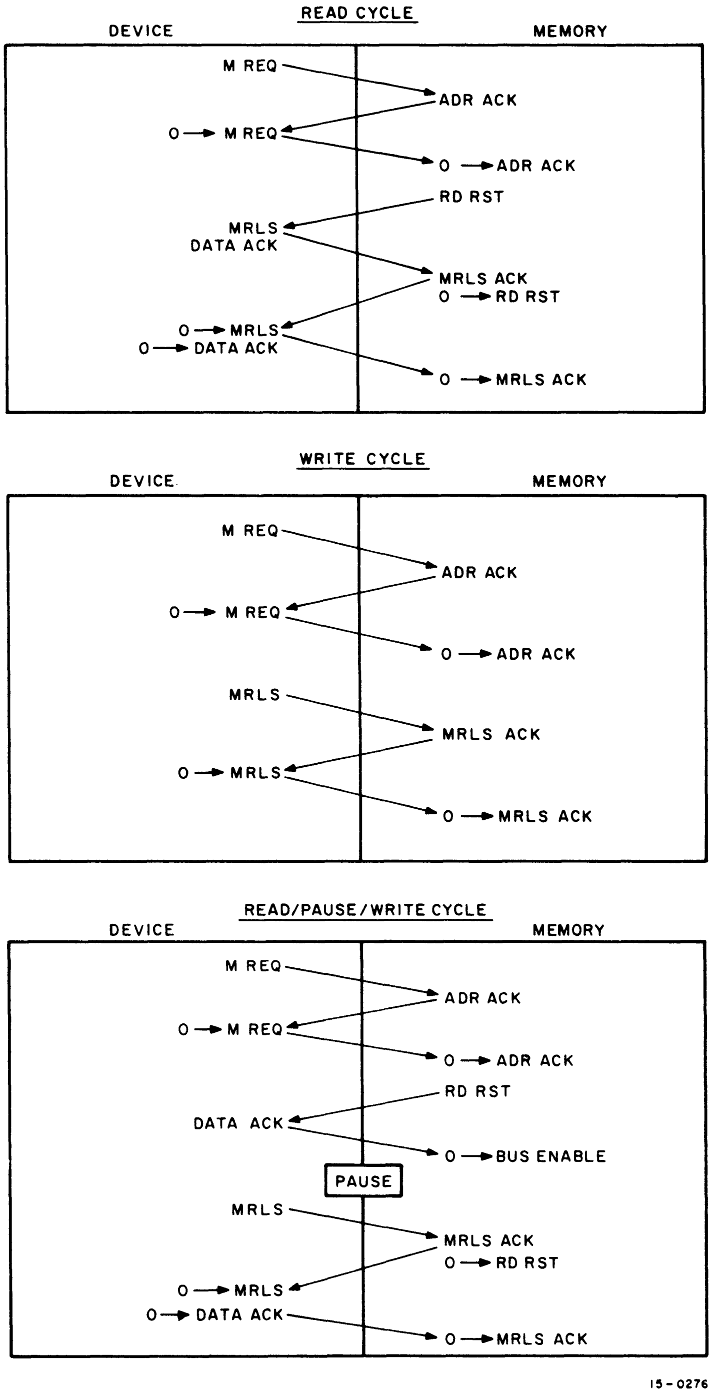


Figure 2-4 Basic Device-Memory Control Signal Flow

The memory cycle starts with the Memory Request level from the device. This level is generated 50 ns after the above levels to allow for settling time on the bus. (Settling time is needed to avoid potential skewing problems.) The device will wait until it receives an acceptance level from the memory called Address Acknowledge (ADR ACK (1) B L). This occurs approximately 110 ns after M REQ. The initiation of this level tells the device that the memory has accepted the request for access to memory, the mode of operation, and has strobed the memory address lines into the MA latch register. The Address Acknowledge causes Memory Busy to go low and also clears the Memory Request level. The fall of the Memory Request level clears the Address Acknowledge level. At this point the sequence of the memory cycle may go into any of the three modes of operation. Each is described in the following paragraphs.

2.4.1 Read/Restore Cycle

For this mode of operation the RD CON (read control) flop would have been set and the memory would proceed to read data from the address requested. The 18 bits of data read from core are strobed into the Memory Buffer register and enabled on the bus (MDL 00-17) approximately 150 ns after ADR ACK is set. A parity bit is also strobed, if the option is present (MDL PAR (1) B L). In order to allow for settling time on the bus, the device only strobes this data when a Read/Restart level (RD RST (1) B L) is placed on the bus by the memory, approximately 100 ns after the data is on the bus. The total data access time is approximately 350 ns.

After a delay, the memory sets the WR EN (write enable) flop which is ANDed with the WR CON (write control) flop on a zero state to begin the write portion of the cycle. The sequence is to rewrite the data from the memory buffer registers into the same location from which it was read. In parallel with the memory data rewrite, the device sends a level called Data Acknowledge (DATA ACK L) which the memory uses to clear the bus enable register, removing data from the bus. A Memory Release level (MRLS L) is sent at the same time as the Data Acknowledge level to tell the memory that the device has accepted the data. When the MRLS level has been received and the write enable flop has been set in the memory, the memory sends to the device an acknowledge called MRLS Acknowledge (MRLS ACK (1) B L). The device uses the MRLS ACK level to clear its MRLS level which, in turn, clears MRLS Acknowledge. (This sequence occurs in every mode of operation.) The memory cycle then terminates with the Memory Bus level (M Busy B L) going high.

2.4.2 Clear/Write Cycle

For this mode of operation the WR CON (write control) flop is set, and the memory proceeds to read the data from the address requested as is done in a Read/Restore cycle. The exceptions are that the data is not loaded into the Memory Buffer, and the bus enable and read/restart flops are not set. The function of this read is to clear the core location for the writing process. In parallel with this, the device accepts the Address Acknowledge level and places the data to be written into memory on the bus. To ensure settling time on the bus, a 50 ns delay occurs before the MRLS level is sent to memory. In the memory, an MB load pulse is generated with the MRLS level and the data is loaded into the Memory Buffer register. A Data Acknowledge level is not needed at this time with the MRLS level because it is only used to clear the bus enable register.

The MRLS level sets the MRLS ACK flop telling the device that the memory has accepted the data. The MRLS ACK flop ANDed with the Write Enable flop (detecting the termination of the read portion of this cycle) begins the write portion of this cycle, writing the data into the address requested.

During the write portion, a delayed sequence sets the inhibit flop and the write flop. The inhibit flop ANDed with the memory buffer bits produces inhibit currents that are applied to each individual bit plane, depending on the status of the memory buffer bit. A zero bit initiates the current pulse. The memory cycle then terminates with a Memory Busy level going high.

2.4.3 Read/Pause/Write Cycle

For this mode of operation both the read control and write control flops are set. The memory proceeds to read the data from the address requested as in a Read/Restore cycle. When the Read/Restart level is sent out to the device, the device strobes the data into its registers and proceeds to modify the data. The device at this time sends to memory a Data Acknowledge level which in turn clears the bus enable flop taking the memory data off the bus. The MRLS level is not sent at this time. The write enable flop does not initiate a write portion of the cycle. Instead the memory cycle halts. This is called the Pause portion of the Read/Pause/Write cycle.

While the memory is in the Pause state, the device modifies the data it has previously read from the memory and places it back on the bus. In order to provide settling time on the bus, the MRLS level is sent to the memory approximately 50 ns after the data is on the bus. When the memory receives MRLS, the modified data is strobed into the memory buffer registers. The write portion of this cycle begins with the MRLS ACK flop and the Write enable flop set. The data is written into the same address as requested at the beginning of this cycle. A Memory Address Hold flop ensures the address will be saved throughout the cycle. The memory cycle then terminates with a Memory Busy level going high.

The advantage of a Read/Pause/Write cycle is that a memory word may be read, modified, and rewritten in one memory cycle. An example of this is when a device wishes to update a current address location by adding one and restoring it. The flow chart in Figure 2-5 shows these same sequences with a little more attention to the internal logic flow.

Table 2-2 describes control flop and register nomenclature and major responsibilities. It is an aid in understanding the detailed flow chart shown in Figure 2-5.

Table 2-2
Control Flops and Register Responsibilities

Abbreviation	Control Flops or Register	Responsibility
RD Con WR Con	Read Control Write Control	Strobed when a request is accepted. Holds data throughout memory cycle. Tells memory which one of the three modes to pursue.
MA HOLD	Memory Address Hold	Cleared and set when a request is accepted. It holds the memory address information in the latches throughout the memory cycle.
MA00 through MA17	Memory Address 00 through 17 (bus address to memory direction only)	A set of 18 latches which contain the address to be accessed during the memory cycle.
ADR ACK	Address Acknowledge (bus signal)	This control flop tells the device that it has accepted the request for memory use and strobed in the address. It will clear itself with the loss of the Memory Request on the bus.
RDY	Read Y-matrix	This control flop turns on the y-matrix currents during a read portion of a memory cycle. It automatically clears after approximately 200 ns.
RDX	Read X-matrix	This control flop turns on the x-matrix currents during a read portion of a memory cycle. It sets 50 ns after the RDY register to minimize stack noise. It clears at the same time the RDY register clears.
MB00 through MB17 PAR MB	Memory Buffer 00 through 17 Parity Memory Buffer (bus data, bidirectional)	A set of 18 buffers plus a parity register are cleared when a request has been accepted. During a Read/Restore cycle they are directly set from the sense amplifiers. During a Clear/Write cycle they are strobed, accepting data from the memory bus.
BUS EN	Bus Enable	This control flop enables the data in the memory buffers, that was read out of memory, onto the memory bus. A Data Acknowledge from a device clears it.
RD RST	Read/Restart (bus signal)	This control flop is used to tell the device that data from memory is stable on the memory bus. It sets only during a readout cycle and clears with a Memory Release Acknowledge.

Table 2-2 (Cont)
Control Flops and Register Responsibilities

Abbreviation	Control Flops or Register	Responsibility
WR EN	Write Enable	This control flop is used to set up a definite delay between the readout of memory and the rewrite back into memory on a Read/Restore cycle. It also sets up a network to accept a Memory Release from the device.
MRLS ACK EN	Memory Release Acknowledge Enable	This control flop ensures a delay time before a Memory Release is accepted by the memory. This delay is important during a Clear/Write cycle. It sets with the Write Enable register setting, and clears on the next Memory Address Hold register clear in the next cycle.
MRLS ACK	Memory Release Acknowledge (bus signal)	This control flop is used to tell the device that it has accepted the devices Memory Release. It will clear itself with the loss of the Memory Release on the bus.
INH	Inhibit	This control flop enables all the inhibit drivers only during a write portion of a memory cycle. The on-time is approximately 200 ns.
WR	Write	Not being concerned with stack noise at this time, this register turns on both the x- and y-matrices only during the write portion of the memory cycle. The on-time is approximatėly 175 ns. It is delayed 25 ns after the Inhibit register to allow for the slower rise time of the inhibit current to skew up with the Write.
BUS DONE	Bus Done	This control flop sets with a setting of the Memory Release Acknowledge register. It denotes the extent of time that the memory bus is busy. The next setting of the Read X register will clear it.
WR DONE	Write Done	This control flop is used to set up a definite delay between the write portion of the memory cycle and the read portion of the next memory cycle. It sets with the termination of the write and clears on the next setting of the Read X register.

Table 2-2 (Cont)
Control Flops and Register Responsibilities

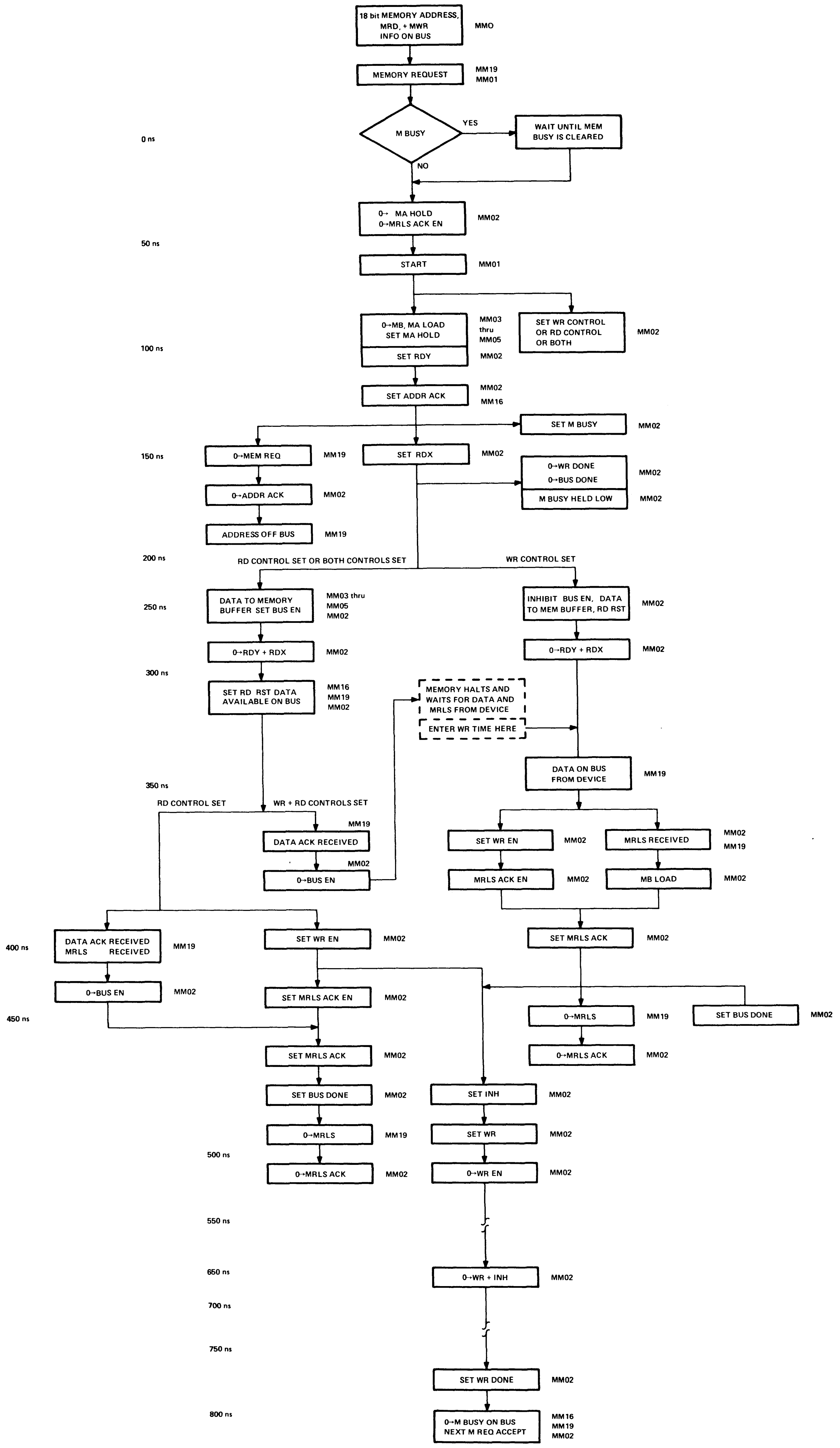
Abbreviation	Control Flops or Register	Responsibility
M BUSY	Memory Busy (bus signal)	This signal, not being a control flop directly, is a combination of three control flops to tell the device that the memory bus is busy and any memory tied on this bus will not accept a Memory Request until the presently active memory completes its cycle. The three control flops that control this bus signal are Address Acknowledge, Bus Done, and Write Done. Address Acknowledge is the first control flop to bring the bus signal low, then when Read X clears the other two control flops they continue the low signal. The bus signal will then only come up when both Bus Done and Write Done control flops are set.

2.5 MEMORY PORT SWITCH

The memory port switch (KP26) establishes the priority between the CPU and IPU. The IPU is given the higher priority to minimize latency and prevent data loss to devices on the I/O bus. The port switch also provides the synchronizers necessary for switching between the IPU and CPU. De-skewing is provided by the port switch such that the address is put on the memory data lines 50 to 60 ns before memory request, and data is put on the lines 50 to 60 ns before memory release. The port switch also contains the drivers that send control signals to memory.

The memory interface control (KP32), which is an integral part of the port switch, generates the CPU control signals for memory and control signals for the CPU. The first stage of CPU synchronization CP MEM REQ HOLD is also shown on KP32.

When the CP is running and the IPU is inactive, the MPX flop is set. As long as MPX is set, I/O ACT cannot be set, and CP ACT may be set. Each time the CP needs memory, the CP MEM REQ HOLD flop is set. This occurs during TS01 PHASE 2. CP ACT is set during TS02 PHASE 0. Each event in the sequence is typically separated by one HS CLOCK or 65 ns. The HS CLOCK is used to provide the necessary de-skewing of address and request. If however, the IPU needs memory, it raises a sync (DCH SYNC or CLK SYNC) which will cause MPX to be reset. MPX (0) inhibits CP ACT from being set again. When the CP completes its present memory cycle, it is then prevented from asking for memory again until the IPU completes its use of memory. MPX is reset before the IPU is ready to



BUS SIGNAL SEQUENCE

DIRECTION FROM	RD CYCLE	WR CYCLE	RD PAUSE WR CYCLE
DEVICE	MA + M REQ + RD	MA + M REQ + WR	MA + MREQ + RD + WR
MEMORY	ADDR ACK	ADDR ACK	ADDR ACK
MEMORY	M BUSY LOW	M BUSY LOW	M BUSY LOW
MEMORY	RD RST		RD RST
MEMORY	DATA ON BUS		DATA ON BUS
DEVICE	DATA ACK. + MRLS		DATA ACK
DEVICE		DATA ON BUS	DATA ON BUS (MODIFIED)
DEVICE		MRLS (MB STROBE)	MRLS (MB STROBE)
MEMORY	MRLS ACK	MRLS ACK	MRLS ACK
MEMORY	M BUSY HIGH	M BUSY HIGH	M BUSY HIGH
	M BUSY LOW	M BUSY LOW	M BUSY LOW

15-0311

Figure 2-5 Memory Cycle Detailed Flow Chart

use memory to provide a minimum wait when it is finally ready. When it is ready, I/O MREQ is set, then I/O ACT, and then MREQ. The IPU also uses the high-speed clock of the CPU to provide deskewing between address and request.

The control signals to memory have their drivers shown on print KP26. Each control signal may be produced by either the CPU or the IPU. The control signals are MREQ, MRLS, DATA ACK, MRD, and MWR. Read cycle and write cycle flow charts are shown in Figures 2-6 and 2-7.

As previously explained, MRD and/or MWR as well as the address must be on the memory bus 50 to 60 ns before MREQ. On print KP32, Memory Interface Control, the CPU generates START READ and START WRITE. START READ is generated any time the CPU needs to read information from memory. This will occur no later than the end of TS01 PHASE 2. It may occur as early as TS01 PHASE 0. START WRITE is generated by the CPU any time a write into memory is to be performed. The CPU will never raise MREQ with both START READ and START WRITE since the CPU is not capable of performing read/pause/write cycles.

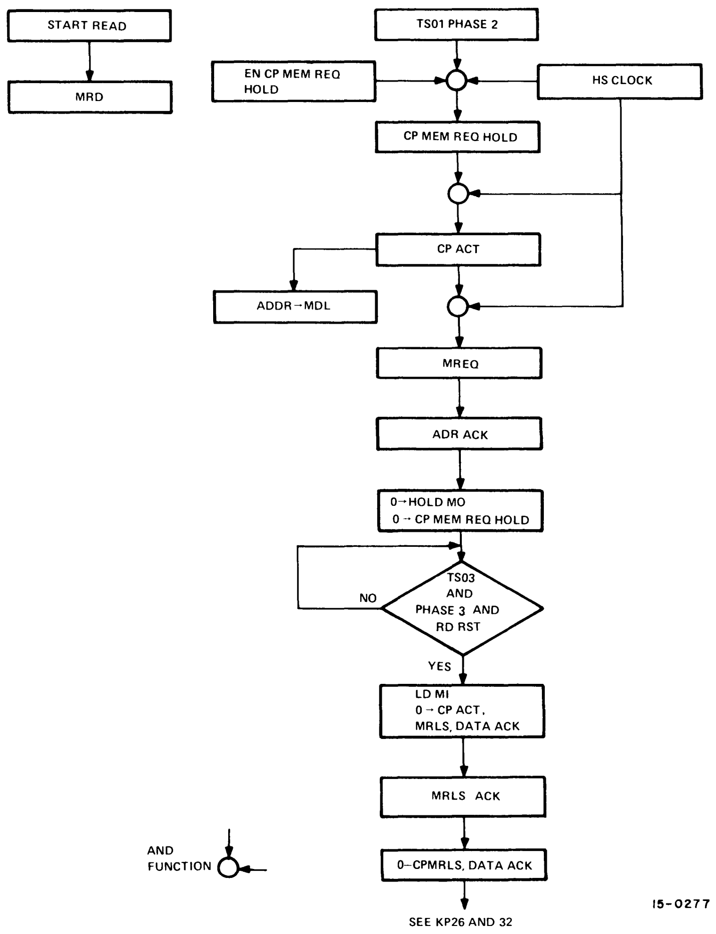


Figure 2-6 Central Processor Read Cycle

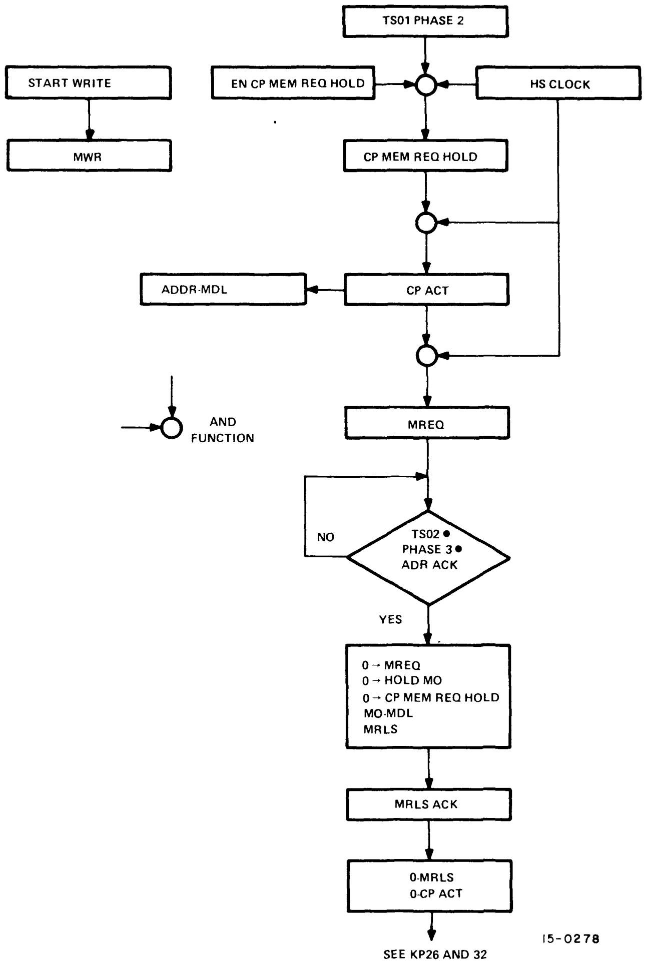


Figure 2-7 Central Processor Write Cycle

For CP cycles, after MREQ is issued, the memory replies with ADR ACK. ADR ACK is issued to remove the address from the MDL and also clear MREQ. In the case of write cycles, it is used to produce MRLS at the TS02 PHASE 3 time. On read cycles, the next response from memory is RD RST. This signifies that data is present on the MDL's. The CP cannot use the data until TS03 PHASE 3, so the data is not loaded into the MI register until that time. At the same time that data is loaded into the MI, END OF CP CYCLE, MRLS, and DATA ACK are issued signifying theCP has completed its use of memory.

For I/O cycles, control signals from memory are gated with I/O ACT to determine that they are meant for the IPU. The chain of events is similar to that of the CP with one exception. The CP issues DATA ACK and MRLS simultaneously. The IPU issues DATA ACK first to remove memory data from the lines. Then, when the IPU has placed data on the lines to be written into memory, it issues MRLS.

CHAPTER 3
CENTRAL PROCESSOR

3.1 REGISTERS

This section describes the internal CPU registers of the PDP-15. Most of these are 18 bits long; they appear in prints KP01 through KP18, one bit of each register per page. The exceptions are the 6-bit Instruction Register on KP31 and the 1-bit Link on KP22. The buses mentioned below are described in Paragraph 3.2. Refer to the block diagram shown on drawing KP70.

3.1.1 Accumulator (AC)

This is the main data register in the CPU; most instructions reference and/or modify the accumulator. It is loaded from the D (shift) bus by the LD AC strobe (KP24).

3.1.2 Link (L)

Essentially a high-order extension of the AC, the Link is also strobed by LD AC. Its input circuitry (KP22) is similar to the AC shift bus, but is more complex because of its arithmetic functions: it contains the carry from the high order bit after a TAD, and indicates an arithmetic overflow if set after an ADD. The Link may also be cleared, complemented, and tested by OPR instructions. It is also used by the extended arithmetic element (EAE) option.

3.1.3 Program Counter (PC)

The Program Counter contains the address in core of the next instruction to be executed by the PDP-15. It is loaded from the SUM bus by the LD PC strobes (KP24). To accommodate the various addressing modes (see Paragraph 3.5), the PC bits are broken into three groups which are strobed separately. It is possible to load bits 6-17, bits 5-17, or all bits of the PC.

3.1.4 Memory Input Register (MI)

All words read from memory into the CPU first enter the MI. The LD MI strobe (see KP32) causes the Memory Data Lines to be read into the MI. This register is made up of clocked set-reset flops for speed of operation.

3.1.5 Memory Output Register (MO)

This register holds all information going from the CPU to the Memory Data Lines. This includes all addresses, as well as data for write cycles. The MO is loaded by LD MO from the SUM bus. The PC may be loaded directly into the MO by the signal JAM PC to MO (see KP23, KP24). This causes a direct set of all MO bits, then clears all MO bits whose corresponding PC bits are 0.

3.1.6 Operand Address Register (OA)

The OA is used for temporary storage of the operand address, computed for all memory reference instructions. It is loaded from the SUM bus by the LD OA strobe (see KP24).

3.1.7 Instruction Register (IR)

This is a 6-bit register used to hold the 4-bit instruction code, the indirect bit, and the index bit (see KP31). It is loaded from MI bits 0-5 by the load IR strobe (see KP31). Some of the more time-critical instruction decoding is done directly from the MI, but most is done from the IR after it is loaded (see KP28, KP29, and KP30).

3.1.8 Index Register (XR)

This register holds a quantity which is added in to the operand address of indexed instructions. It is loaded from the SHIFT bus by LD XR (see KP24, KP29).

3.1.9 Limit Register

This register holds a quantity which can be tested against the XR by an AXS instruction. It is loaded from the SHIFT bus by the LD LR strobe (see KP24, KP29).

3.2 BUS STRUCTURE

Most of the internal data routine of the CPU is handled by five internal buses, designated A, B, C, D (or SHIFT), and Sum. Like the registers, these appear on prints KP01-KP18, one bit of each per page (see Figure 3-1).

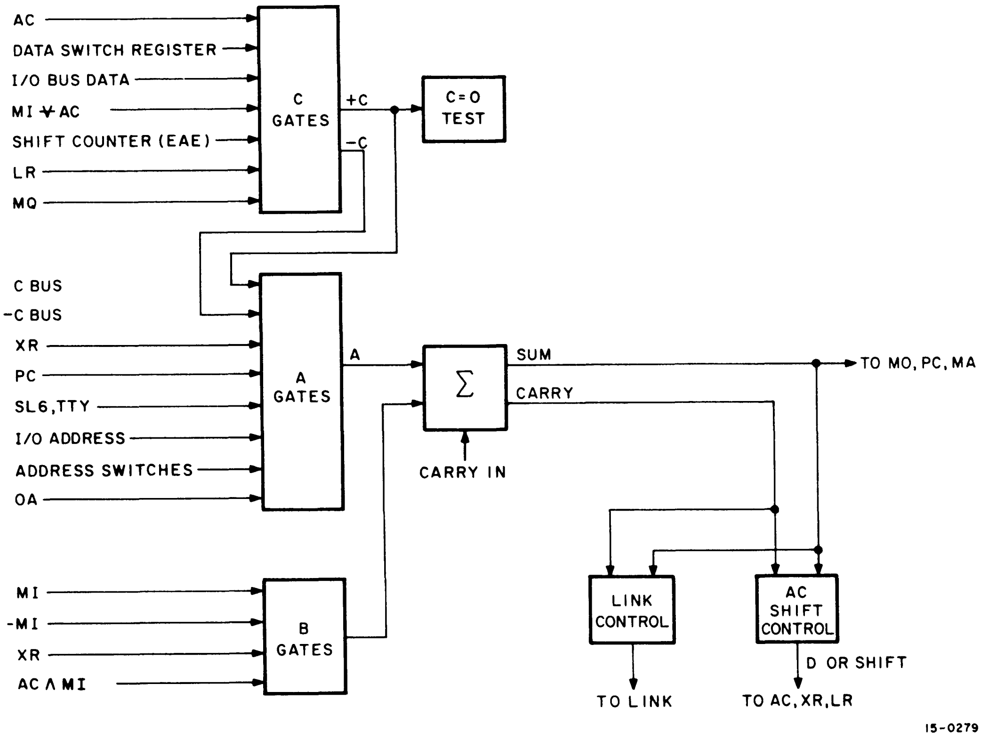


Figure 3-1 CPU Bus Structure

3.2.1 C Bus

This bus carries one of seven possible signals, according to the enabling level it receives:

a. The AC (buffered, called BAC), enabled by the BAC-C signal (see KP19).

b. The DS register (data switches, see Chapter 8), gated by DS-C (see KP19).

c. The I/O bus data lines, gated by I/O BUS - C (see KP19).

d. The exclusive OR of the MI and the AC, gated by XOR-C (see KP19).

e. The Step Counter (from the EAE, print KE03) gated by SC-C (see KP19, KE04).

f. The MQ (from the EAE, print KE01, KE02) gated by MQ-C (see KP19, KE04).

g. The LR, gated by LR-C (see KP19, KP29).

3.2.2 A Bus

The A bus is one of the inputs to the adder, and carries one of eight possible signals:

a. The C bus, gated onto the A bus by C-A (see KP19)

b. The complement of the C bus, gated by -C-A (see KP19)

c. The XR, gated by the XR-A signals (see KP19). These are split into XR-A 0-2, XR-A 3-5, and XR-A 6-17 to accommodate the various address modes (see Paragraph 3.5).

d. The PC, gated by the PC-A signals (see KP19). These signals, too, are divided into groups of bits to accommodate the various address modes. The groups are 1-2, 3-4, 5, and 6-17. Bit 0 is never enabled.

   The signal L BM UM-A causes the link, the bank mode flop, and the user mode flop to be placed on the high-order three bits of the A bus, and the PC on bits 3-17. This is used by the JMS and CAL instructions to store the processor status.

e. The AC, left-shifted 6 places, with TT DATA lines (from KP66) filling in the six low order bits, gated onto the A bus by SL6-A (see KP19). This is used during TTY hardware readin.

f. The I/O ADDRESS lines, gated by IO ADD-A (see KP19).

g. The console ADDRESS SWITCH signals (see Paragraph 3.10) gated by ADDR SW-A (see KP19).

h. The OA gated by OA-A.

3.2.3 B Bus

This bus is the second input to the adder and carries one of four signals, depending on the enabling level received:

a. The MI, gated by the MI-B signals (see KP47). To allow for the various addressing modes, the MI-B signal is split to allow gating of groups of bits, as follows: 0-2, 3-4, 5, 6-8, and 9-17.

b. The MI complemented, gated by the -MI-B signals (see KP19). These are split into -MI-B 0-8 and -MI-B 9-17.

c. The XR, gated by the XR-B signals (see KP19). These are split into XR-B 0-2, 3-5, and 6-17.

d. The logical AND of the MI and the AC (buffered), gated by AND-B (see KP19).

3.2.4 Sum Bus

The Sum bus is the output of the 18-bit adder. It carries the sum of the A bus and the B bus, plus 1 if the low order CARRY INSERT of the adder is high. Often, only one of the buses will have an enable signal, the other bus being 0.

3.2.5 Shift Bus (D Bus)

Although the Shift bus is used as input to the XR and LR as well as the AC, its primary purpose is to implement the various AC rotates and shifts. This is done by gating various shifted positions of the AC onto the bus, then strobing the bus back into the AC. The Link input circuitry contains gating corresponding to the Shift bus gating. The eight Shift bus signals and their enabling levels are as follows:

a. Sum bus unshifted, gated by NO SHIFT-D (see KP19). This is used for all unshifted transfers into the XR, LR, and AC.

b. AC with halves swapped (bits 0-8 swapped with bits 9-17), gated by SW-D (see KP19).

c. Sum bus, rotated 1 left (including Link), gated by RAL-D (see KP19).

d. Sum bus, rotated 1 right (including Link), gated by RAR-D (see KP19).

e. Sum bus, rotated 2 left, (including Link), gated by RTL-D (see KP19).

f. Sum bus, rotated 2 right (including Link), gated by RTR-D (see KP19).

g. Sum bus, shifted 1 left, with AC00 entering Link and MQ00 entering AC17, gated by DIVSHIFT-D (see EAE).

h. Sum bus, shifted 1 right, with Link entering AC00, gated by MULSHIFT-D (see EAE).

3.3 DATA MANIPULATION HARDWARE

The bus structure described in Paragraph 3.2 not only transports data, but performs the logical AND, XOR, and complement operations and all shifts. The adder performs the addition of the A and B bus. Normally this is a simple binary add, compatible with two's complement representation for negative numbers, with the high order carry placed in the Link. The adder is also used for all incrementing, by forcing a low order carry insert (see KP48). One's complement add uses the same adder circuits, but any high order carry must be re-inserted at the low end, and the adder must be allowed to settle a second time. The link is used to indicate an arithmetic overflow, in which both operands are of the same sign and the result has the opposite sign (see KP22).

Internally, the adder uses a conditional sum technique for speed. The 18-bit adder is divided into three groups of six bits each. Within each group, the addition is performed by two separate 6-bit adders. One adder assumes a carry in, and the other assumes no carry in. The low order adder requires about 48 ns to settle, but the second and third groups can operate much faster; when they receive the carry they gate the proper sum, already calculated, to the output. Thus, total adder settling time is 82 ns.

Comparison and testing of data is handled in several ways. A high-order carry from the adder, while incrementing, indicates that the ISZ skip should be taken (see KP23). The zero tests for SAD and OPR are performed by a C-bus zero test circuit (see KP33). The test portion of the AXS instruction is carried out by subtracting the LR from the XR (add, with LR complemented and carry insert high) and skipping if the result is positive (see KP29).

3.4 CONTROL STATE GENERATION

At any given time, the CPU may be in one of five major states: Fetch, Defer, Execute, Increment, or EAE. These states each last for about 780 ns and correspond to the approximate time required for one complete memory cycle. From one to five of these states are used in the execution of an instruction, the exact number and sequence depending upon the instruction type. See Paragraphs 3.7, 3.8 and 3.9 for more information on this subject.

Each major state is made up of three 260-ns time states, designated TS01, TS02, and TS03. The only exception to this occurs during the Execute state of the ADD (one's complement) instruction. In this single instance, an extra time state (designated TS02A) is inserted between time states TS02 and TS03, thus stretching the Execute cycle to 1040 ns. This allows extra time to perform the end-around carry as described in Paragraph 3.3.

Each of the 260 ns time states is further divided into four 65 ns phases, numbered 0-3. Each phase corresponds to a complete cycle of a 65 ns pulse generator called the high-speed clock.

The circuitry to produce the major states is shown in KP20; the state, phase, and clock logic are shown in KP21; a general timing diagram appears in KP79. Each major state, time state, and phase is represented by a flip-flop; only one flop in each of the three groups is set at once. The phase is changed by the falling edge of the HS CLOCK; the time state changes on the phase 3-phase 0 transition; the major state changes on the time state TS03-time state TS01 transition. A 30-ns pulse called CLOCK is generated during the last half of every phase 3. This is used to generate most of the register load strobes.

Because the memory has its own internal timing mechanisms, and the I/O processor has priority over central processor requests, some provision is needed to stop the control state progression while awaiting completion of memory service. This is accomplished by means of the STOP CLK signal (see KP21) which holds the high-speed clock in the high condition. As soon as this line is released, normal cycling resumes. In this way the processor can be held in time state TS03, phase 3, to await RD RST (see Chapter 2) during memory read cycles, and the time state TS02, phase 3, to await ADR ACK during memory writes. Other signals inhibit processor state changes by freezing the enabling levels on the flip-flops. These are described in later paragraphs, along with their uses. Table 3-1 indicates the timing involved in a typical 2-cycle central processor operation.

Table 3-1
Control States for LAC Instruction

Major State	Time State	Phase	Elapsed Time (ns)
Fetch	TS01	0	65
		1	130
		2	195
		3	260
	TS02	0	325
		1	390
		2	455
		3	520
	TS03	0	585
		1	650
		2	715
		3	780
Execute	TS01	0	845
		1	910
		2	975
		3	1040
	TS02	0	1105
		1	1170
		2	1235
		3	1300
	TS01	0	1365
		1	1430
		2	1495
		3	1560

3.5 ADDRESSING

3.5.1 Page Mode

The address portion of a PDP-15 instruction is 12 bits long, sufficient to directly address only 4K of the computer's possible 128K of core memory. The address, therefore, refers to a word in the 4K memory page in which the program is currently running; PC bits 1-5 are appended to MI bits 6-17 to form the address sent out on the MO. Bit 0 is not needed. (Refer to Figure 3-2.)

If the instruction is indirect, the address is formed as above, and a deferred address word is obtained from the location referenced. Bits 0, 1, and 2 of this word are reserved for processor status bits (Link, Bank Mode, User Mode), which are deposited by JMS or CAL and used later to restore the CPU to its previous status. Thus, only 15 bits of the deferred word may be used as a final address. PC bits 1-2 are appended to MI bits 3-17 in this case.

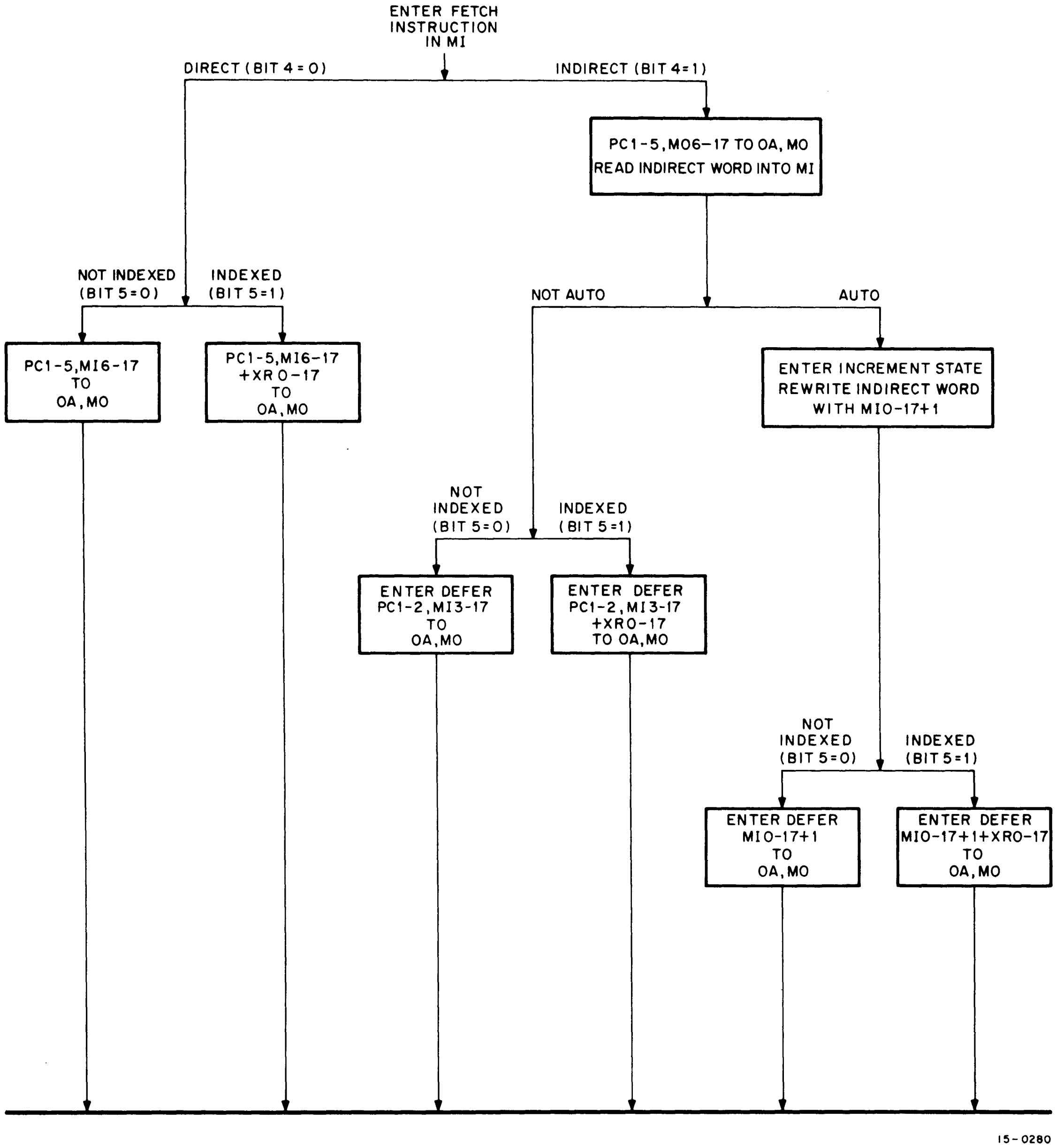

ENTER FETCH INSTRUCTION IN MI
DIRECT (BIT 4 = 0)
INDIRECT (BIT 4=1)
PC1-5,MO6-17 TO OA, MO READ INDIRECT WORD INTO MI
NOT INDEXED (BIT 5=0)
INDEXED (BIT 5=1)
NOT AUTO
AUTO
PC1-5,MI6-17 TO OA, MO
PC1-5,MI6-17 +XR 0-17 TO OA, MO
ENTER INCREMENT STATE REWRITE INDIRECT WORD WITH MI0-17+1
NOT INDEXED (BIT 5=0)
INDEXED (BIT 5=1)
ENTER DEFER PC1-2, MI3-17 TO OA,MO
ENTER DEFER PC1-2, MI3-17 +XR0-17 TO OA,MO
NOT INDEXED (BIT 5=0)
INDEXED (BIT 5=1)
ENTER DEFER MI0-17+1 TO OA, MO
ENTER DEFER MI0-17+1+XR0-17 TO OA, MO
15-0280

Figure 3-2 Page (PDP-15) Mode Address Formation

If the instruction is to be indexed, the XR is added to the final operand address. In this way the programmer can reference an address outside the 32K bank in which his program is running. Since the MI bits use part of the B bus, and the PC bits use part of the A bus, the XR must be split between the two to add properly. For direct addressing the A bus carries PC 1-5 and XR 6-17, while the B bus carries XR 0-5 and MI 6-17. For indirect addressing the A bus hold PC 1-2 and XR 3-17, and B carries XR 0-2 and MI 3-17.

3.5.2 Bank Mode

For compatibility with PDP-9 programs the processor may be put into Bank mode. (Refer to Figure 3-3.) This eliminates all indexing, and thus the possibility of addressing outside the current 32K. Bit 5 is used as part of the instruction address field, expanding direct addressing capabilities to 8K. PC bits 1-4 are now used, along with MI bits 5-17. Deferred addresses are handled in the same way as in Page mode.

During jumps, the final address, formed as described, is placed in the PC. During normal program incrementation and skips, however, only PC bits 6-17 (5-17 in Bank mode) are altered. This causes the program to wrap-around within its own 4K or 8K, when an attempt is made to cross the boundary without a jump.

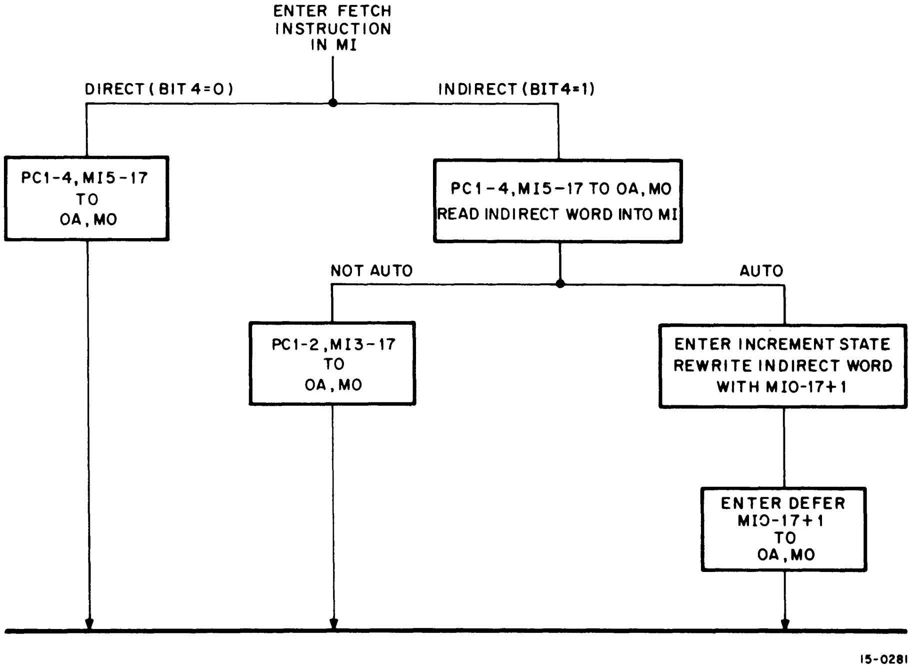


Figure 3-3 Bank (PDP-9) Mode Address Formation

3.6 MEMORY READ AND WRITE

The following paragraphs describe how a memory read or write cycle relates to CPU states. The operation of the memory and of the memory port switch are described in Chapter 2.

Each memory cycle occupies exactly one major state. Refer to Table 3-2. This state can be either a read or a write cycle 1; read/pause/write is used only by the I/O in referencing memory. Time state TS01 is used for address calculation. The information is placed on the A and B buses, goes through the adder, and is strobed into the MO by the CLOCK signal at the end of TS01. The memory is then started to read from or write into this address.

On a read cycle, normal CPU operation continues until phase 3 of the time state TS03 is reached. If the memory has not yet returned the RD RST signal, the processor stops its clock at this point and waits. When RD RST arrives, the data from memory is available on the MDL; it is strobed into the MI and the clock cycles are resumed.

On a write cycle, time state TS02 is used to gate the data to be written through the adder and onto the Sum bus. The clock is stopped (if necessary) at TS02, phase 3 to await ADR ACK from memory. At this point, the data is strobed into the MO replacing the address, and MRLS is sent to the memory, allowing it to complete the write. The normal clock cycles are resumed.

3.7 INSTRUCTION FETCH

Although every instruction begins execution in the Fetch state, this is not the state in which the instruction is fetched from memory. The instruction has already been fetched during the final state (Execute, Fetch, or Defer) of the previous instruction. The Fetch state is entered with the instruction word already in the MI, and with the PC already incremented and pointing to the following instruction.

To accomplish this, the following procedure takes place during the final major state of each instruction. In time state TS01, the address of the instruction to be fetched is sent to the MO. Normally, this is done by JAM PC TO MO signal (see Paragraph 3.1, Memory Output Register). If, however, the SKIP flip-flop (see KP23) is set at this time, or the SAD or OPR SKP signals are high (see KP23), the PC is incremented by sending it through the adder with a CARRY INSERT (see KP48) and strobed into the MO and PC. A memory read cycle is started, as above, and the new instruction is loaded into the MI at the end of the state.

Meanwhile, during the time state TS03, the PC is incremented by gating it to the adder, causing a Carry Insert, and strobing the Sum bus back into the PC. To provide the wrap-around mentioned in Paragraph 3.5.2, only PC bits 6-17 (5-17 in Bank Mode) are strobed. This increment occurs regardless of whether a skip was taken in time state TS01.

3.8 INSTRUCTION OPERATION DETAILS

Paragraphs 3.1 through 3.7 describe the data handling structures and basic operation necessary to implement the PDP-15 instructions. The following paragraphs describe the sequence in which these operations occur for each instruction. Detailed information is provided in Tables 3-3 and 3-4. Supplementary explanations are included in the text.

All descriptions in this paragraph assume direct (non-deferred) addressing and Page (or PDP-15) Mode. Indirect and auto-incrementing address modes are described in Paragraph 3.9. For Bank (or PDP-9) Mode, convert all references to PC 1-5, MI 6-17 to PC 1-4, MI 5-17, and remember that no indexing is possible.

All statements of the form "PC TO OA, MO" mean that the PC is enabled to the bus structure during the entire time state, and the output of the adder is strobed into the OA and MO at CLOCK time (end of phase 3). All incrementing uses the adder with a carry inserted (see Paragraph 3.3).

3.8.1 Read Group (LAC, ADD, TAD, AND, XOR)

These instructions use a Fetch state, in which the operand is brought into the MI, and an Execute state, in which the next instruction is fetched. During time state TS02 of Execute, the data is sent through the adder, operated on appropriately (see Paragraph 3.3), and strobed into the AC. As noted in Paragraph 3.3, ADD requires an extra time state (TS02A) to complete its end-around-carry.

3.8.2 DAC and DZM

These instructions consist of a Fetch state, in which the AC or 0 is written into memory, and an Execute state, in which the next instruction is fetched.

3.8.3 JMP

JMP uses only the Fetch state. During time state TS01, the address is formed and strobed into the PC; at the same time, it is sent to the MO and is used as the address for an instruction fetch.

3.8.4 JMS and CAL

These instructions begin with a Fetch state, in which the Link, Bank mode, User mode, and PC bits 3-17 are written into memory. For the JMS, the address is formed in the usual way from PC 1-5 and MI 6-17, and sent to the OA and MO. For CAL, however, a constant address of 20 must be generated. This is done by enabling the -C-A for bit 13 only, and strobing the OA and MO.

Table 3-2
CPU/Memory Interaction

Time State		Phase	Read Cycle	Write Cycle
MAJOR STATES	TS01	0	PC 1-5 →A BUS MEM IN 6-17 →B BUS	PC 1-5 →A BUS MEM IN 6-17 →B BUS
		1	START READ, MRD	
		2	1 →CP MEM REQ HOLD	START WRITE , MWR 1 →CP MEM REQ HOLD
		3	1 →CP ACT LD MO, OA MO →MDL	1 →CP ACT LD MO, OA MO →MDL
	TS02	0	M REQ	M REQ AC →C BUS
		1		
		2		
		3	ADR ACK (Occurs sometime after M REQ determined by memory) 0 →M REQ 0 →CP MEM REQ HOLD 0 →HOLD MO 0 →MO →MDL	STOP CLK (Wait for ADR ACK) ADR ACK LD MO (Change MO from address to data) 0 →HOLD MO 1 →CP MRLS MRLS
	TS03	0		
		1		
		2		MRLS ACK 0 →CP MRLS, MRLS 0 →CP ACT 0 →MO →MDL 1 →HOLD MO
		3	STOP CLK (Wait for RD RST) RD RST LD MEM IN END OF CP CYCLE 0 →CP ACT 1 →HOLD MO 1 →CP MRLS MRLS, DATA ACK	
	TS01	0	MRLS ACK 0 →CP MRLS, MRLS, DATA ACK	
		1		
		2		
		3		

Table 3-3
Instruction Operation

Instruction	Fetch			Execute		
	TS01	TS02	TS03	TS01	TS02	TS03
LAC	PC 1-5,MI 6-17 (+XR if indexed) to OA, MO. Start Read.			PC JAM to MO. Start Read.	MI to AC	PC + 1 to PC
ADD	PC 1-5, MI 6-17, (+XR if indexed) to OA, MO. Start Read.			PC JAM to MO. Start Read.	MI + AC to AC TS02A: AC end around carry to AC	PC+1 to PC
TAD	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO. Start Read			PC JAM to MO. Start Read.	MI + AC to AC	PC + 1 to PC
AND	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO. Start Read.			PC JAM to MO. Start Read.	MI ∧ AC to AC	PC + 1 to PC
EOR	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO. Start Read.			PC JAM to MO. Start Read.	MI ⊻ AC to AC	PC + 1 to PC
DAC	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO. Start Write.	AC to MO		PC JAM to MO.		PC + 1 to PC
DZM	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO Start Write	0 to MO		PC JAM to MO.		PC + 1 to PC
JMP	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO, PC Start Read.		PC + 1 to PC			
JMS	PC 1-5, MI 6-17 (+XR if indexed) to OA, MO. Start Write.	L,BM,UM,PC 3-17 to MO.		OA + 1 to MO, PC Start Read		PC + 1 to PC
CAL	20 to OA, MO Start Write	L,BM,UM,PC 3-17 to MO.		OA + 1 to MO, PC Start Read		PC + 1 to PC
SAD	PC 1-5, MI 6-17 (+XR if indexed) to MO, OA Start Read.			MI XOR AC to C BUS. C=0: JAM PC to MO C≠0: PC + 1 to MO, PC. Start Read.		PC + 1 to PC

Table 3-3 (Cont)
Instruction Operation

Instruction	Fetch			Execute		
	TS01	TS02	TS03	TS01	TS02	TS03
OPR	OPR SKIP: PC+1 to MO,PC $\overline{\text{OPR SKIP}}$: PC JAM to MO Start Read (CLR AC)	(CLR LINK) Operate on AC, L	PC + 1 to PC			
LAW	PC JAM to MO Start Read	MI to AC	PC + 1 to PC			
IOT	IOT Request 0 to AC if MI14=1 No memory cycle.		Stop Run Start Run on IOT	SKIP: PC+1 to MO, PC $\overline{\text{SKIP}}$: PC JAM to MO Start Read.		
XG	No memory cycle.	Transfer, ADD to AC, LR, XR	Done if AXS, TEST. XR=LR: 1→SKIP XR<LR: 0→SKIP	SKIP: PC+1 to MO, PC $\overline{\text{SKIP}}$: PC JAM to MO Start Read.		
PI	0 to IR (CAL) I/O Address (=0) to OA, MO Start Write	L,BM,UM,PC 3-17 to MO		OA + 1 to MO,PC Start Read.		
API	1 to IR00 (XCT) I/O address to OA, MO Start Read.			Execute operand as instruction		

An Execute state follows in which the OA + 1 is loaded into the MO and PC, and an instruction fetch is carried out from this location.

3.8.5 ISZ

The ISZ instruction uses three major states; Fetch, Increment, and Execute. (Refer to Table 3-4.) The Fetch cycle reads an operand in the usual manner. The Increment cycle adds 1 and rewrites the word in memory. (Read/Pause/Write is not used.) While the operand is being incremented (in time state TS02), the high-order carry output is checked, and if a carry occurs the SKIP flop (KP23) is set. This implements the skip-on-zero. The Execute state is an instruction fetch, using the PC or the PC + 1 according to the state of SKIP.

Table 3-4
ISZ Instruction Operation

Major State	Time State	Operation
Fetch	TS01	PC 1-5, MI 6-17 (+XR if indexed) to MO, OA Start Read.
	TS02	
	TS03	
Increment	TS01	OA to MO Start Write
	TS02	MI + 1 to MO If carry out, set SKIP
	TS03	
Execute	TS01	$\overline{\text{SKIP}}$: PC JAM to MO SKIP: PC + 1 to MO, PC Start Read.
	TS02	
	TS03	

3.8.6 SAD

The Fetch cycle of SAD brings in an operand. This is followed by an Execute cycle in which the comparison, skipping, and instruction fetch are carried out. During time state TS01 of execute, the XOR of the AC and the MI (operand) is gated onto the C bus. If the C=0 circuitry (see KP33) shows a perfect match, PC JAM to MO is used and the instruction fetch occurs with no skip. If C $\neq$ 0, the PC is incremented before the fetch. This incrementation is set up using the A bus, so it does not interfere with the comparison on the C bus.

3.8.7 XCT

This instruction goes through a Fetch state, getting an operand in the usual manner, and then enters the Fetch state a second time. The operand, now in the MI, is treated exactly as if it had been encountered in normal program flow as an instruction that was fetched into the MI by a previous instruction. All of the instruction states will occur, and the PC will be incremented or altered as usual. Refer to Table 3-5.

Table 3-5
XCT Instruction Operation

Major State	Time State	Operation
Fetch	TS01	PC 1-5, MI 6-17 (+XR if indexed) to MO, OA. Start Read.
	TS02	
	TS03	
Fetch	TS01	Execute operand as instruction
	TS02	
	TS03	

3.8.8 OPR

This instruction group uses a single Fetch state for its execution. In time state TS01, the OPR SKIP logic (see KP23) determines whether a skip will take place, and the fetch for the next instruction is initiated. The actual operation on the Link and AC takes place in TS02, by setting up the appropriate buses and strobing the output into the Link and AC. The only exceptions to this are the clear operations which take place during TS01; CLOCK, for the AC; and TS02, phase 1, for the Link.

3.8.9 LAW

LAW consists of a single Fetch state, used to fetch the following instruction. During TS02, the LAW instruction (still in the MI) is loaded into the AC.

3.8.10 IOT

The IOT begins with a Fetch state, in which no memory cycle occurs. IOT REQUEST is raised, activating the I/O processor (see Chapter 4). If bit 14 of the MI is set, the AC is cleared at clock time of TS01. The Fetch state continues until, at the end of TS03, phase 2, the RUN flop (KP21) is dropped, stopping all CPU phase transitions. By this mechanism, the CPU waits for the IOT DONE signal (see KP55) which sets RUN. The I/O processor, in the meantime, may have strobed the I/O Bus data lines into the AC, and may have set the SKIP flop. An Execute state follows Fetch, and the next instruction is brought in, controlled by SKIP.

3.8.11 Index Group (XG)

These instructions consist of a Fetch state, with no memory operation, followed by a standard Execute state instruction fetch.

In time state TS02 of Fetch, the transfer (PAX, PAL, PXA, PXL, PLA, PLX); clear (CLX, CLR); or add (AXR, AAC, AXS) takes place. The add gates the AC or XR onto the A bus, and MI 9-17 onto the B bus. If MI bit 9 is a 1, the number is negative and bits 0-8 must be filled with the 1s for proper addition. This is done by activating both MI-B 0-8 and -MI-B 0-8 simultaneously during the addition (see KP19).

Time state TS03 of Fetch is used only by the AXS, for comparing the XR to the LR. The XR is gated onto the B bus; the two's complement negative of the LR is added to it by gating the LR to the C bus, -C to the A, and raising carry in. The high order carry output, along with the XR and LR signs, indicates whether SKIP should be set (see KP29). Like signs with a carry or opposite signs with no carry indicate that XR $\geq$ LR and a SKIP is called for. Skip, of course, takes effect in the following Execute state.

3.8.12 Interrupts (API and PI)

While API and PI are not properly instructions, they behave in very much the same manner once they are recognized. The recognition sequence is described in Chapter 6 for API and Chapter 4 for PI.

Whenever the INT ACK + ST signal is asserted and the CPU enters an instruction Fetch cycle, the interrupt begins. An instruction fetch is allowed to take place, but since the instruction will not be used at this time, the PC increment is disabled (KP24). Upon entering the Fetch state, the interrupt takes full precedence over the instruction in the MI.

If the interrupt is a PI, the 00 code (CAL) is forced into the IR. From this point, execution is identical to a CAL, except the I/O Address Lines are used to provide the operand address, instead of the constant 20. These lines will always give a 0 on PI requests. Thus the L, BM UM and PC 3-17 are stored in location 0 and the next instruction is fetched from location 1.

The API forces an XCT code (40) into the IR, and proceeds as an XCT would, except that the address of the operand is read from the I/O Address Lines instead of the MI. The device requesting the API must, when acknowledged, place the appropriate interrupt address on these lines. Thus, some instruction (in an address peculiar to the device) is executed. This will usually be a JMS to the interrupt handler.

3.9 DEFER AND AUTO-INCREMENT

If bit 4 of any memory reference instruction is set, that instruction is to use deferred (or indirect) addressing. Instead of the usual Fetch state, the instruction begins with Fetch and Defer states, as shown in Table 3-6. The Fetch state is always a memory read, which brings in the pointer or indirect address word. The Defer state is almost identical to the Fetch state of the same instruction in non-deferred mode; it brings in or writes out the operand, just as Fetch would have. The only difference is that MI 3-17 are used instead of MI 5-17. In the Defer state, indexing occurs after the indirect-addressed word is obtained. The remaining states of the instruction proceed as though in non-deferred mode.

If a deferred instruction has an address of 10-17 (octal), regardless of the PC contents, the instruction is to use Auto-Increment mode addressing. Refer to Table 3-7. The contents of the addressed word, absolute 000010-000017, are first incremented, then used as a deferred address. This is accomplished by replacing the normal Fetch state with the Fetch, Increment, Defer sequence shown in the table. This sequence is caused by the circuitry on KP33 which sets the AUTO flop during TS01. The Fetch state is used to read the indirect word from memory. Use of locations 10-17 on the lowest core page is assured by gating only MO 6-17 to the MDL. In the Increment state, the pointer is incremented

Table 3-6
Deferred Addressing Operation

Major State	Time State	Operation
Fetch	TS01	PC 1-5, MI 6-17 to OA, MO Start Read
	TS02	
	TS03	
Defer	TS01	PC 1-2, MI 3-17 (+XR if indexed) to OA, MO Start Read or Write
	TS02	Identical to operation specified for non-Defer, Fetch, TS02.
	TS03	Identical to operation specified for non-Defer, Fetch, TS03.

and re-written. In the Defer state the original pointer plus one is used as the address and the operand is either read or written into, as it would be in the Fetch state of a normal mode instruction.

CAL can be deferred, but not auto-incremented. In deferred mode, the contents of location 20 are used as the final address. Operation is identical to that shown in the table, except that an address of 20 is forced in TS01 of Fetch.

In a JMP deferred or auto, the Defer State is the final state of the instruction, and thus fetches not an operand, but the next instruction. In TS01 of Defer, the final address is strobed into the PC as well as the OA and MO.

ISZ auto uses the Increment state twice; the major state sequence is Fetch, Increment, Defer, Increment, Execute. The first increment is for rewriting the address pointer, while the second is for the normal increment of the operand. The skip-on-zero circuit can only set SKIP during the second Increment state, while AUTO is zero.

All 18 bits of the auto-increment register are used as an address. This provides another method of addressing more than 32K of core memory.

Table 3-7
Auto-Increment Operation

Major State	Time State	Operation
Fetch	TS01	PC 1-5, MI 6-17 to OA,MO 1 → AUTO Start Read (MO 6-17 to MDL)
	TS02	
	TS03	
Increment	TS01	OA to MO Start Write (MO 6-17 to MDL)
	TS02	MI + 1 to MO
	TS03	
Defer	TS01	MI 0-17 + 1 (+XR if indexed) to OA, MO Start Read or Write
	TS02	Identical to non-Defer, Fetch, TS02
	TS03	Identical to non-Defer Fetch, TS03 0 → AUTO

3.10 CONSOLE OPERATION

3.10.1 Console Cable Multiplexer

Only two 18-conductor flexprint cables are used between the CPU and the console. These are multiplexed six ways by a free-running 36 kHz clock and a modulo -6 counter (see KP45, KP46). Three lines of the cable carry the state identification (console zero=high order), and 24 of the others comprise the I bus which carries information to or from the console, depending on the console state. The I bus assignments and timing are shown in KP72.

During console states 0, 1, and 2, various internal CPU signals are placed on the I bus, and three different sets of console lights are sequentially enabled to display this information. Each light, when on, is only enabled for one sixth of each 333 µs console cycle, but brightness is maintained by using

30 volts instead of the normal 18 volts to drive the lamps. Console state 0 and 2 display permanently assigned CPU signals; state 1 displays one of 24 registers, as selected by the rotary switch on the console.

Console states 3, 4, and 5 are used to read the status of the various console switches into the CPU. In general, these are read into active registers so that they can be referenced by the CPU at any time. The rotary switch status register is shown in KP44. The various key functions are stored in the flops shown in KP45 and KP46. The data switch register appears with the main CPU registers in KP01 through KP18. The address switches are not stored in a register; whenever the CPU references these switches it must wait for console state 4 (CF pulse).

3.10.2 Key Functions

The control flow of the various console key functions (start, continue, execute, examine, and deposit) is shown in KP73. This paragraph provides additional explanation. Most of the associated logic appears in KP34.

The Stop switch causes the RUN flop (see KP21) to drop as soon as the CPU enters phase 3, time state TS03, of the final major state of an instruction. (Set Fetch is high.) The CPU is effectively frozen at the end of an instruction, with the next instruction in the MI, but with the PC not yet incremented.

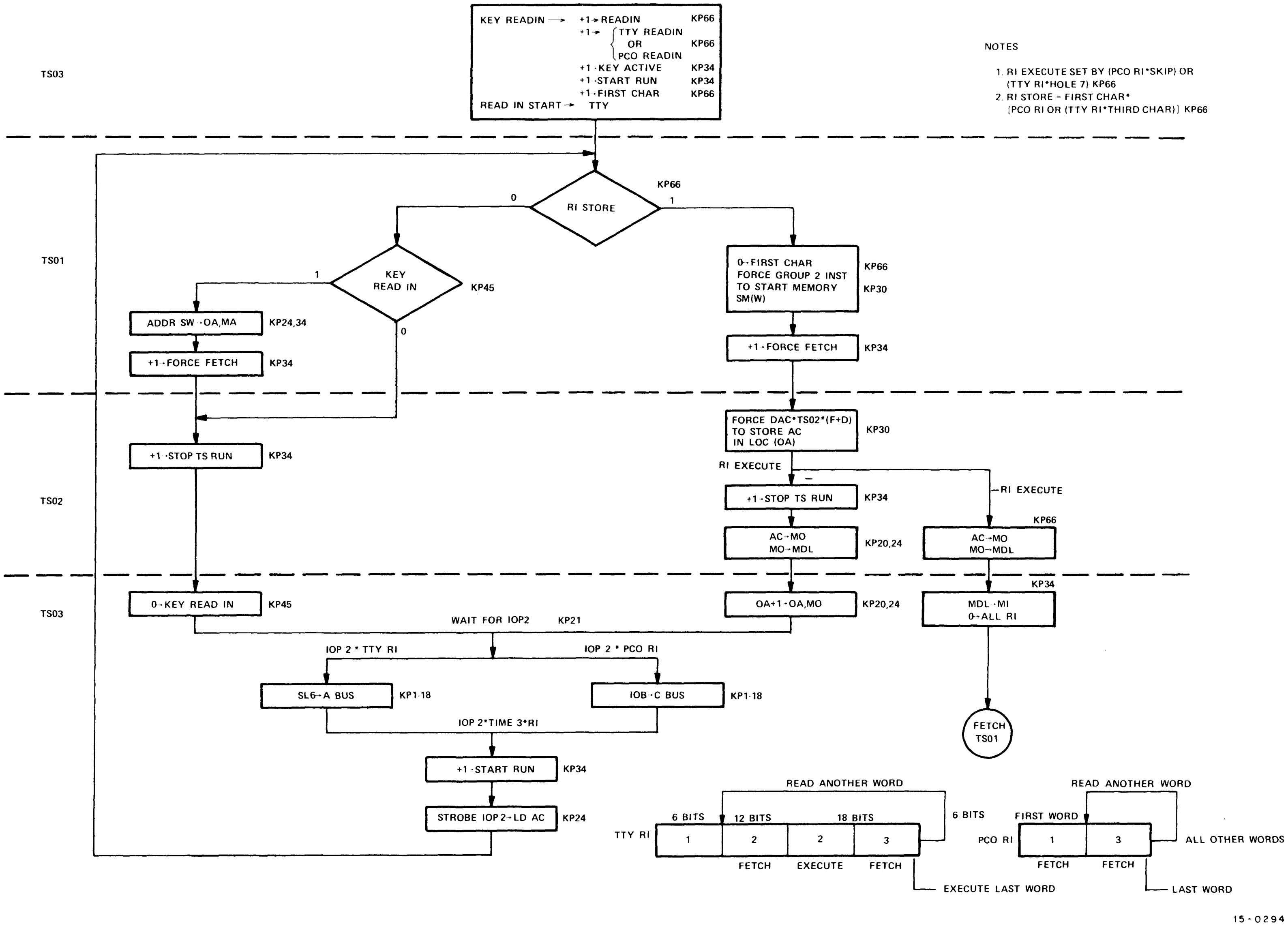
Once the CPU is stopped, deposits and examines may be performed. These start RUN, go through a forced Fetch state in which a memory word is read or written, then drop RUN again, under control of the STOP TS RUN flop (see KP34). Executes operate similarly, using a forced fetch to load the data switches into the MI, then executing this as an instruction. RUN is stopped after this operation by the XSW IN PROG flop (see KP34).

START and CONTINUE (when stopped at TS03, Set Fetch) both use a forced Fetch state to read in the next instruction. START uses the Address Switches as an address, while CONTINUE uses the PC. The PC is incremented, and the CPU resumes normal operation. If CONTINUE is used and the CPU is not in TS03 of a Set Fetch state, RUN is simply raised and operation resumed.

Single Instruction, Single Step, and Single Time operate by dropping RUN at the appropriate intervals (see KP21).

3.11 READ IN

To load binary tapes or bootstraps (refer to Figure 3-4), the PDP-15 uses a hardware read-in function, which loads core from either the Teletype or the high-speed paper-tape reader, depending on an



NOTES

1. RI EXECUTE SET BY (PCO RI*SKIP) OR (TTY RI*HOLE 7) KP66
2. RI STORE = FIRST CHAR* [PCO RI OR (TTY RI*THIRD CHAR)] KP66

Figure 3-4 Read-In Flow Diagram

internal jumper wire shown in KP66. Each frame of the tape carries data in bits 1-6; three consecutive frames form an 18-bit word for storage. Bit 7, if punched, causes the last word read to be executed as an instruction, terminating the read. Bit 8 is always punched. The read-in logic is shown in KP66 and KP49; the flow diagram is in KP75.

In Teletype read-in operation, the address switches are first read into the OA and MO. These give the core location where loading is to begin. The time states are allowed to run, but are stopped by STOP TS RUN (see KP34) when TS03 is reached. Meanwhile, the Teletype reader has been started by the READ IN START signal (KP66), and eventually this returns with a character. This causes an IOP2 pulse which stores the character in the AC via ALS6 (see Paragraph 3.2.2, A bus), restarts the time states, and increments the character counter (see KP66). When three such characters have been brought in, they are written into memory, the OA is incremented, and the process continues. When a bit 7 punch is detected, the AC is strobed into the MI instead of being written into memory, all the read-in enables are dropped, and the MI is executed like an instruction. Usually this will be a jump to the start of the program.

Operation using the high-speed reader is quite similar to Teletype read-in. The principal difference is that the reader interface, instead of the CPU, packs the three 6-bit characters into a word, so each cycle after the first is a store and read cycle. The character counter and the ALS6 logic is not needed.

CHAPTER 4
I/O PROCESSOR

4.1 KD15 INPUT/OUTPUT PROCESSOR

The I/O processor coordinates data transfer between the central processor and peripheral devices and also between core memory and peripheral devices. Data transfer between the CPU and peripheral devices is called program control transfer and is implemented by the input/output transfer (IOT) instructions. Data transfer between core memory and peripheral devices is accomplished by a request/grant priority scheme and is referred to as the data channel. Table 4-1 summarizes the PDP-15 input/output facilities.

Program control transfers occur as a result of the IOT instruction execution. These instructions, contained in the body of the main program or in appropriate subroutines, are microcoded to effect response of a specific device interfaced to the I/O bus system. The microcoding includes the issue of a unique device selection code and appropriate processor generated pulses to initiate device operations such as transmitting data from the device to the central processor, or from the processor to the device. All program control transfers are executed through the accumulator in 18-bit words. This portion of the I/O processor also contains facilities for skipping on device flags and interrupts which cause a break in the normal flow of central processor operations.

The data channel facility provides for high-speed transfer of data in blocks between peripherals and system core memory. Since the I/O processor and central processor of the PDP-15 are asynchronous, a data channel transfer request raises an I/O memory request, which is granted and transmitted to memory at the conclusion of the current central processor memory reference in progress. These requests will continue to take priority over the central processor until the transfer is completed. The types of transfers available to the data channel facility are single cycle input and output transfers, multicycle input and output transfers, add to memory transfers, and increment memory transfers. The I/O processor consists of five sections:

Timing Generator
Request Synchronizer
IOT Control Logic
I/O Bus Control Logic
Data Channel

Table 4-1
Summary of PDP-15 Input/Output Facilities

Facility	Remarks
	Data Transfers To/From Memory
Multi-Cycle Data Channel Input	Used to transfer data directly to core memory in up to 18-bit words at high speed (250 kHz).
Multi-Cycle Data Channel Output	Used to transfer data directly from memory in 18-bit words. Maximum speed is 188 kHz.
Add to Memory	Used to add the contents of a device register to the contents of a specified core location in 18-bit words. Maximum speed is 188 kHz.
Increment Memory	This facility allows an external device to increment the content of a core location by 1. Maximum speed is 333 kHz.
Single-Cycle Data Channel Output	With this facility a device can transfer a burst of data from core memory at 1 mHz in 18-bit words.
Single-Cycle Data Channel Input	Used to transfer a burst of data from a device to core memory at 1 mHz per 18-bit word.
	Data Transfers To/From CPU
Addressable I/O Bus	With this facility, devices can transfer data in 18-bit words to or from the central processor. A typical rate is one transfer every 200 µs.
	Command and Status Transfers
Addressable I/O Bus	Command and status information can be transferred to or from the CPU in the same manner as ordinary data.
Read Status	This is a special facility designed to allow the user to monitor all vital flags in the system. Each device is assigned a bit for its flag(s), which is read onto the addressable I/O bus and into the CPU when the Read Status command is given. No two devices should use the same bit.
Skip	The addressable I/O bus allows the computer to test the status of a flag (typically) by issuing a pulse which will echo if the addressed flag is up. Every flag that posts a program interrupt must be identifiable by the skip facility.
	Interrupts
Program Interrupt	All devices share a common program interrupt line. When a device posts an interrupt the computer is forced to location 0, bank 0, and then on to a service routine designed to identify the requesting device using the skip facility.

Table 4-1 (Cont)
Summary of PDP-15 Input/Output Facilities

Facility	Remarks
	Interrupts (Cont)
Automatic Priority Interrupt	This facility reduces the time to service a requesting device and establishes a priority among devices so that important interrupts can be handled quickly and without interference.

4.2 TIMING GENERATOR

The timing for the I/O processor is controlled by an M401 clock located on KP51 which runs freely, with the exception of waiting for memory, on multicycle and single cycle output transfers. This clock steps a 2-bit counter which is decoded on KP55 into 4 times called TIME 1, TIME 2, TIME 3 and TIME 4. This is shown in Figure 4-1. TIME 1 is called I/O SYNC, and is used to synchronize devices on the I/O bus to the I/O processor. The frequency of the I/O clock is 4 MHz. The overall frequency of the I/O processor is 1 MHz and therefore the I/O clock on KP51, N21-E2 should be set so that pulses are 250 ns apart.

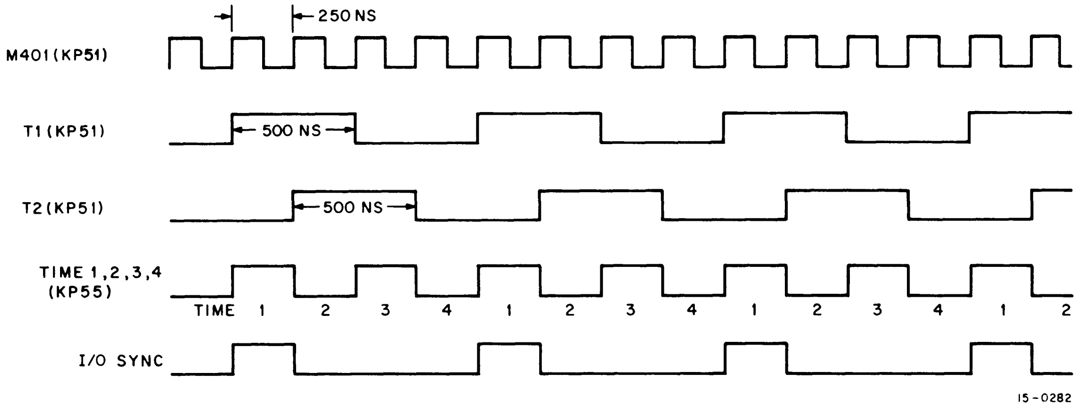


Figure 4-1 Basic I/O Timing

4.3 REQUEST SYNCHRONIZATION

A priority structure is set up in the PDP-15 as follows:

a. Data channel request

b. Clock requests

c. Automatic priority interrupt requests

d. Program interrupt request

e. Main program

In order to establish these priorities a two stage synchronizer is provided in the I/O processor as shown in KP51. Each priority contains two stages of synchronization. The first stage consists of the DCH, CLOCK, API, PI, and IOT flops on KP51. The various requests for activity on the data input to these flops are clocked at TIME 4. Any one or all of these flops may become set at this time. 250 ns later, at TIME 1, a clock pulse is provided to the sync flops immediately above each of the first stage synchronizer flops, and sets one of these flops. During the previous 250 ns, if a higher priority request had been set into one of the first stage flops, a clear would have been transmitted to all lower request flops and, therefore, only one of the second stage synchronizer flops would become set. The setting of the sync flop will allow highest priority request activity to proceed. Once one of the lower priority sync flops is set, such as in program control transfers, it will remain set until the transfer is completed.

4.4 IOT

The IOT instruction is a command from the central processor to transfer data from the central processor (accumulator) to the device or to read data back from the device into the accumulator of the central processor. The instruction format in Figure 4-2 consists of the instruction code; a 6-bit device select and a 2-bit subdevice select code which are put onto the I/O bus to designate the device with which the processor wishes to communicate; a clear the AC bit which if set, will cause the clearing of the accumulator; and three I/O pulses; any one or all of which may be given in a single IOT instruction.

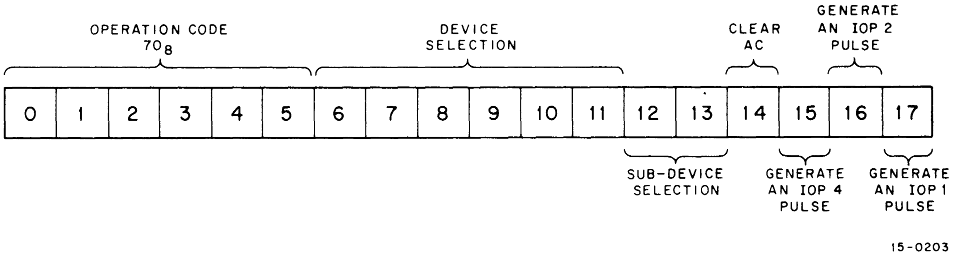


Figure 4-2 IOT Instruction Format

The general purpose of these pulses is as follows: IOP 1 transmits data to the device, tests the device flag, and causes the program to skip the next sequential instruction; IOP 2 transmits data to or from the device via the accumulator; IOP 4 transmits data to the device from the accumulator.

The IOT instruction requires the operation of both central processor and the I/O processor. The central processor, upon detecting the IOT instruction, sets the IOT request flip-flop on KP35. If MI bit 14 is set, the AC will be cleared. When the processor reaches FETCH, TS03, and CLOCK, it halts and waits for the I/O processor to finish its execution and respond with IOT DONE.

The I/O processor has to synchronize the IOT on KP51 with the other priority requests. Upon synchronization, the information in the accumulator is enabled to I/O bus lines along with the device select and subdevice select bits from MI bits 6 to 13. IOP 1 is enabled on the bus if bit 17 in the MI is set. IOP 1 in the I/O processor lasts for 1 μs. IOP 1 on the bus is 750 ns long. At the end of IOP 1, if neither bit 15 nor 16 in the MI is set, IOT DONE is generated. Otherwise, IOP 2 in the I/O processor is set. If bit 16 in the MI is set, IOP 2 is enabled on the I/O bus for 750 ns. If READ REQUEST is true at this time the AC is disabled from the I/O bus. 750 ns after IOP 2 is placed on the bus, the AC is strobed and information that was on the I/O bus is placed in the accumulator. At the end of IOP 2 if bit 15 of the MI is not set, an IOT DONE is generated. If it is set, IOP 4 is set. IOP 4 is enabled onto the bus for 500 ns. After this time, an IOT DONE is generated; the IOT and IOT SYNC flops on KP51 are cleared out and the central processor is restarted. Figure 4-3 shows the IOT instruction timing, giving the synchronization time between central processor and I/O processor. An IOT flow diagram is in Figure 4-4.

4.5 I/O BUS

The I/O bus cables are shown in Figure 4-5. The I/O bus signals are described in Table 4-2.

4.6 DATA CHANNEL FACILITY (DCH)

Refer to Figure 4-6. The hardware to implement the memory to peripheral device data transfer includes: a bus buffer for temporary storage (I/O Buffer 0-17, KD04, KD05); a data storage register, an input mixer and adder used to transfer information from the devices to memory and from memory to the devices (DSR KD01, KD02, KD03); priority logic for the synchronization of requests (KP51) and generation of GRANT (KD06); and DCH control logic (KD04, KD05, and KD06).

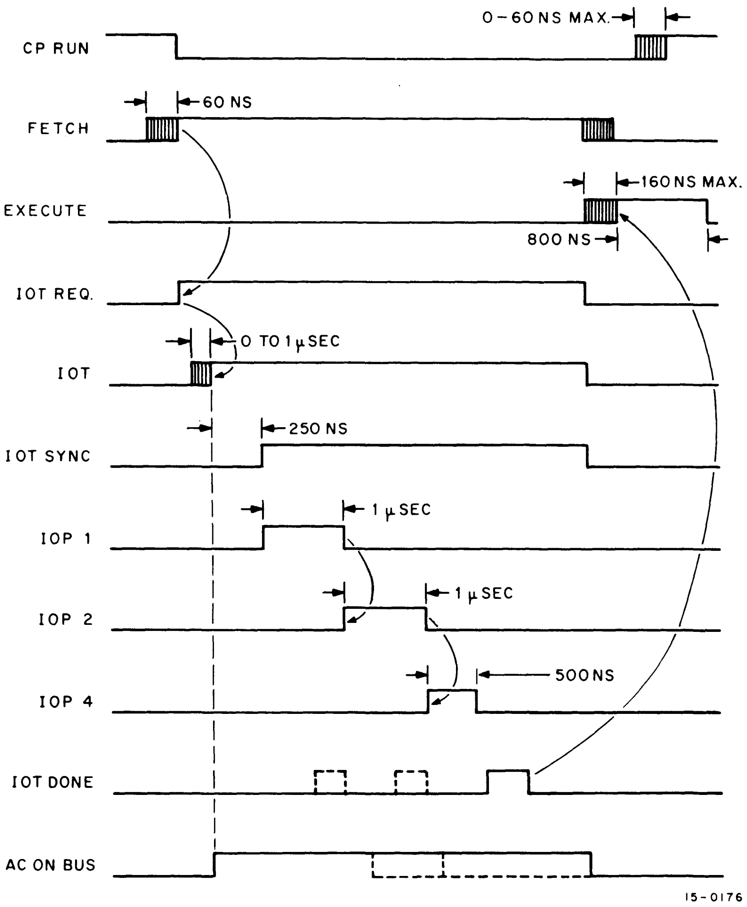


Figure 4-3 IOT Instruction Timing

Data channel devices on the I/O bus are initialized by IOT commands, and the data transfer process is initiated also by IOT. After initialization, the device, when ready for a transfer, raises its flag, which is then synchronized to the I/O processor. The device then transfers the data through the I/O processor to or from memory.

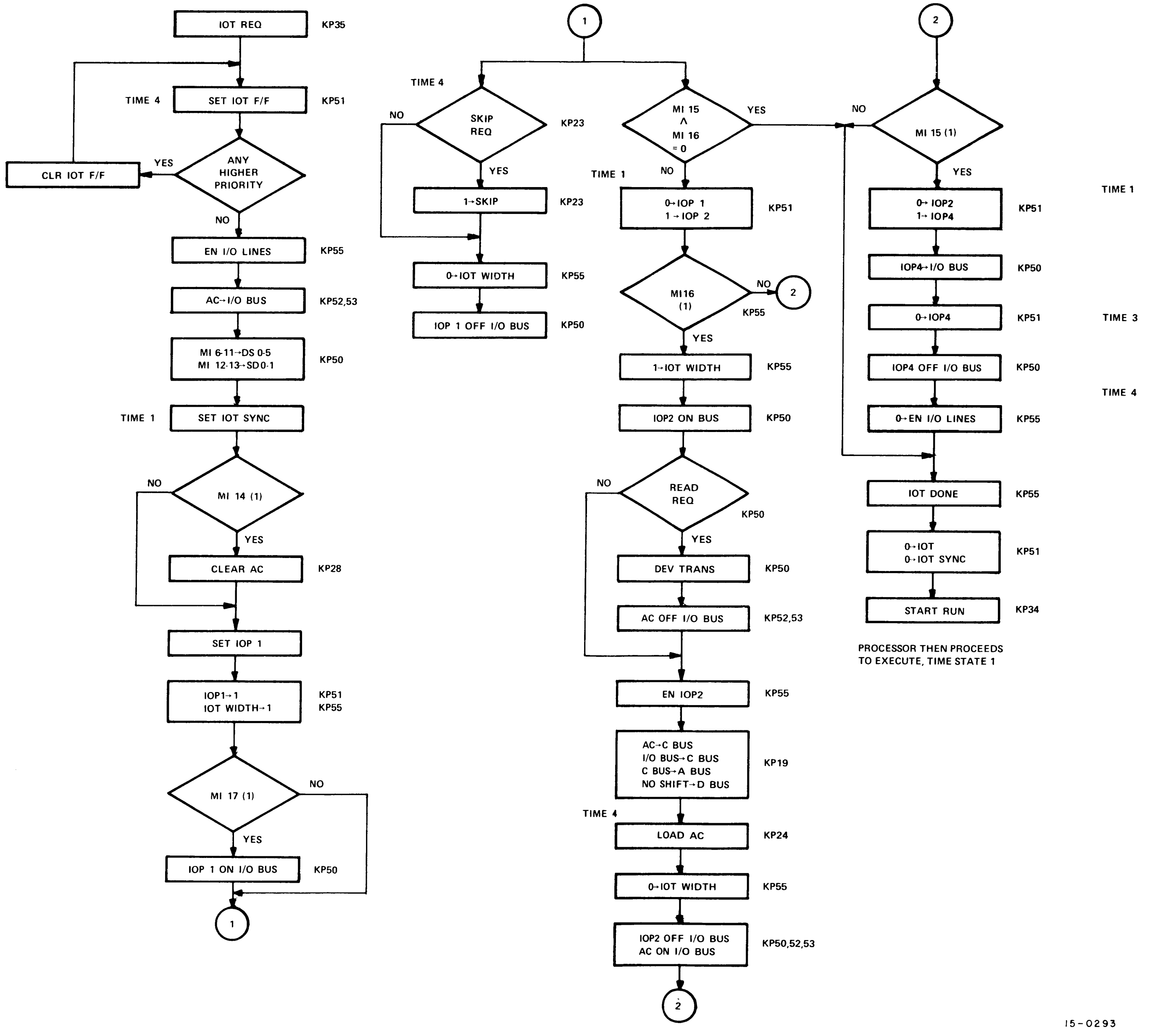


Figure 4-4 IOT Flow Diagram

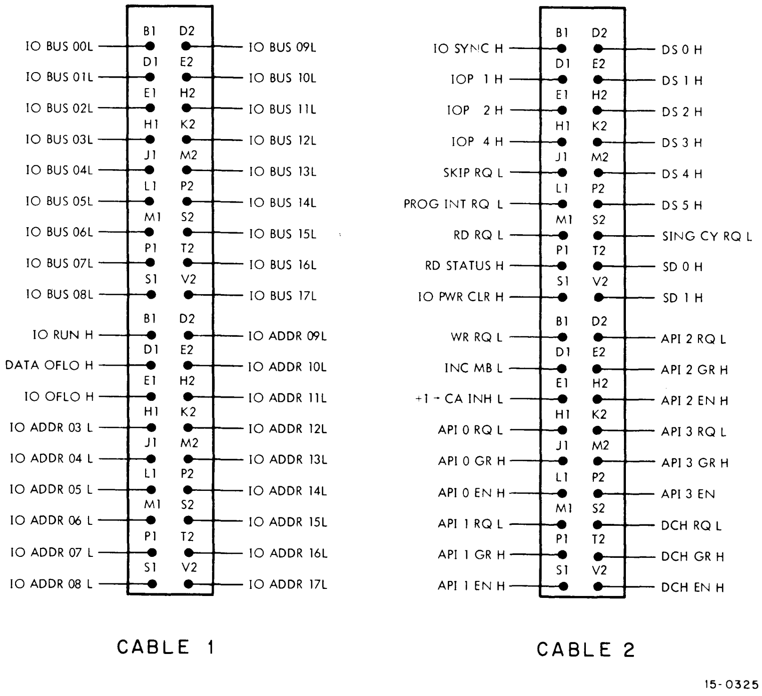


Figure 4-5 I/O Bus Cables

Table 4-2
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
API 0 EN H	2BL1	This enable signal, one of four in the API system, is a dc level originating in the I/O Processor and daisy chained from device to device on the same level. The M104 logic in each controller can interrupt this level, cutting the level off all devices that follow it on the bus. A device receives it as API 0 EN IN H and transmits it as API 0 EN OUT H.	Each device can post a request to its API level only if the incoming API EN level is true. By posting a request the device immediately inhibits all controllers below it on the bus. In this way priorities on each level are established when devices request simultaneously.
API 1 EN H	2BS1	Same as API 0 EN H	Same as API 0 EN H
API 2 EN H	2BH2	Same as API 0 EN H	Same as API 0 EN H
API 3 EN H	2BP2	Same as API 0 EN H -	Same as API 0 EN H
API 0 GR H	2BJ1	One of four possible signals issued by the I/O processor indicating that it grants the API request at the corresponding level.	The device uses this signal to gate the address of its API entry location onto the I/O ADDR lines.
API 1 GR H	2BP1	Same as API 0 GR H	Same as API 0 GR H
API 2 GR H	2BE2	Same as API 0 GR H	Same as API 0 GR H
API 3 GR H	2BM2	Same as API 0 GR H	Same as API 0 GR H
API 0 RQ L	2BH1	One of four API request signals on channels 0-3. This signal is set by the device at I/O Sync time.	The device uses this signal to inform the I/O processor of its request for API priority level 0, the highest of the four.
API 1 RQ L	2BM1	Same as API 0 RQ L	Request API priority level 1.
API 2 RQ L	2BD2	Same as API 0 RQ L	Request API priority level 2.
API 3 RQ L	2BK2	Same as API 0 RQ L	Request API priority level 3.
DATA OFLO H	1BD1	This signal is gated onto the bus by the I/O processor during the third cycle of an add-to-memory operation when the sum (1's complement) of two like-signed numbers has an opposite sign.	This signal is used by the device to notify it when an incorrect sum occurs, because of overflow during an add-to-memory operation.

Table 4-2 (Cont)
I/O Bus Signal Functions

<table>
<tr><th>Signal Mnemonic</th><th>Connector Pin Number</th><th>Signal Definition</th><th>Signal Function</th></tr>
<tr><td>DCH EN H</td><td>2BV2</td><td>This enable signal is a dc level originating at the I/O processor and daisy chained from device to device. The M104 logic in each device can interrupt this level, cutting the level off all devices that follow on the bus. A device receives it as DCH EN IN H and transmits it as DCH EN OUT H.</td><td>Each device can post a DCH request only if the incoming DCH EN level is true. By posting a request, the device immediately inhibits all controllers below it on the bus. In this way priorities are established when devices request simultaneously.</td></tr>
<tr><td>DCH GR H</td><td>2BT2</td><td>Issued by the I/O processor when it acknowledges a device's DCH RQ L.</td><td>The device uses DCH GR to gate the address of its word count onto the I/O ADDR for 3-cycle transfers and gates memory address during 1-cycle transfers.</td></tr>
<tr><td>DCH RQ L</td><td>2BS2</td><td>A signal from a device to the I/O processor indicating either a request for a multicycle data channel transfer or, when posted with a single cycle request, showing that an input transfer must be effected. The table below shows how the two functions relate.
<table>
<tr><th>DCH RQ L</th><th>SING CY RQ L</th><th>FUNCTION</th></tr>
<tr><td>0</td><td>0</td><td></td></tr>
<tr><td>0</td><td>1</td><td>Single Cycle Transfer Out</td></tr>
<tr><td>1</td><td>0</td><td>Multi Cycle Transfer (In or Out)</td></tr>
<tr><td>1</td><td>1</td><td>Single Cycle Transfer In</td></tr>
</table></td><td>This signal is interpreted by the I/O processor in two ways:
If it is present without a single-cycle request, it implies that some device wants to carry out a multicycle transfer, an increment memory, or add to memory.
If a single-cycle request is also posted, then the two signals are ANDed to inform the I/O processor that a single-cycle transfer into memory is to be effected. Otherwise, the I/O processor assumes an outgoing single-cycle transfer is required.</td></tr>
<tr><td>DS0 H</td><td>2AD2</td><td>The first of six device select lines decoded from bit 6 of the IOT instruction.</td><td>This signal together with DS1-DS5 is decoded by the device select logic in the controller, which responds to its unique code only.</td></tr>
<tr><td>DS1 H</td><td>2AE2</td><td>The second of the six device select lines.</td><td>See DS0 H</td></tr>
<tr><td>DS2 H</td><td>2AH2</td><td>The third of the six device select lines.</td><td>See DS0 H</td></tr>
</table>

Table 4-2 (Cont)
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
DS3 H	2AK2	The fourth of the six device select lines.	See DS0 H
DS4 H	2AM2	The fifth of the six device select lines.	See DS0 H
DS5 H	2AP2	The sixth of the six device select lines.	See DS0 H
INC MB L	2BD1	Forces the I/O processor to increment the contents of the memory location specified by the 15-bit address lines on the I/O bus.	This feature allows a device to increment memory locations in one cycle without disturbing the CPU.
I/O ADDR 03 L	1BH1	One of fifteen lines which constitute an input bus for devices which must deliver address data to the processor.	This address bus has two uses: a) To deliver the device's API entry location during its API break. b) To deliver the device's word count address during a multicycle DCH transfer, an increment memory operation, or add to memory. To deliver an absolute address during single cycle transfers.
I/O ADDR 04 L	1BJ1	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 05 L	1BJ1	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 06 L	1BM1	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 07 L	1BP1	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 08 L	1BS1	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 09 L	1BD2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 10 L	1BE2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 11 L	1BH2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 12 L	1BK2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 13 L	1BM2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L

Table 4-2 (Cont)
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
I/O ADDR 14 L	1BP2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 15 L	1BS2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 16 L	1BT2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O ADDR 17 L	1BV2	Similar to I/O ADDR 03 L	Similar to I/O ADDR 03 L
I/O BUS 00 L	1AB1	The first of 18 data lines which constitute the bidirectional facility for transferring data in bytes of up to 18 bits between the device and either the CPU or memory. This is the MSB.	These data lines (I/O BUS 00 L through I/O BUS 17 L convey data between a) the AC of the CPU and a selected device information buffer register or b) the bus buffer of the I/O processor and a selected device buffer register during data channel operations.
I/O BUS 01 L	1AD1	Data line two	See I/O BUS 00 L
I/O BUS 02 L	1AE1	Data line three	See I/O BUS 00 L
I/O BUS 03 L	1AH1	Data line four	See I/O BUS 00 L
I/O BUS 04 L	1AJ1	Data line five	See I/O BUS 00 L
I/O BUS 05 L	1AL1	Data line six	See I/O BUS 00 L
I/O BUS 06 L	1AM1	Data line seven	See I/O BUS 00 L
I/O BUS 07 L	1AP1	Data line eight	See I/O BUS 00 L
I/O BUS 08 L	1AS1	Data line nine	See I/O BUS 00 L
I/O BUS 09 L	1AD2	Data line ten	See I/O BUS 00 L
I/O BUS 10 L	1AE2	Data line eleven	See I/O BUS 00 L
I/O BUS 11 L	1AH2	Data line twelve	See I/O BUS 00 L

Table 4-2 (Cont)
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
I/O BUS 12 L	1AK2	Data line thirteen	See I/O BUS 00 L
I/O BUS 13 L	1AM2	Data line fourteen	See I/O BUS 00·L
I/O BUS 14 L	1AP2	Data line fifteen	See I/O BUS 00 L
I/O BUS 15 L	1AS2	Data line sixteen	See I/O BUS 00 L
I/O BUS 16 L	1AT2	Data line seventeen	See I/O BUS 00 L
I/O BUS 17 L	1AV2	Data line eighteen This is the LSB	See I/O BUS 00 L
I/O OFLO H	1BE1	This signal is issued during the first cycle of a multicycle data channel transfer or an increment memory cycle if the content (2's complement) of the word count assigned to the currently active data channel device becomes zero when incremented.	This signal indicates to the device that the specified number or words have been transferred at the completion of the transfer in progress. It is normally used to turn off the respective device and to initiate a program interrupt or API request.
IOP 1 H	2AD1	Microprogrammable control signal part of an IOT instruction-specified operation within a device. Decoded from bit 17 of the IOT.	Used for I/O skip instructions to test a device flag or other control functions. Cannot be used to read a device buffer register. In general, a designer should be wary of using IOP pulses for multiple purposes. Never clear and skip on a flag, with the same IOT, for example!
IOP 2H	2AE1	Same as IOP 1 H and it is also issued during a multicycle data channel transfer into memory. Decoded from bit 16 of the IOT.	Usually used to effect a transfer of data from a selected device to the processor or memory, to clear a device register, but may be used for other control functions. May not be used to determine a skip.
IOP 4 H	2AH1	Same as IOP 1 H and it is also issued during a multi- or single-cycle data channel transfer out of memory. Decoded from bit 15 of the IOT.	Usually used to effect transfer of data from the CPU or memory to the device or control. May not be used to determine a skip condition or to effect a transfer of data from a selected device to the CPU.

Table 4-2 (Cont)
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
I/O PWR CLR H	2AS1	System clear signal generated in response to: 1) Power on or off 2) CAF instruction 3) I/O RESET key 1 mHz, 250-ns pulse width	This signal is treated as an initializing signal for all devices (controllers) attached to the I/O bus. All registers are reset to "initial" status.
I/O RUN H	2BB1	This level becomes high when the IPU is running.	Can be used to disable a device if the CPU stops.
I/O SYNC H	2AB1	The I/O processor clock pulse issued every microsecond; 1 mHz, 250-ns pulse width.	This signal is used to synchronize device control timing such as API RQ and DC H RQ to the I/O processor.
PROG INT RQ L	2AL1	This signal can cause the program to CAL to location 000000. The instruction resident in location 000001 is fetched and executed.	A device delivers this level to the I/O processor to request interruption of the program in progress in order that the device be serviced.
RD RQ L	2AM1	Indicates to the processor that the device is sending it a data word.	Used by the device to specify to the I/O processor an input-to-CPU data transfer is required.
RD STATUS H	2AP1	A signal issued when the CPU issues an IORS instruction or when the console switch is placed on I/O STATUS.	Used by the device to gate its status onto the I/O bus data lines (one line per status bit) which is then read into the AC of the CPU.
SD0 H	2AT2	The first of two subdevice select lines decoded from bit 12 of the IOT instruction.	This signal and DS1 H can be decoded by the device for mode selection.
SD1 H	2AV2	Same as SD0 H except it is decoded from bit 13 of the IOT instruction.	Same as SD0 H
SING CY	2AS2	Indicates when a device wants to carry out a single -cycle data transfer to memory.	This device uses this line to request from the I/O processor a single-cycle transfer. If a DCH RQ signal is sent with it, then the I/O processor responds to an input (to computer) transfer. Otherwise it determines an output transfer.
SKIP RQ L	2AJ1	The return of the signal to the I/O processor during IOP 1 indicates that an IOT instruction test for a skip condition has been satisfied. The PC is subsequently incremented by one.	Used by a device to inform the program of the state of its interrupt flag. Also used in Single Cycle breaks as address line 01.

Table 4-2 (Cont)
I/O Bus Signal Functions

Signal Mnemonic	Connector Pin Number	Signal Definition	Signal Function
WR RQ L	2BB1	Indicates to the I/O processor that the device requires a transfer from memory during a multicycle data channel. Also used during Single Cycle breaks as address line 02.	The device uses this signal to inform the I/O processor that it wants a word from memory (during a multicycle data channel transfer). Also used during Single Cycle breaks as address line 02.
+1 → CA INH L	2BE1	If the I/O processor sees this signal during multicycle transfers, it inhibits normal incrementing of the device's assigned current address memory location.	This facility is used by such peripherals as DECtape and magnetic tape when they search for records. It is also useful during device checkout.

4.7 I/O BUS DEVICE PRIORITY AND SYNCHRONIZATION

In DCH and API, (described in Paragraph 6.4), a priority exists among the eight devices which may be placed on the DCH or API level 0, 1, 2 or 3 lines. The device closest to the I/O processor is given priority. The M104 Multiplexer Module is used in determining the priority by issuing the DCH or API request and controlling the device during the transfer.

An enable signal is daisy-chained from device to device on the bus. An enable signal to a device allows its request to be raised. When a device raises a request, it disables the enable transmitted to the next device on the bus, clearing or inhibiting the request of any device further down the bus. The enable is disabled until the action requested is completed.

The synchronization of a device starts when the device raises its device flag. If the enable (EN IN) is true the REQ will be set at the next I/O SYNC. Approximately 1 μs later, a GRANT signal will be sent out on the I/O bus and will load the contents of the REQ flip-flop into ENA flop. This microsecond is used to allow the EN OUT signal that was taken away, to propagate down the bus and clear any other requests which may have been set. ENA is used to gate the address from the device onto the I/O ADDRESS lines. ENB which is set one microsecond after ENA is used in Multicycle DCH breaks to specify the data transfer.

Figure 4-7 is a simplified description of the M104 logic. Timing is shown in Figure 4-8.

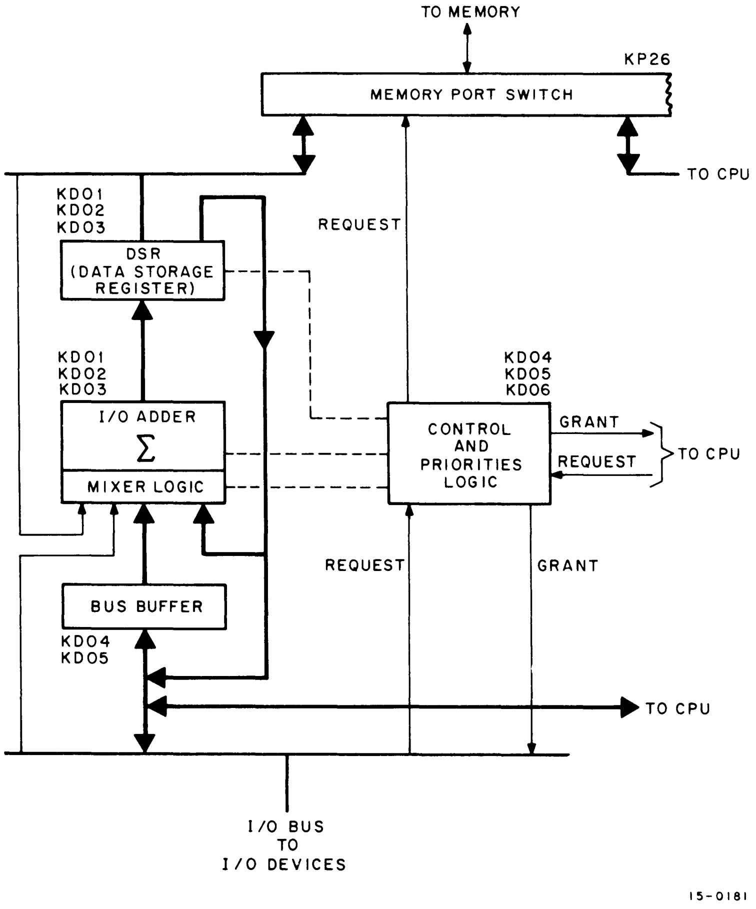


Figure 4-6 Data Channel Block Diagram

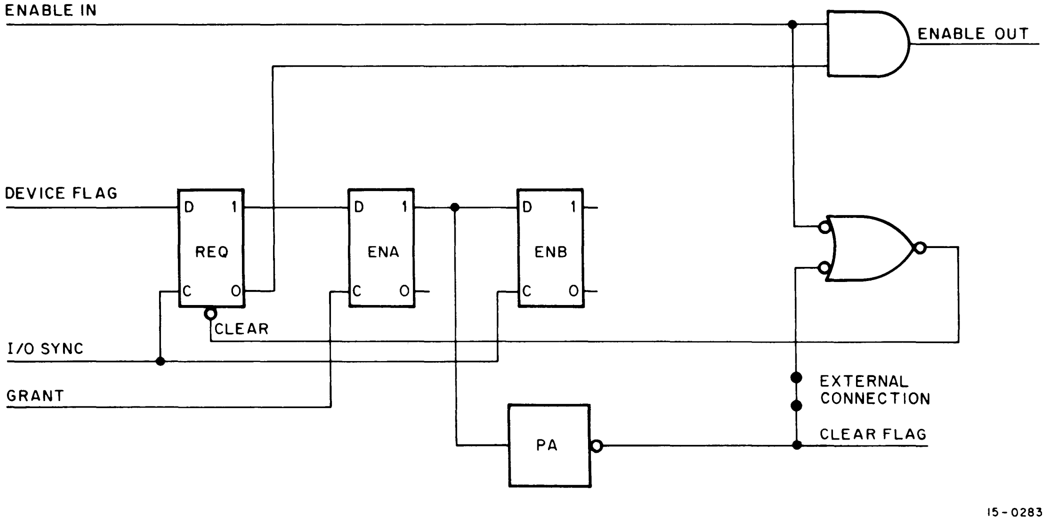


Figure 4-7 Simplified M104 Module Diagram

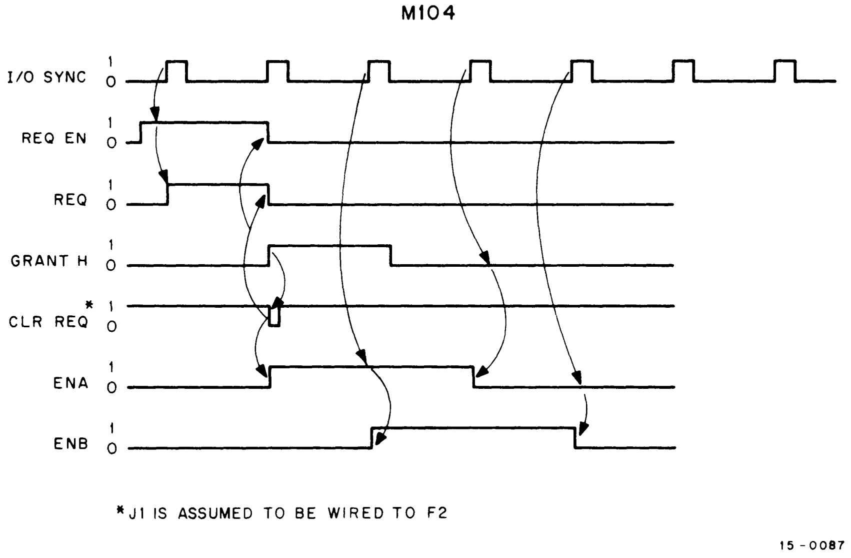


Figure 4-8 M104 Timing Diagram

4.8 SINGLE CYCLE INPUT TRANSFERS

In utilizing this type of transfer, a device must first be synchronized to the I/O processor as described in Paragraph 4-3. Both address and data must then be transmitted over the I/O bus to the I/O processor. The processor then makes a memory request to store data.

In performing this operation (refer to Figure 4-9), the address from the device is loaded into the DSR register and the data into the I/O buffer register. A memory write request is generated and, when the ADR ACK signal is returned from memory, the I/O buffer is loaded into the DSR, the DSR enabled on the MDL, the MRLS signal is sent back to memory and the I/O cycle is terminated. Figure 4-10 is a flow diagram of single cycle input transfers.

If the active device maintains the request signal when the address is strobed into the DSR, a BACK to BACK transfer will be performed and another GRANT will be sent out to the device, instead of terminating the cycle. The device can change its address and data when GRANT is removed from the bus, but it must have both enabled to the bus when GRANT becomes true again. Other devices are prevented from syncing, because I/O SYNC is disabled from the bus. Figure 4-11 is a flow diagram of single cycle back-to-back transfers.

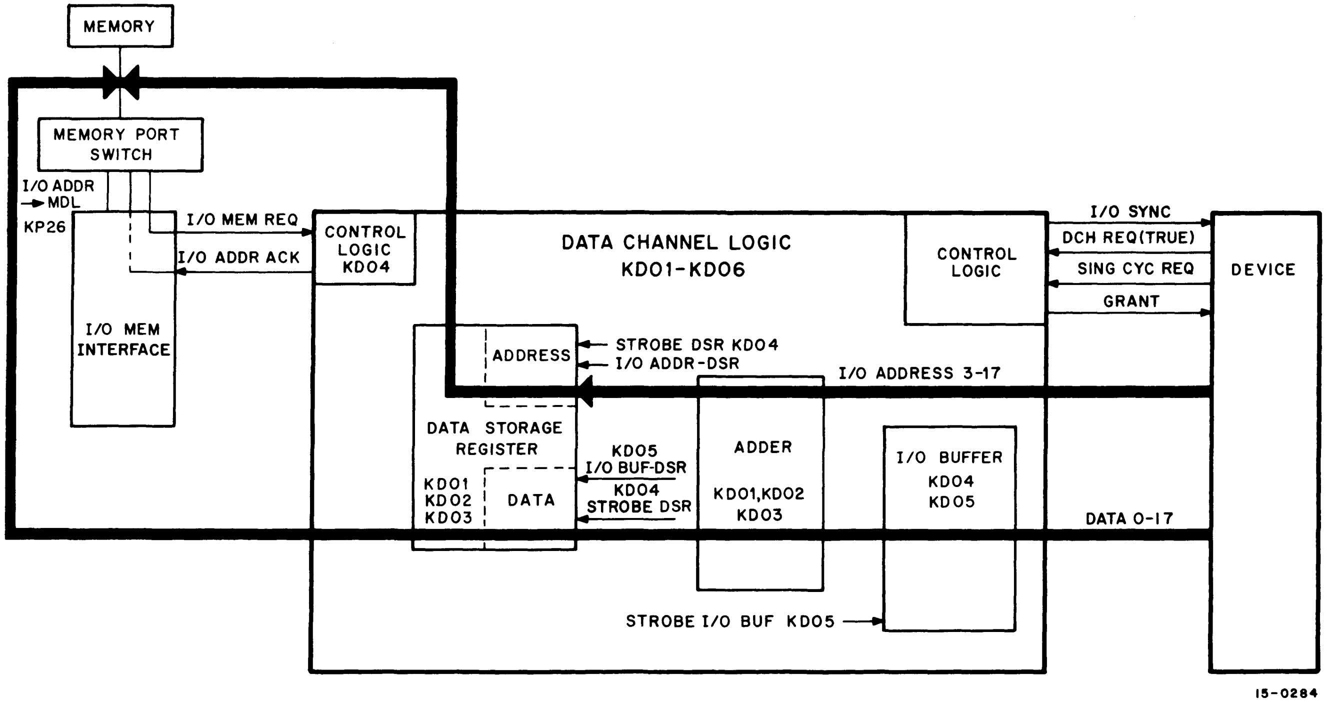


Figure 4-9 Single Cycle Data In Transfer, Block Diagram

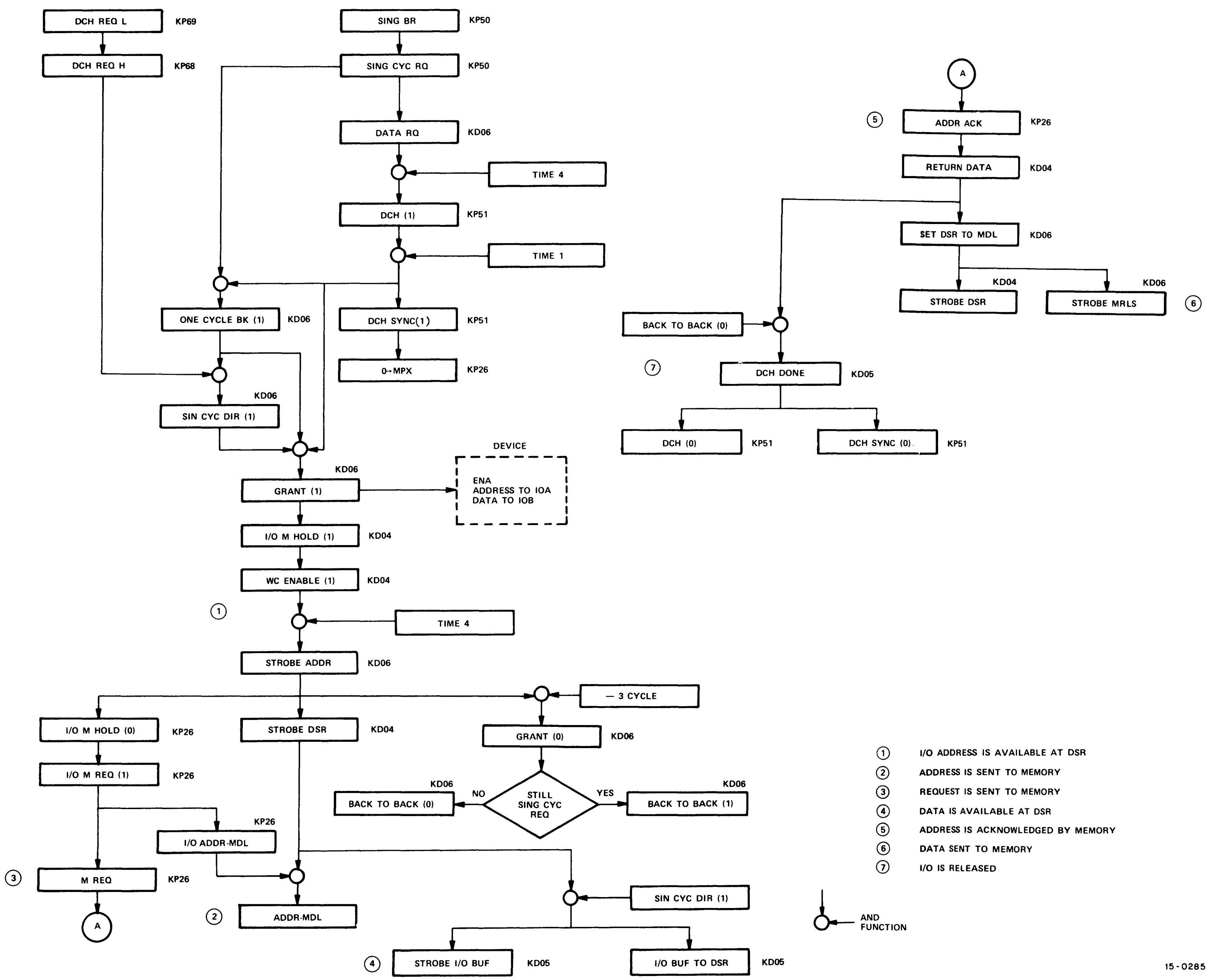


Figure 4-10 Single Cycle Data In Transfer, Detailed Flow Chart

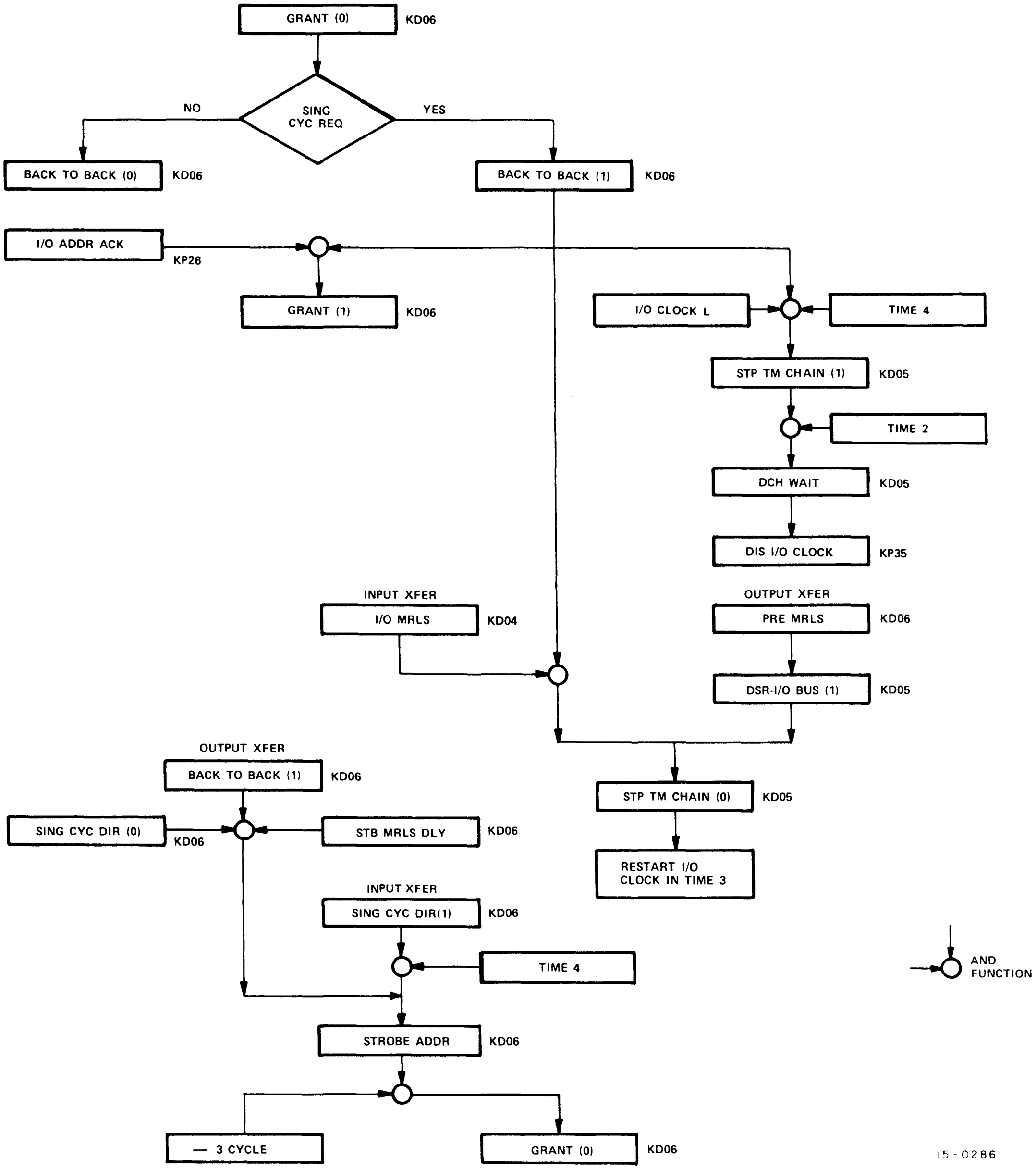


Figure 4-11 Single Cycle Back-To-Back Transfer, Detailed Flow Chart

After the device has taken all the breaks required (device word count register overflows), it may interrupt the I/O processor through the PI or API system.

4.9 SINGLE CYCLE OUTPUT TRANSFERS

For this type of transfer, a device, after synchronizing with the I/O processor, transmits its address over the I/O address lines. The I/O processor requests memory, reads the data from memory, and transmits the data back out to the device.

Figure 4-12 shows a block diagram of this type of transfer, in which the address is loaded into the DSR, a memory read request is made, the DSR is loaded with the memory data when RD RST is received, and then enabled on the I/O bus, and an IOP 4 is generated to load the data in the peripheral device.

Back-to-back transfers are detected in a similar manner as input transfers. Figure 4-13 is a detailed flow diagram of single cycle output transfers.

4.10 MULTICYCLE INPUT TRANSFERS

Multicycle transfers rely on word count and current address registers located in core memory. For input transfers, the device specifies the word count location by an address on the I/O address lines and places the data on the I/O bus. The I/O processor increments both the word count and the current address location and writes the data into the memory location specified by the incremented current address location. This takes 3 I/O processor cycles and 3 memory cycles.

Figure 4-14 shows a block diagram of multicycle input transfers. The word count address, specified by the device on the I/O address lines, is loaded into the DSR, and a memory read/pause/write cycle is requested. One is added to the data read from memory, the result is loaded into the DSR, and then is rewritten into memory. This is called the word count cycle. The address on the I/O address lines is then incremented by one, to point to the current address location, loaded into memory, and another read/pause/write cycle is requested. The data from memory is incremented by one, loaded into the DSR, and written back into memory. This is the current address cycle. The number now in the DSR points to the address into which data is to be written. Data from the device is loaded into the I/O buffer during the current address cycle by IOP 2. A memory write request is now made, and then ADR ACK is received from memory, the I/O buffer is loaded into the DSR, and the data written into memory. The I/O cycle is then terminated.

If the word count register has overflowed when it was incremented (there was a carry), an I/O OFLO signal is sent to the device to prevent it from requesting another break. The current request in progress is completed. Refer to Figure 4-15 for a timing diagram and Figure 4-16 for a detailed flow diagram.

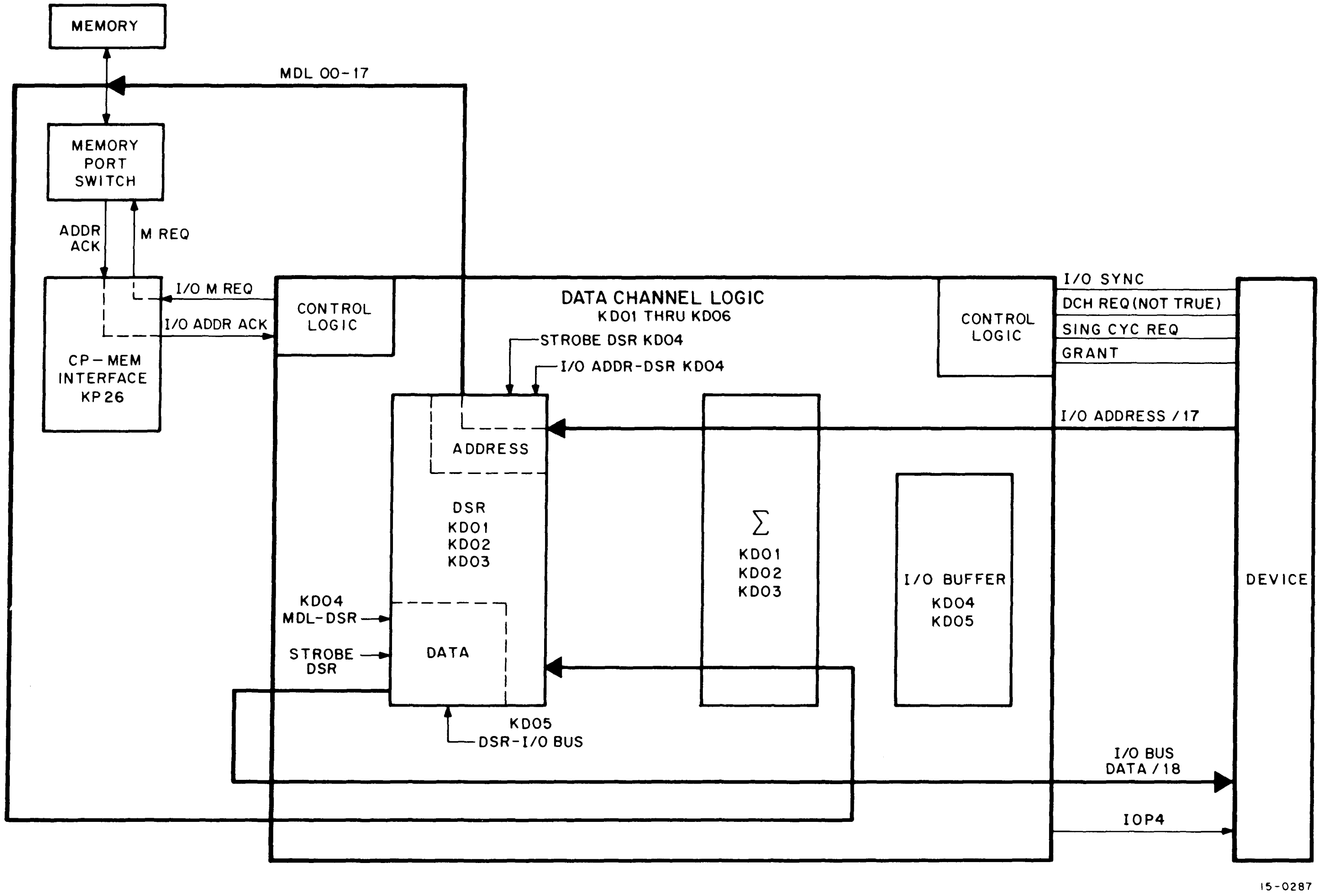


Figure 4-12 Single Cycle Data Out Transfer, Block Diagram

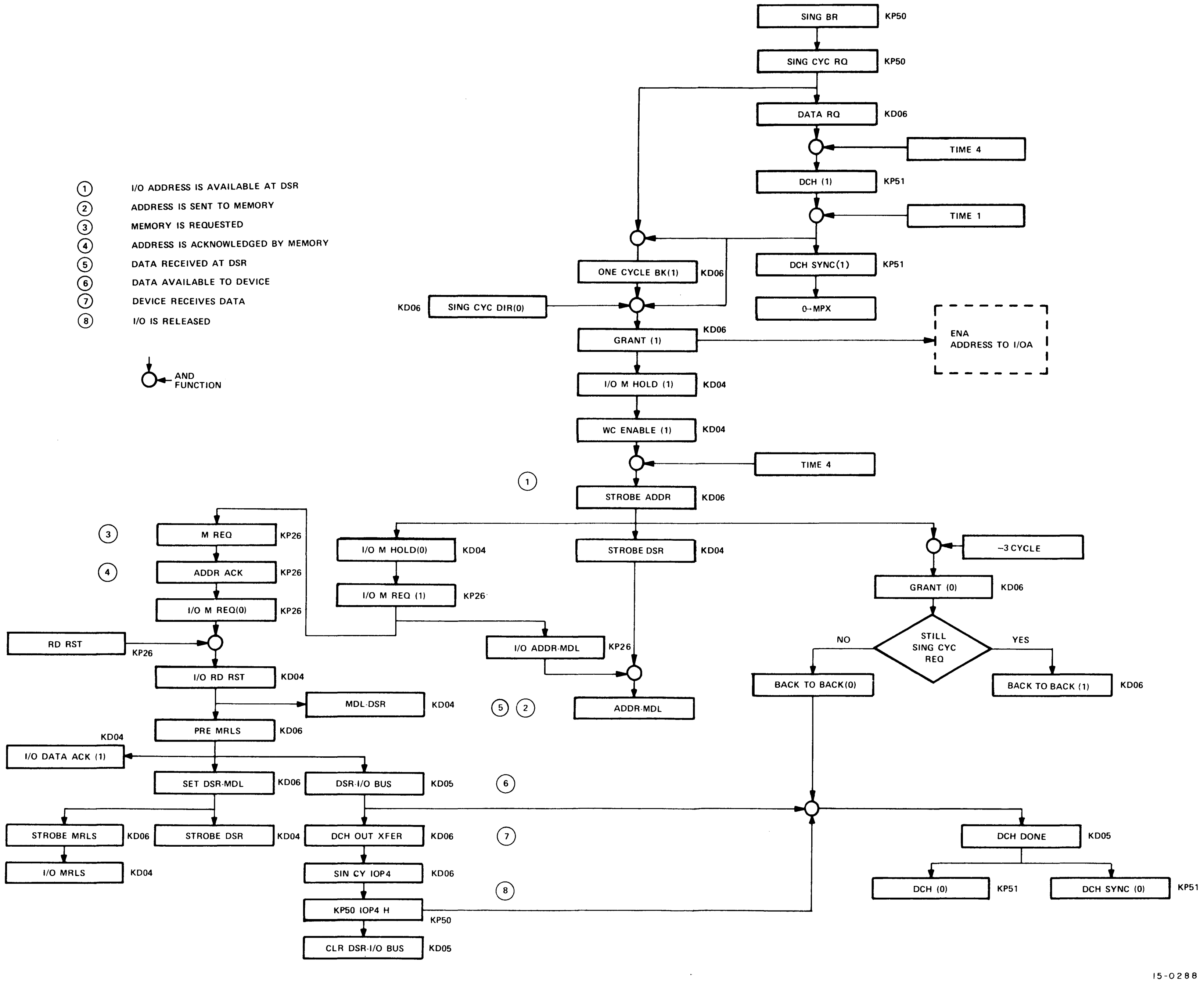


Figure 4-13 Single Cycle Data Out Transfers, Detailed Flow Chart

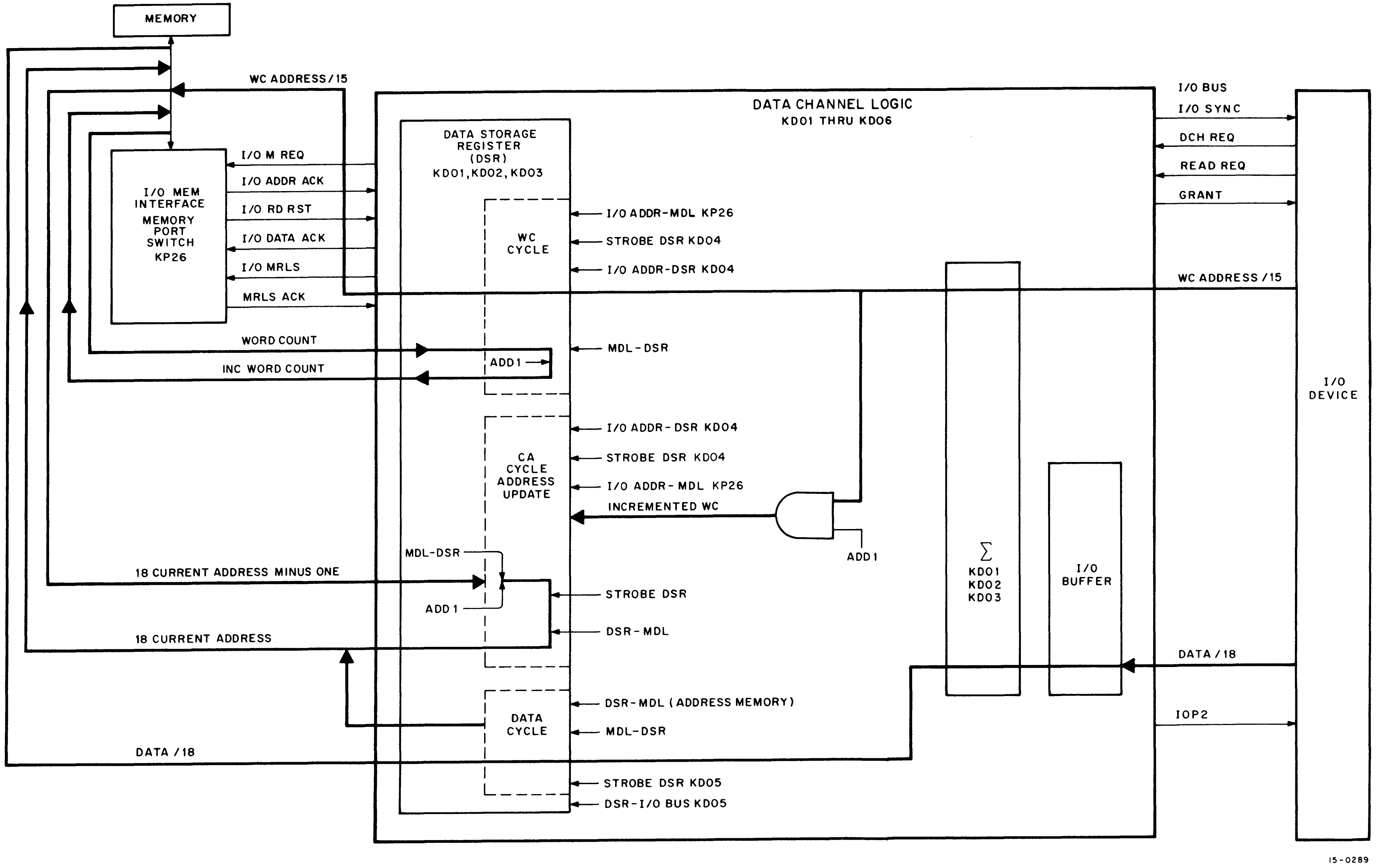


Figure 4-14 Multicycle Data In Transfer Block Diagram

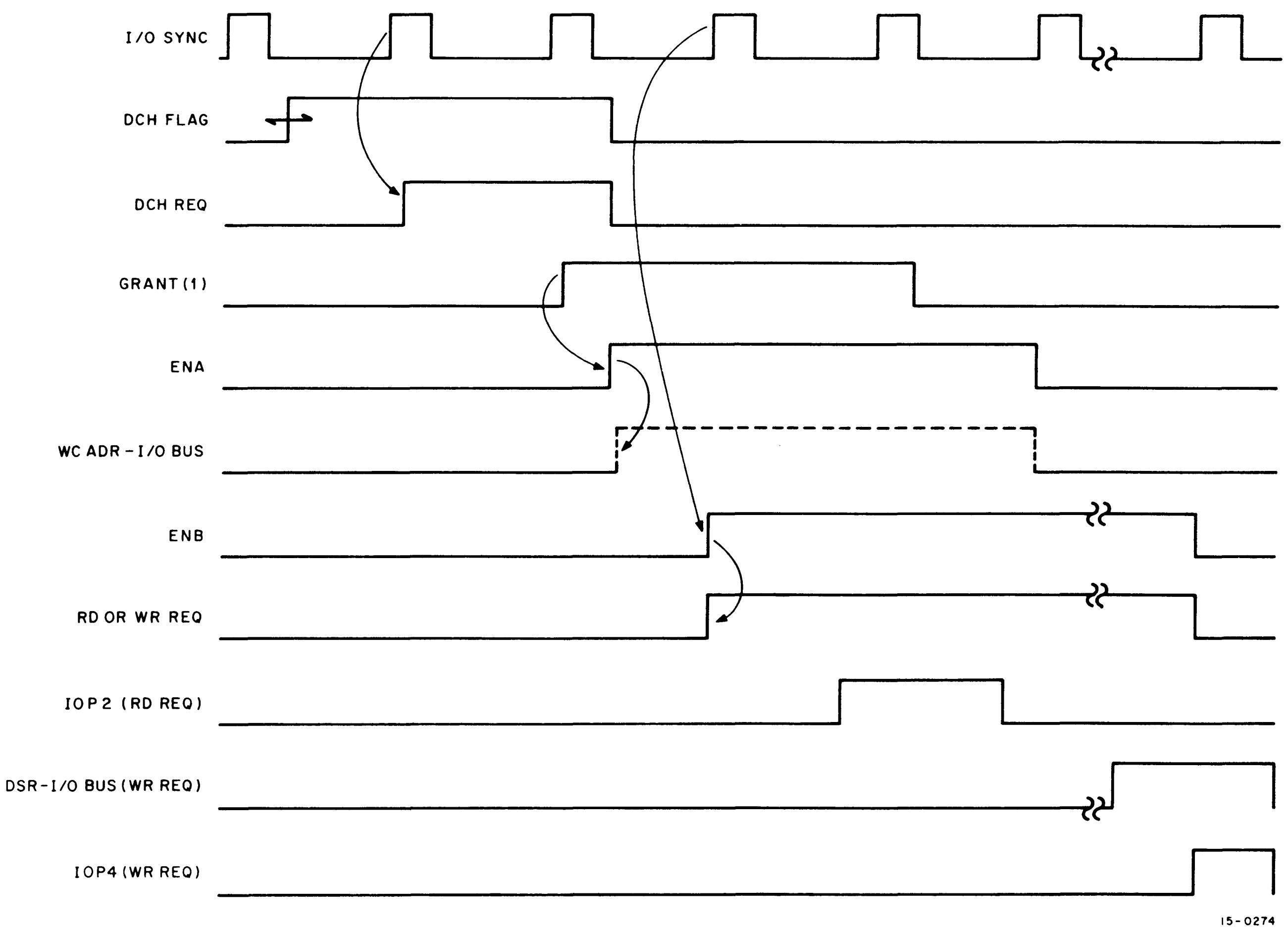


Figure 4-15 Multicycle Timing Diagram

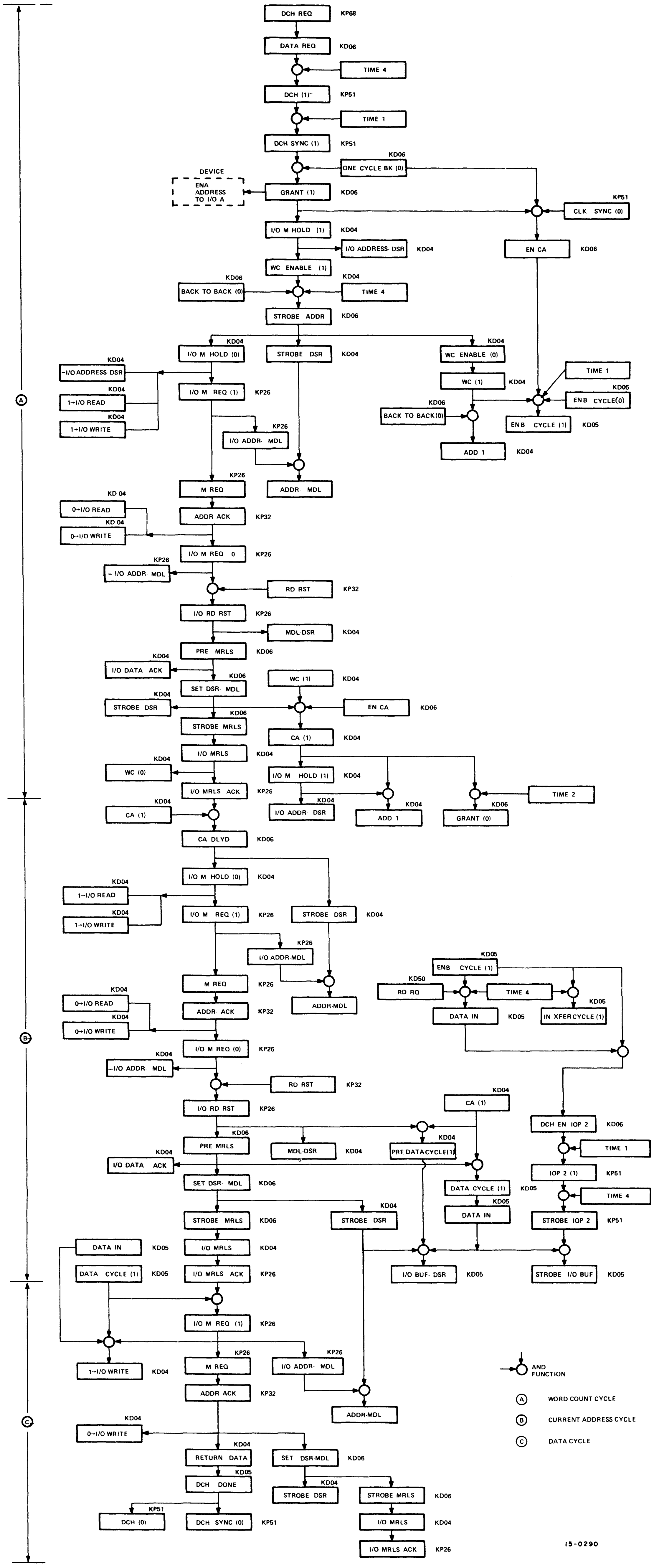


Figure 4-16 Multicycle Data In Transfer, Block Diagram

4.11 MULTICYCLE OUTPUT TRANSFERS

Output transfers use the word count and current address location as the input transfer. The data is fetched from memory and loaded into the device in the third cycle.

Both the word count and current address cycles are similar to the input transfers. Figure 4-17 is a block diagram of the multicycle output transfer. During the data cycle a memory read request is made, and the I/O processor stopped. When the data is read from memory into the DSR, the I/O processor is restarted, data in the DSR put onto the I/O bus, and an IOP 4 generated to load the data into the device.

Figure 4-18 is a detailed flow chart of the multicycle output transfers.

4.12 ADD TO MEMORY

This feature is similar to the multicycle transfers but differs in that, during the data cycle, data from memory is added to data from the device, and written into memory and transmitted to the device.

A read/pause/write cycle is requested in the data cycle and, when data is available from memory, it is enabled through one input on the DSR adder while the I/O buffer is enabled through the other and the DSR is then loaded with the result. The data is then rewritten into memory, enabled onto the I/O bus, and an IOP 4 is generated to load the device buffer if desired.

This type of cycle is performed by the device enabling both RD RQ and WR RQ on the I/O bus.

4.13 INCREMENT MEMORY

This feature increments a memory location in one I/O processor cycle. Only the word count cycle of a multicycle break is performed. The location specified by the address on the I/O address line is incremented by one. Enabling the INC MB line on the I/O bus along with DCH REQ will cause this type of cycle.

4.14 INHIBIT INCREMENT THE CURRENT ADDRESS

This line on the I/O bus inhibits the current address from being incremented. ADD ONE is not turned on during the data portion of the current address cycle.

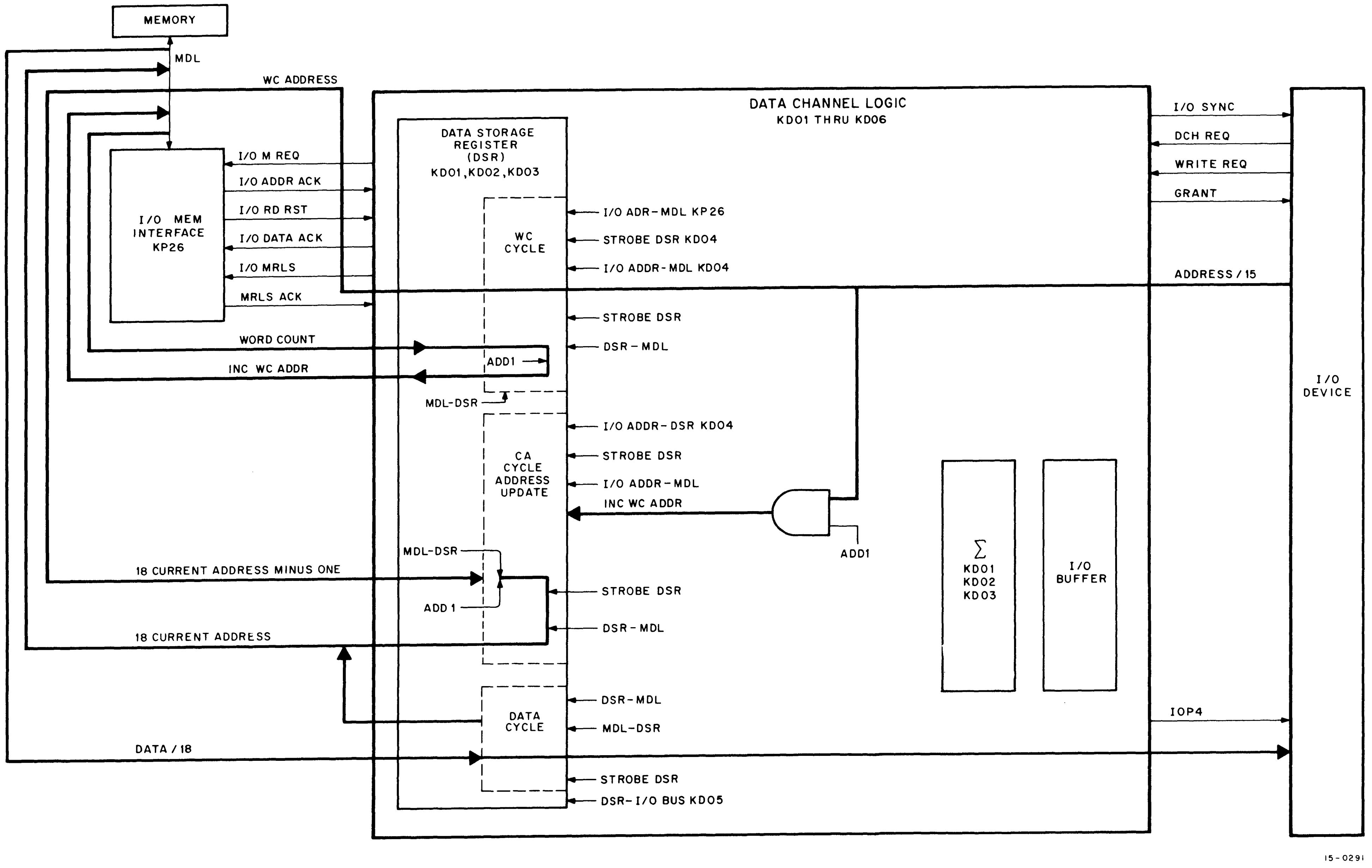


Figure 4-17 Multicycle Data Out Transfers, Block Diagram

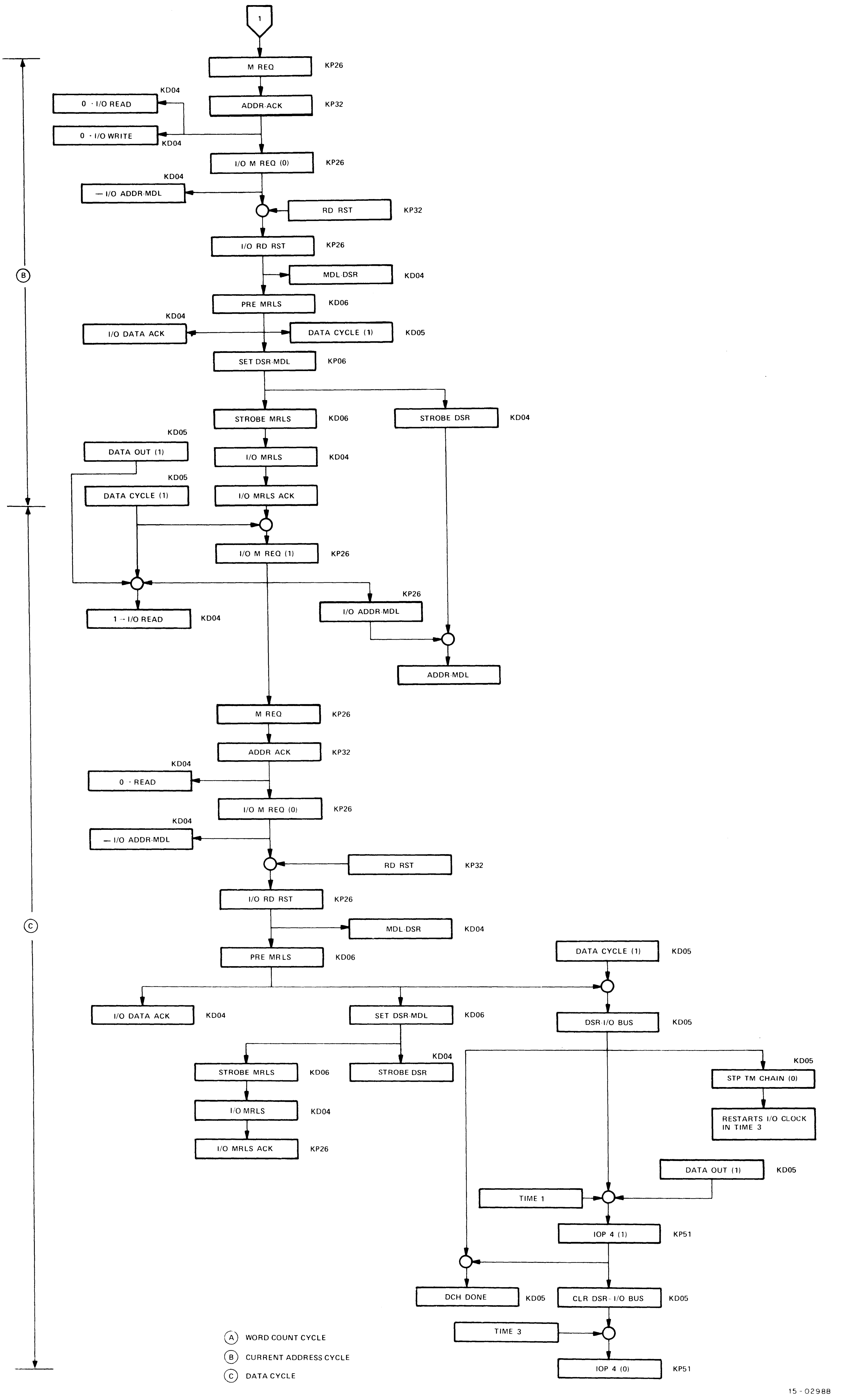


Figure 4-18 Multicycle Data Out Transfer, Detailed Flow Chart, Sheet 1 of 2.

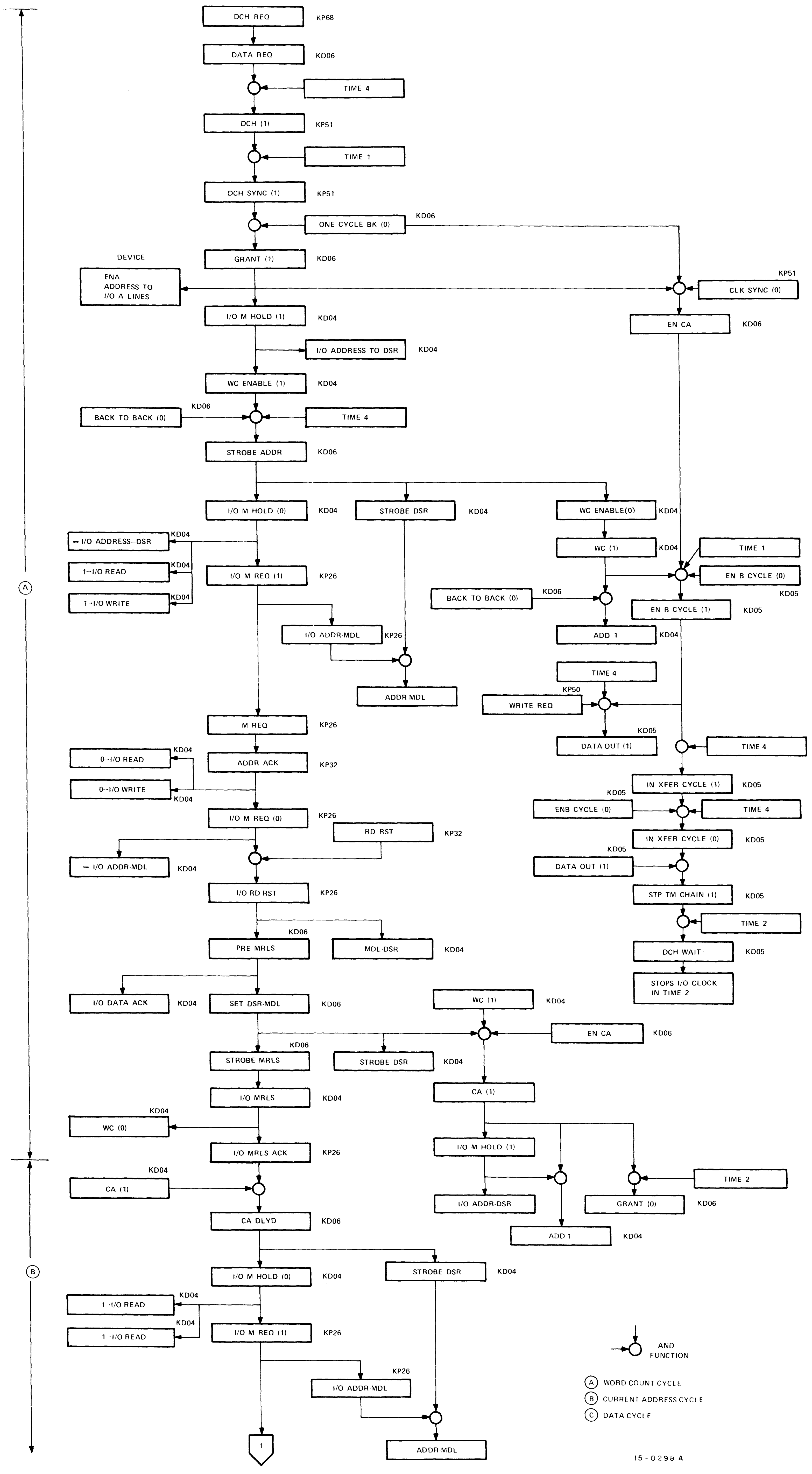


Figure 4-18 Multicycle Data Out Transfer, Detailed Flow Chart, Sheet 2 of 2.

4.15 DATA CHANNEL LATENCY

Latency is defined as the amount of time which is required to transfer data after the request is made by a device. When a worst case latency time is stated, it is assumed that the requesting device has priority over all other devices on the I/O bus.

Worst case latency in the PDP-15 occurs when a request is granted to a multicycle output device. The worst case latency for a single cycle output device is 8.5 μs.

4.16 PROGRAM INTERRUPT

The program interrupt facility has two IOTs associated with it:

a.	ION	700042	Enable the PI
b.	IOF	700002	Disable the PI

When the PI is disabled, the computer does not respond to any program interrupt requests (PROG INT RQ). However, when the PI is enabled, an interruption of normal program flow will occur. Upon receipt of a PROG INT RQ, the computer proceeds to complete its present instruction before interrupting. At clock time of TS02, during the last cycle of an instruction, interrupt acknowledge (INTRPT ACK) is set. This causes RUN to be cleared at TS03, Phase 3, preventing the CP from continuing. During TS03, the I/O address lines are placed on the A Bus. The lines should contain all zeroes. At clock time of TS03, a PI REQ is raised. This is used in the Request Synchronizer described in Section 4.2. PI is set at Time 4 and PI SYNC at Time 1. On the next Time 4, INTERRUPT STB is issued which loads the MO with the I/O address, sets INTERRUPT STATE, and sets START RUN allowing the CP timing to continue. The IR is cleared at TS01, forcing the CP to do a CAL. However, the CAL is to location zero, because the MO was loaded with zero. A flow diagram of PI/CPU interaction is in Chapter 6.

CHAPTER 5
POWER DISTRIBUTION

5.1 GENERAL

The power distribution system for the basic PDP-15 and 32K of memory is contained in the CP/IO mainframe. It comprises a 715 Power Supply with bulk regulation, regulation at the logic end, and a power monitoring network. It is designed to offer a high percentage of heat dissipation at the power supply end, and be able to supply each logic rack with maximum load-to-voltage response. The power monitoring network was designed to protect the logic and assist troubleshooting with the loss of dc power.

Figure 5-1 is a block diagram that shows the power distribution of the cabinet. The following paragraphs will explain each block.

Figure 5-1 Power Distribution Block Diagram

5.2 715 POWER SUPPLY

The 715 Power Supply has a ferro-resonant transformer and has the capability of accepting 110 Vac or 220 Vac, 50/60 cycle with small modification to the ac power control. (Tables are shown on the power supplies back door panel to aid in implementing such modifications.) An autotap with multiple outlets (P8 and P9) is provided for operating the required system fans.

The 715 Power Supply circuit schematic, D-CS-715-0-1, is shown in Volume 2.

5.3 AC CONTROL

Power protection is controlled at the ac inlet prior to the primary of the transformer. A two-line circuit breaker CB1, provides the main transformer T1 overcurrent protection. At the next node the ac lines are tied to the primary of transformer T2. The next component on the ac line is a 35 A capacity, double line break relay K1 for remote control of power on/off. At the next node the ac lines are tied to the main transformer's T1 primary winding, and the autotap plugs. A dual ac outlet is connected through a noise filter network to this same node to provide power for Teletypes plus remote power control in other cabinets.

Relay K1 is operated by remote control at the console end. Twelve volts are used to energize the primary coil of the relay. This voltage is supplied by step down transformer T2. The secondary winding is fused (F13) and is in series to the relay coil and connections that are wired to the console switch shown in drawing D-CS-5408392-0-1 in Volume 2.

Switch SW1, connected in parallel with the console switch connections in the power supply provides a console lock out feature. To complete this feature a second section of this switch, a ground signal, is wired to an outlet plug (P2, pin 6). A wire run to the console, completes the connection, and locks out the console control switches. This feature prevents the operator from disturbing the running program.

Transformer T2, besides supplying the relay 12 volts, supplies 6 volts ac and -6 volts dc. The 6 Vac is used in the real time clock option and the -6 Vdc is used to bias the console switch card.

5.4 DC POWER SOURCE

The secondary network of the main transformer T1 is comprised of four full wave rectified windings and one resonator winding (part of the ferroresonant network). The four windings produce +11V with 80 to 85A capacity, -11V, -30V, and +30V each having 55.5A maximum capacity.

All four raw voltage sources are used for the system logic needs. The +11V is used primarily to bias all the +5V local voltage regulator units (module G821). The -11V is used to supply a 6V negative bias for the sense amplifiers on the memory G100 modules through a negative voltage regulator (module G822). The regulator also supplied the threshold voltage for the sense amplifiers. The -30V is used to supply a -24V potential for producing x,y and inhibit currents through a voltage regulator and a passive element (modules G823 and G825 respectively). The +30V is used to power the console lights.

All the windings are accurately fused for burn out protection at the power supply. A table is provided on the rear door panel of the 715 Power Supply to facilitate a correlation between the potential voltage system racks, power supply plug, fuse holder, and fuse capacity.

CAUTION

All fuse capacities noted on table must be used for maximum system protection.

5.5 BULK REGULATION

A bulk regulation network is incorporated to distribute a maximum of 80A to the system from the +11V source and keep the IR drops, line reflections, and ac ripple at a minimum. The network consists of three bulk regulator circuits each capable of supplying 30A maximum.

The three regulators provide an 8.1V potential and are accurate from no load to full load. Each 8.1V is used to supply the current need through the local regulators (G821 +5V regulator module). One added advantage of using +8.1V instead of +11V is that it minimizes power dissipation in the series pass element at the local regulator. The harness distribution allows the 8.1V to be distributed to the system in 10A chunks. Each 10A chunk is fused at the power supply for maximum system protection.

As well as having fused 10A chunks, each bulk regulator has an overcurrent protection in case of power supply short circuits. This overcurrent protection is fused through circuit breaker CB2 for each regulator. The circuit breaker reset switch houses a 3-pole single throw with the three poles ganged together. This switch is located on the front door panel of the power supply.

The bulk regulators are modular in form and are part of the subassembly of the power supply. Each regulator has a large heat sink attached to it, with one fan blowing air past the three units. This same subassembly contains the two rectifier diodes for the +11V. The fan must be running properly at all times to ensure proper heat dissipation. Care must be taken not to block the air path by placing objects on top of the power supply screen. The screen cover must never be removed from the power supply for any length of time, to protect hands and prevent foreign objects from falling into the power supply.

5.6 POWER HARNESS

The power harness in this system is built for minimum IR drop and maximum reflection immunity. Each hot line sent to the logic has a ground return line back to the power supply and is paired in a twisted form. The wire size used is an AWG #12, and throughout the system not more than 10A flow through any one line. The wire distribution supplies a total of 16 voltage regulators in a 32K memory system, console power, real time clock option, and the systems fan assembly.

Volume 2, drawing D-IC-PDP15-0-14 shows a pictoral sketch of the power distribution between the power supply and the logic racks.

5.7 LOCAL REGULATION

Local regulation is incorporated to supply the transient current needs with maximum voltage stability possible. An added advantage of local regulation in each rack is isolation from the other racks and minimization of noise flow on the power bus throughout the system.

In a 32K memory system there is a total of eight G821 modules (+5V), four G822 modules (-6V), and four G823 modules with each having a pass element module G825 (-24V). The G821 modules supply a maximum of 7A each and have overvoltage protection and undervoltage/overcurrent protection. The G822 modules supply 2A maximum each and have zero voltage and overvoltage protection. The G823 modules supply 4A maximum each and have zero voltage and overvoltage protection.

Descriptions of these four modules are provided in the PDP-15 Systems Module Manual. Their adjustment procedures may be found in Volume 2, drawing D-BS-MM15-0-20.

5.8 POWER MONITOR NETWORK

The power monitor network is incorporated to detect abnormal operation of any voltage regulator and, upon detection, shut down the -30V source (memory voltage). This, in turn, sequentially halts the computer, and turns the console PWR light off.

The network in the mainframe and back door consists of a small amount of circuitry on each regulator module, and a signal line which reaches each regulator slot with a receiver at the CP end to monitor the line. Each +5V regulator has an open collector inverter on the module that ties the line to ground if any of the following voltage errors occur:

a. The +5V line is shorted to ground. Result: high current drain, so a +8V fuse blows out and the low voltage detector turns on.

b. Excess amount of voltage on +5V, $\geq$5.5V.

b. (Cont) Results: SCR turns on and crowbars the +5V line which blows out a +8V fuse and turns on the low voltage detector.

c. Low voltage on +5V ≥ 4.75V. Results: low voltage detector turns on.

d. In the memory section only, when the -6V or the -24V regulator has a shorted output to ground or zero voltage. Results: the respective fuse blows out for both regulators with a short circuit only, but in both situations a ground signal is sent to the respective +5V regulator (turning on the power-not-OK inverter).

e. In the memory section only, when the -6V has an excess amount of voltage, ≥6.5V. Results: The SCR turns on and crowbars the -6V line, which blows out the -11V fuse and sends a ground signal to the +5V regulator (turning on the power-not-OK inverter).

f. In the memory section only, when the -24V has an excess amount of voltage, ≥26V. Results: an operational amplifier switches polarity and a ground signal is sent to the +5V regulator (turning the power not OK inverter on).

Refer to a complete block diagram of the power monitor network in Volume 2, drawing D-BS-KP15-0-81.

As well as placing a ground on the signal line by each +5V regulator, each regulator has an inverter which controls a light on the back panel of each rack. When power in the system is OK all the lights are on. If one rack should show trouble, the respective light will go out, providing aid to quick troubleshooting. If the +11 V, which is a bias voltage for all the regulators, goes to zero, all the lights will go out.

The remaining portion of the network comprises a redrive section on the G827 module, a power line from the logic rack to the power supply, and a relay control in the power supply for the -30V source. The G827, located in the central processor, monitors the signal line observing the regulators and redrives the signal noninverted. The output is an open collector driver that draws current through a relay at the power supply. Once a ground is detected by the redrive network the relay (K2) in the power supply is energized. This in turn opens the -30V source line not allowing memory X, Y, and Inhibit currents to occur while the system is having power problems. The G827 also turns off the PWR light on the console for troubleshooting assistance.

CHAPTER 6
OPTIONS

6.1 KE15 EXTENDED ARITHMETIC ELEMENT (EAE)

The KE15 Extended Arithmetic Element (EAE) option facilitates high-speed multiplication, division, shifting, normalizing, and register manipulation.

The EAE enables fast, flexible, hardware execution of the following signed or unsigned functions.

a. Shifting the contents of the primary arithmetic registers (AC MQ) right or left, requires 2.9 to 5.2 µs.

b. Normalizes the quantity in the primary arithmetic registers, i.e., shifts the contents left to remove leading binary zeros (or ones in the case of negative numbers) for the purpose of preserving as many significant bits as possible. The time required is 2.9 to 5.2 µs.

c. Multiplication is performed in 2.75 to 6.4 µs.

d. Division including integer divide and fraction divide, require 2.75 to 6.6 µs. Divide overflow indication is furnished by the link when signed division produces a quotient exceeding 377777_8 in magnitude, or unsigned division produces a quotient exceeding 777777_8 in magnitude.

e. Basic setup instructions to manipulate the data in the registers prior to execution of the above instructions require 1.3 µs.

6.1.1 General Operation

Control logic for the KE15 option consists of a 6-bit event time counter, an instruction decoder, bus control circuitry, load signals, an instruction register, quotient detection circuitry, and other control flip-flops. In addition to this control logic there is an 18-bit MQ/shift register.

During event time A, AC modifications occur; MQ transfers happen at time B; AC transfers occur at time C; nothing takes place at time D. All shifting occurs at time E; complementing takes place at event time F for multiply and divide operations if necessary. For further information refer to EAE flow drawing KE06.

There are two types of EAE instructions: setups and non-setups. Setups use only four event times whereas non-setups use all six.

Figure 6-1 is a general flow diagram for EAE instructions. Table 6-1 lists EAE instructions and Table 6-2 lists EAE microinstructions.

Table 6-1
EAE Instructions

Octal Code *	Mnemonic	Operation
640000	EAE	Basic EAE instruction. Acts as an NOP instruction.
640001	OSC	Inclusive-OR the SC with the AC.
640002	OMQ	Inclusive-OR the MQ with the AC.
640004	CMQ	Complement the MQ.
641001	LACS	Load AC12 through 17 with the contents of the SC.
641002	LACQ	Load the AC with the contents of the MQ.
644000	ABS	Get the absolute value of the AC.
650000	CLQ	Clear the MQ.
652000	LMQ	Load the MQ with the contents of the AC.
664000	GSM	Get the sign and magnitude of the AC.
6405XX	LRS	Long right shift.
6605XX	LRSS	Long right shift, signed.
6406XX	LLS	Long left shift.
6606XX	LLSS	Long left shift, signed.
6407XX	ALS	Accumulator left shift.
6607XX	ALSS	Accumulator left shift, signed.
640444	NORM	Normalize.
660444	NORMS	Normalize, signed.
6531XX	MUL	Multiply.
6571XX	MULS	MULTIPLY, signed.
6403XX	DIV	Divide.
6443XX	DIVS	Divide, signed.
6533XX	IDIV	Integer divide.
6573XX	IDIVS	Integer divide, signed.
6503XX	FRDIV	Fraction divide.
6543XX	FRDIVS	Fraction divide, signed.

* "XX" indicates the number to be loaded into step counter and depends upon the number of shifts required or answer precision required.

Table 6-2
EAE Microinstructions

Bit	Binary Code	Function
4	1	Enters AC00 into the Link for signed operations.
5	1	Clears the MQ.
6	1	Reads AC00 into the EAE SIGN register prior to a signed multiply or divide operation.
6, 7	10	Takes the absolute value of the AC after the AC00 bit is read into the EAE SIGN register.
7	1	Inclusive-ORs the AC with the MQ and places the result in the MQ.
8	1	Clears the AC.
9, 10, 11	000	SETUP instruction code. Accompanies code in bits 15, 16, 17.
9, 10, 11	001	MUL instruction code.
9, 10, 11	010	Unused instruction code.
9, 10, 11	011	DIV instruction code.
9, 10, 11	101	LONG RIGHT SHIFT instruction code.
9, 10, 11	110	LONG LEFT SHIFT instructions code.
9, 10, 11	100	NORMALIZE instruction code.
9, 10, 11	111	ACCUMULATOR LEFT SHIFT instruction code.
12-17		Specifies the step count for all EAE codes (9-11) except SETUP.
15	1	For SETUP instruction code only, complements the MQ contents.
16	1	For SETUP instruction code only, inclusive-ORs the MQ with the AC and places the result in the AC.
17	1	For SETUP instruction code only, inclusive-ORs the AC with the SC and places the result in the AC.

6.1.2 Normalize Instructions

The NORM and NORMS instructions are commonly used within a subroutine to convert an integer into a fraction and exponent for use in floating-point arithmetic. The algorithm for normalize is to shift the contents of the AC and MQ left until AC00 differs with AC01. For signed, normalize positive numbers, this results in AC00 (0) and AC01 (1). For signed, normalized numbers the sign (AC00)

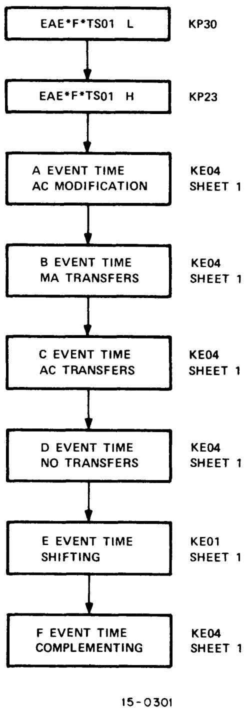


Figure 6-1 EAE General Flow Diagram

is first duplicated in the Link. For unsigned numbers the LINK is usually initialized to 0. In both cases the content of MQ00 enters AC17, the content shifted out of AC00 is lost, and the content of the LINK enters MQ17, on each shift. When shifting halts, the contents of the SC reflects the number of shifts executed to reach the normalized condition. The SC contents are available through the use of the EAE OSC or EAE LACS instruction.

For normalized numbers, the binary point is assumed to be between AC00 and AC01. The value of the exponent is in the SC. The number in the SC, after normalization, is actually the sum of the pre-established characteristic and the exponent (n) in 2's complement form. The characteristic is a number equivalent to the total number of bit positions in the AC and MQ, 36_{10} or 44_8. The NORMS instruction contains this number in bits 12 through 17 and loads it into the SC in 2's complement to establish the exponent in excess 44 code. This means that the exponential range of the fraction when normalized is 2^0 to 2^{35}, or $-44_8 + n$.

For example, if the integer +3 is stored in the MQ (MQ16, MQ17 are 1s) and it is desired to convert this to a fraction and exponent, the following program sequence is required.

```
NORM(S)       /NORMALIZE CONTENTS OF AC, MQ
DAC           /DEPOSIT AC IN MEMORY
LACQ          /MOVE MQ TO AC
DAC           /DEPOSIT MQ IN MEMORY
LACS          /MOVE SC TO AC
TAD (44       /SUBTRACT CHARACTERISTIC FROM STEP COUNT
DAC           /DEPOSIT RESULT (EXPONENT) IN MEMORY
```

In the process of normalization, a total of 33 shifts is required to shift MQ16(1) into AC01. This leaves the SC with a step count of:

```
011100  initialized step count
100001  plus 33 steps
------
111101  final step count
```

Because the step count is in 2's complement, the TAD 44_8 instruction (2's complement add) in effect subtracts the characteristic from the final step count to arrive at the exponent:

```
111101  final step count
100100  TAD Characteristic
------
100001  exponent
```

In order to save the contents of the SC after a NORM instruction if an interrupt occurs, an interrupt is held off for 2 cycles after a NORM instruction. This allows time to store the SC. Restoration of the step counter requires that the 2's complemented quantity, taken from the SC at the time of interrupt, be complemented, then combined with the pseudo-NORM instruction. The step count following the TAD, AND operation below is one less (1's complement) than the actual value produced by the previous normalization (2's complement). Execution of the pseudo-NORM instruction, then, 2's complements the step count into the SC, and in shifting the AC and MQ left one bit position adds the necessary 1 to the SC to produce the correctly restored step count (the 6404XX present in the AC from TAD, AND operation shifts to become 501XXX).

Restoration program

```
LAC SCSAVE
XOR (77          /COMPLEMENT STEP COUNT
TAD (640402      /DEVELOP PSEUDO-NORM
AND (640477      /DELETE POSSIBLE STEP COUNT OVERFLOW
DAC .+1          /PLACE NORM IN SEQUENCE
HLT              /STEP COUNT TO SC
LAC MQSAVE
LMQ              /LOAD THE MQ
LAC ACSAVE       /LOAD THE AC
DBR              /RESTORE PC, LINK, ETC.
JMP I SUBENTR
```

6.1.3 MUL(S) Instruction

The MUL(S) instruction multiplies the contents of the AC (multiplier) by the contents of the next sequential core memory location (multiplicand) to form a product in the AC and MQ.

Bits 12 through 17 in the instruction are usually programmed for a step count of 22_8 (18_{10}), representing the multiplication of one 18-bit quantity (the sign bit and 17 magnitude bits for MULS) by another to produce a 36-bit product. When full precision is not required, the step count may be decreased by subtracting the appropriate number n from the instruction code. The product is always scaled 18-n from MQ17. If n is programmed in the instruction, the 18-n lower bits in the long register are meaningless.

For a MUL instruction the link must previously have been initialized to 0. During the preparatory phase, the multiplier is transferred from the AC to the MQ, the AC is cleared, and the step counter (SC) is set to the 2's complement of bits 12 through 17 of the instruction. A core memory cycle reads the multiplicand into the MI. The arithmetic phase, executed as multiplication of one unsigned quantity by another (binary point of no consequence), halts when the SC counts up to 0 (see Figure 6-2).

For a MULS instruction, a previous LAC/GSM/DAC CAND sequence stores the absolute value of the multiplicand in memory and places the original sign of the multiplicand in the link. During the preparatory phase of MULS, a memory cycle reads the multiplicand into the MI, compares the link (sign of multiplicand) with AC00 (sign of multiplier) and sets the product quotient flip-flop if they differ and resets the link. The multiplier is transferred from the AC to the MQ and is complemented if negative, the AC is zeroed, and the SC is initialized. The arithmetic phase is complete when the SC counts to 0. AC00 and AC01 each receive the sign of the product; the remaining AC and MQ bits represent the magnitude. If initially the multiplier and multiplicand had had unlike signs (P/Q neg (1)), then the resulting MQ after the arithmetic operation would have been complemented.

The algorithm for multiplication using the EAE is simple: add and shift right. Each bit of the multiplier is sampled, starting with the least significant bit. If the sampled bit is a 1, the multiplicand is added to the partial product. The partial product and the multiplier are then shifted right one position for the next multiplier bit sampling. If the sampled bit is a zero, zeros are added to the partial product. With each shift the contents of the least significant bit are lost. Multiplication ends when the SC, up-counted with each shift, reaches 0.

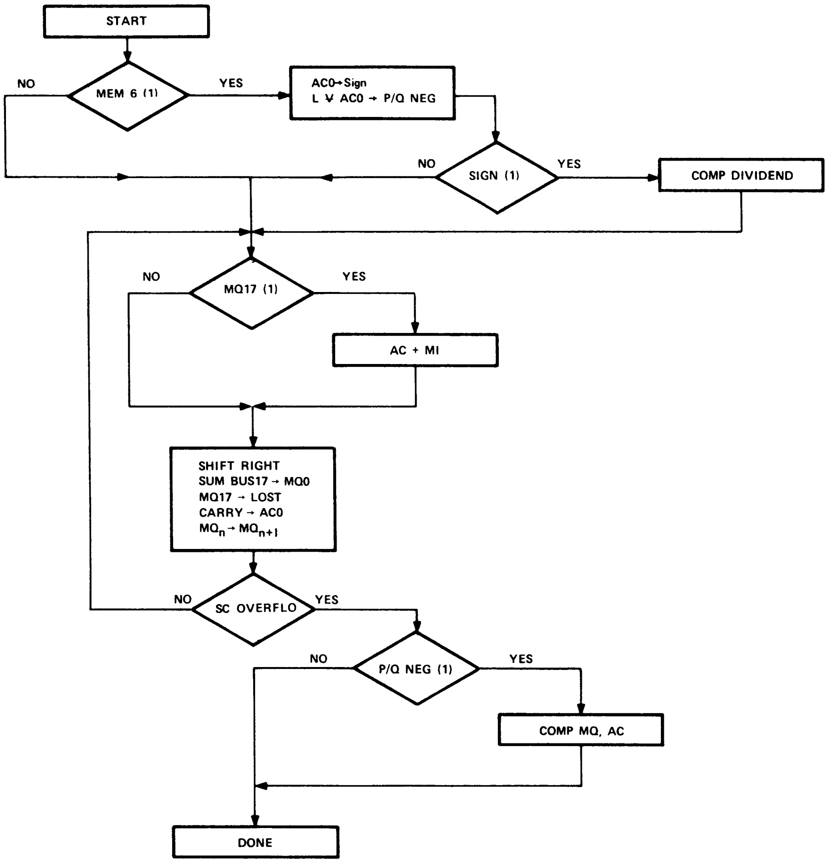


Figure 6-2 EAE Multiply

Programming Example

Multiply $2_8 \times 5_8$

```
START   200   200100   LAC CAND       /Load multiplicand into AC
        201   100500   JMS MPY        /Store main program address in 500 and jump
                                        to MPY subroutine
        202   200101   LAC PLIER      /Load multiplier into AC
        203                           /Main program re-entry

MPY     500   000202   PC             /Main program address
        501   664000   GSM            /Absolute value in AC
        502   040505   DAC.+3         /Deposit CAND in 505
        503   420500   XCT I MPY      /Load multiplier into AC
        504   657122   MULS           /Fetch |CAND| and multiply

        505   000002
        506   440500   ISZ PC         /Increment main program address
        507   620500   JMP I 500      /JMP to main program
CAND    100   000002   MULTIPLICAND
        101   000005   MULTIPLICAND
```

L	AC	MQ (multiplier)	MB (multiplicand)	SC	OPERATION
0	0000	0101	0010	1100	
	0010				ADD
	0010				
0	0001	0010	0010	1101	SHIFT
	0000				ADD
	0001				
0	0000	1001	0010	1110	SHIFT
	0010				ADD
	0010				
0	0001	0100	0010	1111	SHIFT
	0000				ADD
	0001				
0	0000	1010	0010	0000	SHIFT
		ANSWER = 12_8			

Tables 6-3 through 6-15 illustrate the operations that take place in each instruction.

Table 6-3
EAE NOP 640000

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	MQ → MQ MQ1→C L EAE LD MQ H	------ KE04, Sheet 1 KE04, Sheet 2
TS02	C	AC → AC EAE ENAB AC L EAE LD AC L	------ KE04, Sheet 2 KE04, Sheet 2
TS03	D	------	------

Table 6-4
OSC 640001

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	MQ → MQ MQ1→C L EAE LD MQ H	------ KE04, Sheet 1 KE04, Sheet 2
TS02	C	EAE SC→C L AC → C BUS EAE LD AC	KE04, Sheet 1 KE04, Sheet 2 KE04, Sheet 2
TS03	D	------	------

Table 6-5
OMQ 640002

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	Same as NOP, Event Time B	------
TS02	C	C-MQ2 L EAE ENAB AC L EAE LD AC L	KE04, Sheet 1 KE04, Sheet 2 KE04, Sheet 2
TS03	D	------	------

Table 6-6
CMQ 640004

CP State	Event Time	Functions	Drawings
TS02	A		------
TS02	B	MQ1→C L EAE COMPCAL EAE LD MQ H	KE04, Sheet 1 KE04, Sheet 2 KE04, Sheet 2
TS02	C	Same as NOP, Event Time C	------
TS03	D	------	------

Table 6-7
LACS 641001

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	Same as NOP, Event Time B	------
TS02	C	EAE SC→C L EAE LD AC L	KE04, Sheet 1 KE04, Sheet 2
TS03	D	------	------

Table 6-8
LACQ 641002

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	Same as NOP, Event Time B	------
TS02	C	MQ2→C L EAE LD AC L	KE04, Sheet 1 KE04, Sheet 2
TS03	D	------	------

Table 6-9
ABS 644000

CP State	Event Time	Functions	Drawings
TS02	A	EAE ENAB AC L EAE COMP ABS *A L IF AC00 EAE LD AC L	KE04, Sheet 2 KE04, Sheet 2 KE04, Sheet 1 KE04, Sheet 2
TS02	B	Same as NOP, Event Time B	------
TS02	C	Same as NOP, Event Time C	------
TS03	D	------	------

Table 6-10
CLQ 650000

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	MQ1→C L EAE LD MQ H	KE04, Sheet 1 KE04, Sheet 2
TS02	C	Same as NOP, Event Time C	------
TS03	D	------	------

Table 6-11
LMQ 652000

CP State	Event Time	Functions	Drawings
TS02	A	------	------
TS02	B	EAE ENAB AC L EAE LD MQ H	KE04, Sheet 2 KE04, Sheet 2
TS02	C	Same as NOP, Event Time C	------
TS03	D	------	------

Table 6-12
GSM 664000

CP State	Event Time	Functions	Drawings
F*TS02	A	EAE ENAB AC L EAE COMP C→A L ABS*A L, if AC01(1) EAE LD AC L, if AC00(1), SET LINK	KE04, Sheet 2 KE04, Sheet 2 KE04, Sheet 1 KE04, Sheet 2
F*TS02	B	Same as NOP, Event Time B	------
F*TS02	C	Same as NOP, Event Time C	------
F*TS03	D	------	------

Table 6-13
LRS 6405XX and LRSS 6605XX

CP State	Event Time	Functions	Drawings
F*TS02	A	MEM4 (1) ∧ AC00(1) 1→L MI 12-17→SC LOAD SC	KP22 KE04, Sheet 2
F*TS02	B	Same as NOP, Event Time B	------
F*TS02	C	Same as NOP COUNT SC L (+1)	KE05
F*TS03	D	------	------
EAE*TS01 EAE*TS02	E	LRS DECODED L→AC0, $AC_n \rightarrow AC_{n+1}$ AC 17→MQ 00 $MQ_n \rightarrow MQ_{n+1}$ SC OVERFLO	KE04, Sheet 1 KP01-KP18 KE01 KE01, KE02 KE03, Sheet 1
EAE*TS03 EXECUTE	F	------ ------	------ ------

Table 6-14
LLS 6406XX and LLSS 6606XX

CP State	Event Time	Functions	Drawings
F*TS02	A	Same as LRS	------
F*TS02	B	Same as LRS	------
F*TS02	C	Same as LRS	------
F*TS03	D		------
EAE*TS01 EAE*TS02	E	LLS decoded L → MQ 17 $MQ_n \rightarrow MQ_{n-1}$ MQ00 → AC 17 $AC_n \rightarrow AC_{n-1}$ SC OVERFLO	KE04, Sheet 1 KE02 KE01, KE02 KP18 KP17 KE03, Sheet 1
EAE*TS03	F	------	------
EXECUTE	------	------	------

Table 6-15
ALS 6407XX and ALSS 6607XX

CP State	Event Time	Functions	Drawings
F*TS02	A	Same as LRS	------
F*TS02	B	Same as LRS	------
F*TS02	C	Same as LRS	------
F*TS03	D	------	------
EAE*TS01 EAE*TS02	E	ALS is decoded L → AC 17 $AC_n \rightarrow AC_{n-1}$ SC OVERFLO	KE04, Sheet 1 KP18 KP01-KP17 KE03, Sheet 1
EAE*TS03	F	------	------
EXECUTE	------	------	------

Table 6-16
NORM 640444 and NORMS 660444

CP State	Event Time	Functions	Drawings
F*TS02	A	Same as LLS	------
F*TS02	B	Same as LLS	------
F*TS02	C	Same as LLS	------

Table 6-16 (Cont)
NORM 640444 and NORMS 660444

CP State	Event Time	Functions	Drawings
F*TS03	D	Same as LLS	------
EAE*TS01 EAE*TS02	E	NORMS decoded Same as LLS 1-F when normalized	KE04, Sheet 1 KE03, Sheet 2
EAE*TS03	F	------	------

Table 6-17
MULS 6531XX and MULS 6571XX

CP State	Event Time	Functions	Drawings
F*TS02	A	MI 12-17-SC LD SC MI 06 (1)*AC00 (1) = 1-SIGN MI 06 (1)*L≠AC00 == 1-P/Q NEG	KE03, Sheet 1 KE04, Sheet 2 KE05 KE05
F*TS02	B	If SIGN(0): EAE ENAB AC EAE LD MQ H If SIGN (1): EAE ENAB AC EAE COMP C-A L EAE LD MQ H	 KE04, Sheet 2 KE04, Sheet 2 KE04, Sheet 2 KE04, Sheet 2 KE04, Sheet 2
F*TS02	C	If MI 08 (1): 0-AC	------
F*TS03	D	EAE ENAB AC L EAE LD AC L COUNT SC L (+1)	KE04, Sheet 2 KE04, Sheet 2 KE05
EAE*TS01 EAE*TS02	E	EAE MUL decoded If MQ17(1): EAE MI-B L EAE ENAB AC L EAE SHIFT RT H If MQ17 (0): EAE ENAB AC L EAE SHIFT RT L L-AC00 SUM BUS 17-MQ00 MQ_n - MQ_{n-1} SUM BUS_n -AC_{n-1} EAE LD AC L	KE04, Sheet 1 KE03, Sheet 2 KE04, Sheet 2 KE05 KE04, Sheet 2 KE05 KP01 KE01 KE01, KE02 KP02-KP18 KE04, Sheet 2

Table 6-17 (Cont)
MULS 6531XX and MULS 6571XX

CP State	Event Time	Functions	Drawings
EAE*TS02 (cont)		EAE LD MQ H SC OVERFLO	KE04, Sheet 2 KE03, Sheet 2
EAE*TS03	F	P/Q NEG (1): COMP MQ	KE04, Sheet 2

6.1.4 DIV(S) Instruction

The DIV(S) instruction divides the contents of the AC and MQ by the contents of the next sequential core memory location to form a quotient in the MQ and remainder in the AC.

For a DIVS instruction the link must previously have been set to 0 and remains 0 unless divide overflow occurs. During the preparatory phase, the step counter is set to the 2's complement of the step count in bits 12 through 17 of the instruction. Bits 12 through 17 of the instruction are usually programmed for a step count of 23_8 (19_{10}) unless full precision is not necessary. A core memory cycle takes place to read the division into the MI. The arithmetic phase, executed as the division of one unsigned quantity by another halts when the SC counts up to zero (see Figure 6-3).

For a DIVS instruction, a previous LAC/GSM/DAC/DUR sequence stores the absolute value of the divisor in memory and places the original sign of the divisor in the link. If AC00(1), the sign flip-flop is set; if. L is unequal to AC00, the product quotient flop is set. These flops, if set, act to 1's complement the MQ and AC during the preparatory phase and to perform other complementary functions during the arithmetic phase to arrive at the correctly signed quotient as shown on the EAE flow drawing KE15-0-06.

The algorithm for divide using the EAE is simple: add or subtract, and shift left. The divisor is first subtracted from the AC portion of the dividend, and the result is shifted left. If the result is a negative number (TEMP (1)), the divisor is added to the quotient; if the result is a positive number (TEMP (0)), the divisor is subtracted (2's complement add) from the quotient. The result is then shifted left one position for the next sampling. Divide overflow occurs, if in the first subtraction the divisor is not greater than the AC portion of the dividend. Divide overflow sets the link and stops the divide operation.

Programming Example

Divide $12_8 \div 5_8$

```
        500    200100    LAC DIVR        /Load divisor into AC
        501    100200    JMS DIV         /Store program address in 200 and jump
                                         /to DIV subroutine
        502                              /main program re-entry

DIV     200    502       PC              /program address
        201    664000    GSM             /Store DIVR sign in link and absolute value
                                         /in AC
        202    040207    DAC.+5          /Deposit DIVR in 207
        203    200101    LAC DIVD1       /Load half dividend into AC
        204    652000    LMQ             /Move to MQ
        205    200102    LAC DIVD2       /Load half dividend into AC
        206    644323    DIVS            /Fetch DIVR and divide
        207    000005
        210    620200    JMP I 200       /Return to main program

        100    000005    DIVR
        101    000012    DIVD1           /(Least significant)
        102    000000    DIVD2           /(Most significant)
```

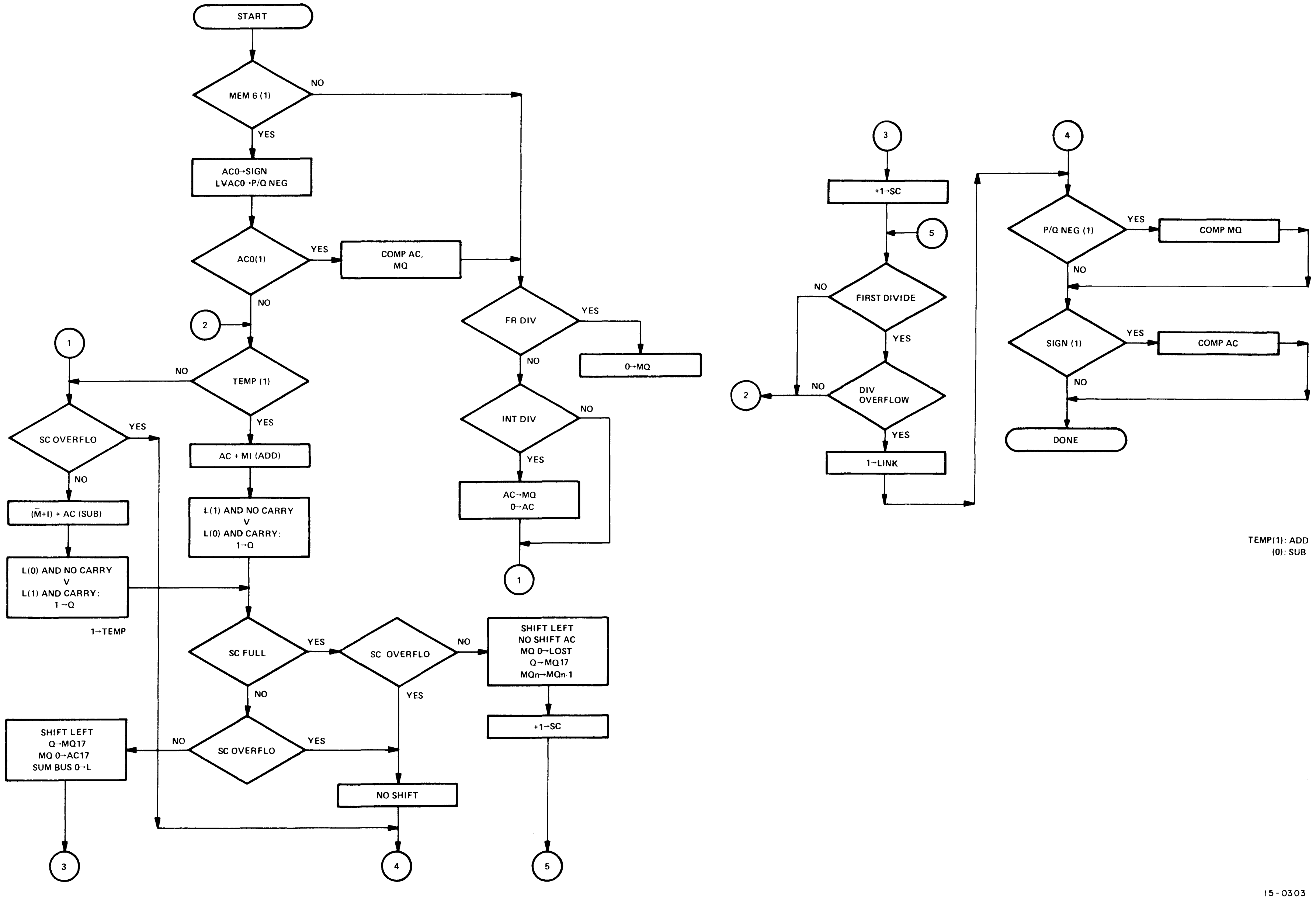
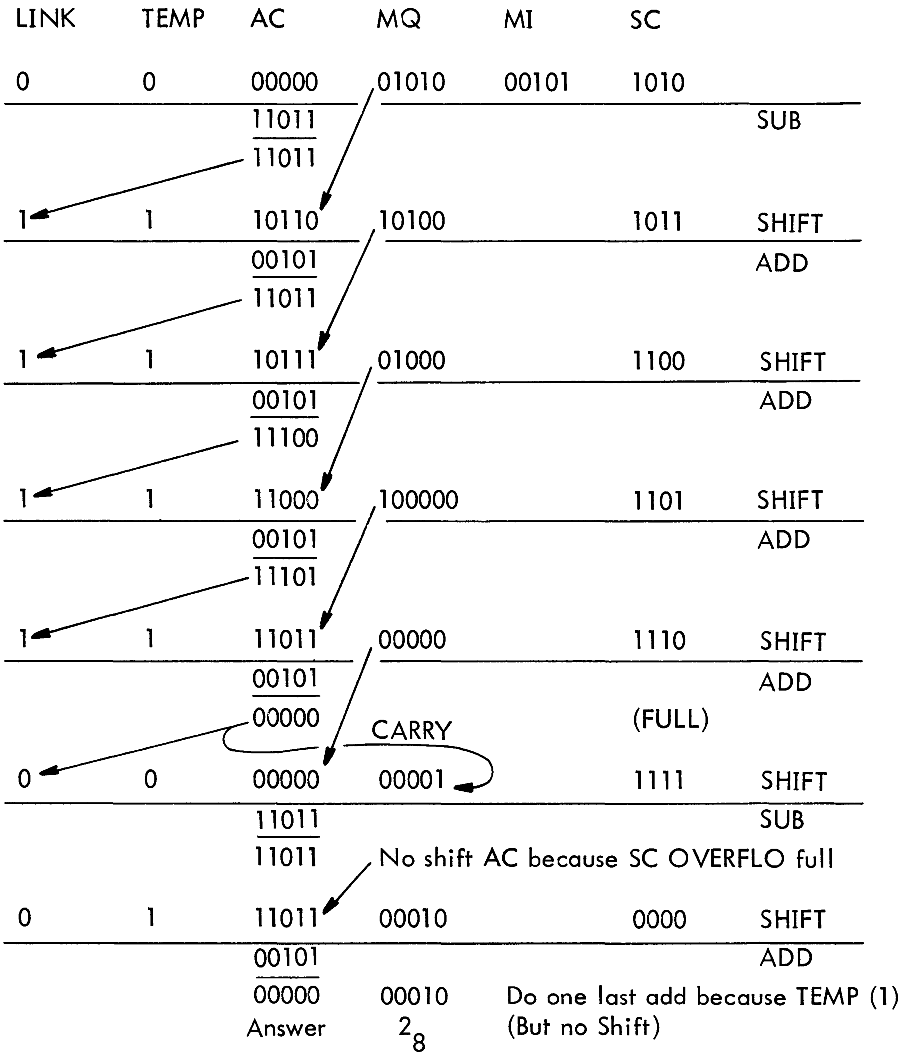


Figure 6-3 EAE Divide

Divide Example: 12 ÷ 5 = 2



```
LINK   TEMP   AC       MQ       MI      SC
0      0      00000    01010    00101   1010
              11011                             SUB
              11011
1      1      10110    10100            1011    SHIFT
              00101                             ADD
              11011
1      1      10111    01000            1100    SHIFT
              00101                             ADD
              11100
1      1      11000    100000           1101    SHIFT
              00101                             ADD
              11101
1      1      11011    00000            1110    SHIFT
              00101                             ADD
              00000    CARRY            (FULL)
0      0      00000    00001            1111    SHIFT
              11011                             SUB
              11011    No shift AC because SC OVERFLO full
0      1      11011    00010            0000    SHIFT
              00101                             ADD
              00000    00010    Do one last add because TEMP (1)
              Answer   2_8      (But no Shift)
```

Table 6-18
DIV 6403XX and DIVS 6443XX

CP State	Event Time	Function	Drawings
F*TS02	A	Same as MUL except if MI 06(1)*AC00(1) COMPAC: EAE ENAB AC L EAE COMP C-AL ABS*A L EAE LD AC L	 KE05, Sheet 2 KE05, Sheet 2 KE04, Sheet 1 KE04, Sheet 2
F*TS02	B	SIGN (1):COMP MQ C-MQ1 L EAE COMP, C- A L EAE LD MQ H	 KE04, Sheet 1 KE04, Sheet 2 KE04, Sheet 2

Table 6-18 (Cont)
DIV 6403XX and DIVS 6443XX

CP State	Event Time	Function	Drawings
F*TS02	C	COUNT SC L(+1)	KE05
F*TS03	D	---	---
EAE*TS01	E	Check for divide overflow:	
EAE*TS02		carry on first or second	KE03, Sheet 2
		if DIV overflow set LINK	KP22
		TEMP(0): EAE-MI -B L	KE03, Sheet 2
		TEMP(1): EAE MI-BL	KE03, Sheet 2
		EAE ENAB AC L	KE04, Sheet 2
		EAE SHIFT LEFT H	KE05
		SUM BUS 00- LINK	KP22
		SUM BUS_n $-\text{AC}_{n-1}$	KP01-KP18
		MQ_n-MQ_{n-1}	KE01, KE02
		Q-MQ 17	KE02
		Q-TEMP	KE05
		EAE LD AC L	KE04, Sheet 2
		EAE LD MQ H	KE04, Sheet 2
		SC OVERFLOW	KE03, Sheet 2
EAE*TS03	F	IF P/Q NEG(1): COMP MQ	KE04, Sheet 2
		IF SIGN(1): COMP AC	
		EAE ENAB AC L	KE04, Sheet 2
		MUL/DIV COMP H	KE04, Sheet 1
		EAE LD AC	KE04, Sheet 2

6.1.5 IDIV(S) Instruction (See Table 6-19)

The instruction IDIV(S) divides the contents of the AC (integer dividend) by the contents of the next sequential core memory location to form a quotient in the MQ and a remainder in the AC.

The arithmetic phase of the instruction(s) is identical to that of DIV(S). The preparatory phase transfers the contents of the AC to the MQ and clears the AC. Thereafter the arithmetic phase in reality performs the division on the long register dividend first as in DIV, with the exception of the most significant portion of the dividend (AC) is at 0.

Table 6-19
IDIV 6533XX and IDIVS 6573XX

CP State	Event Time	Function	Drawings
F*TS02	A	Same as MUL	
F*TS02	B	Same as MUL	
F*TS02	C	Same as MUL	
F*TS03	D		
EAE*TS01 EAE*TS02	E	Same as DIV	
EAE*TS03	F	Same as DIV	

6.1.6 FRDIV(S) Instruction (See Table 6-20)

The FRDIV(S) instruction divides the contents of the AC (fraction dividend) by the contents of the next sequential core memory location and forms a quotient in the MQ and a remainder in the AC.

The arithmetic phase of the instruction (S) is identical to that of the DIV(S). The preparatory phase clears the MQ. The arithmetic phase is a division of the long register with the MQ at 0. For FRDIV, the binary point is assumed at the left of AC00. For FRDIVS the binary point is assumed between AC00 and AC01.

Table 6-20
FRDIV 6503XX and FRDIVS 6543XX

CP State	Event Time	Function	Drawings
F*TS02	A	Same as DIV	
F*TS02	B	0-MQ: MQ1-C L EAE LD MQ H	 KE04, Sheet 1 KE04, Sheet 2
F*TS02	C	Same as DIV	
F*TS03	D		
EAE*TS01 EAE*TS02	E	Same as DIV	
EAE*TS03	F	Same as DIV	

6.1.7 Indicators

The 18-bit MQ register, 7-bit step counter (6-bits plus overflow), and an 18-bit EAE register are all accessible from the rotary indicator switch. The EAE switch position indicates the complement of 18 control functions. Table 6-21 lists these signals and their positions.

Table 6-21
EAE Indicators

Position	Signal
0	A (0) H
1	B (0) H
2	C (0) H
3	D (0) H
4	E (0) H
5	F (0) H
6	-SU H
7	-EAE MUL H
8	-EAE DIV SHIFT H
9	-EAE NORMS H
10	-EAE LRS H
11	-EAE LLS H
12	-EAE ACLS H
13	EAE SIGN (0) H
14	EAE P/Q NEG (0) H
15	DIV OVERFLOW (0) H
16	-EAE FULL H
17	EAE NO SHIFT H

6.1.8 EAE Execution Times

Set Up	1.32 μs
Shifts	2.75 μs + 130 ns/step
MUL + DIV	2.75 μs + 260 ns/step

6.2 KW15 REAL-TIME CLOCK

The KW15 Real-Time Clock option, when enabled, increments location 00007 at a rate specified by a clock in module slot L03. The M515 module, usually supplied with this option, increments every 16.7 ms in 60 Hz systems and every 20 ms in 50 Hz systems. An M401 variable clock may be used instead of the M515.

The IOT's for the Real-Time Clock are:

CLSF	700001	Skip if Clock flag is set

CLOF	700004	Clear Clock Flag and disable clock
CLON	700044	Clear Clock Flag and enable clock

When the CLON IOT is executed, the CLK EN flop is set (see KP57) this enables the CLK REQ flop to be set at the clock frequency if the console is locked or the CLK switch on the console is enabled (front of switch depressed). The CLK REQ is synchronized to the I/O as described in Chapter 4 and a DCH word count cycle is used to increment location 00007.

When location 00007 increments from all 1s to all 0s, a clock overflow occurs and the CLK FLAG is set. This flag is interfaced to the PI and API whose entry address is 51. The clock will continue to count up from zero after overflow until location 00007 is reinitialized or the clock is turned off. Figure 6-4 is a flow diagram of the Real-Time Clock operation.

6.3 KF15 POWER FAIL OPTION

The KF15 Power Fail option protects the PDP-15 computer system from loss of power by providing a means of storing important CP registers, upon loss of power, and the subsequent restoration of these registers and restarting of the program when power returns.

Whether the option is installed or not, the system is always power cleared when turned on or off. With the option installed, there are three ways in which power failures are handled.

If the console is not locked, the system acts exactly as though the power fail option is not installed. A power low condition is detected by the G827 on KP57 which clears the CP RUN flop at the end of the current instruction in progress. INT reset pulses are then generated until the logic cannot function. When power is restored, a series of INT reset pulses are produced as soon as there is enough power to supply the logic. This burst continues 250 μs after the POWER OK line goes positive. The single power fail IOT instruction is:

SPFAL	706201	Skip on power fail flag.

The other two modes of operation are with the console locked and the machine with PI or API enabled. When the power failure is detected as above, a power fail program interrupt request or API request is made.

In the case of API, a level 0 API request is made, synchronized, and location 52 is executed. A JMS should be in this location and the subroutine should store all CPU registers in memory, place a jump to a restart routine in location 0 and then halt.

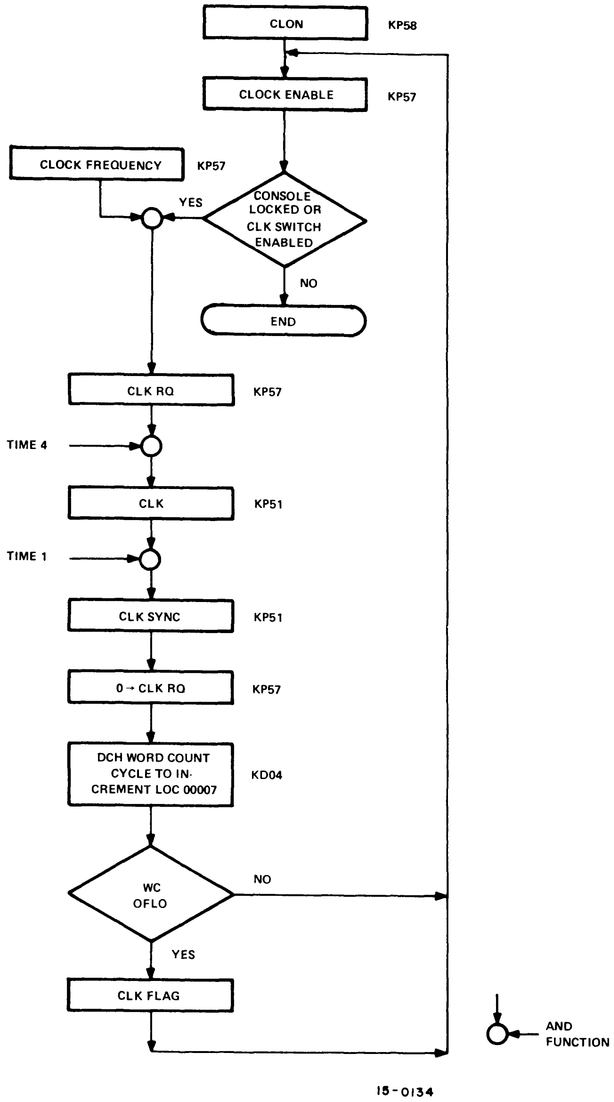


Figure 6-4 Real-Time Clock, General Flow Diagram

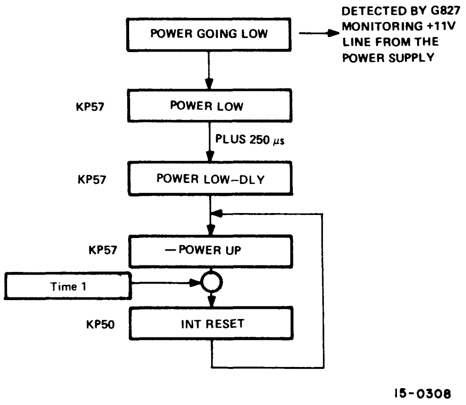


Figure 6-5 Power Fail Sequence With
Power Fail Option Disabled, Flow Diagram

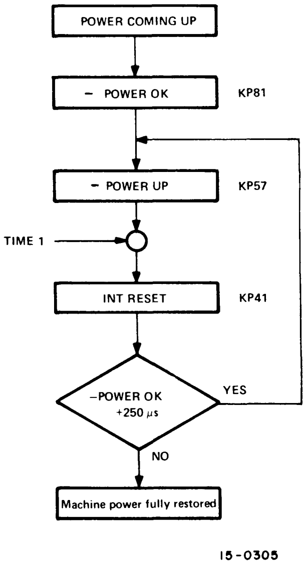


Figure 6-6 Power Restore Sequence With
Power Fail Option Disabled, Flow Diagram

In the case of PI, a program interrupt break is performed and the power fail flag detected by the power fail skip (SPFAL). A subroutine should then store the CPU registers, place a jump to a restart routine in location 0 and then halt.

When power is restored, the machine executes the instruction which was stored in location 0 which restores the registers and returns the machine to its previous program sequence.

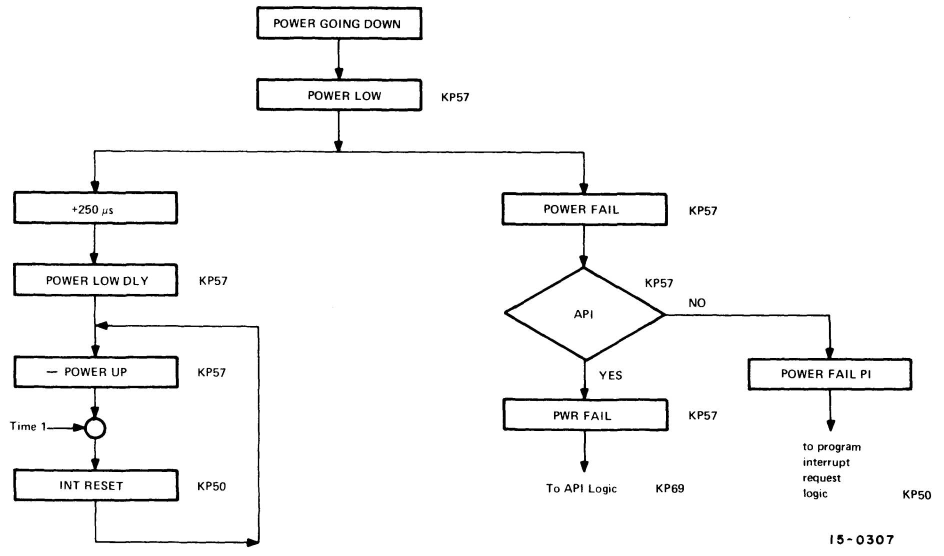


Figure 6-7 Power Fail Sequence With Power Fail Option Enabled, Flow Diagram

6.4 KA15 AUTOMATIC PRIORITY INTERRUPT (API)

The KA15 Automatic Priority Interrupt (API) option is located in the BB15 option panel. The option consists of standard M Series logic with no special modules. The API logic for the internal options, i.e., Power Fail and Real-Time Clock, is contained within the API option. All communications to external devices using the API facility are handled by the standard I/O bus cables. One additional control cable is necessary for communication between the PDP-15 I/O processor and the API option. All external devices using the API facility will be required to use the M104 module or its logical equivalent. See Chapter 4 for a description of the M104 Multiplexer module.

6.4.1 General Description

The API option increases the I/O handling capabilities of the PDP-15 by adding eight levels of priority servicing for up to 28 I/O devices along with four software channels for a total of 32 API channels.

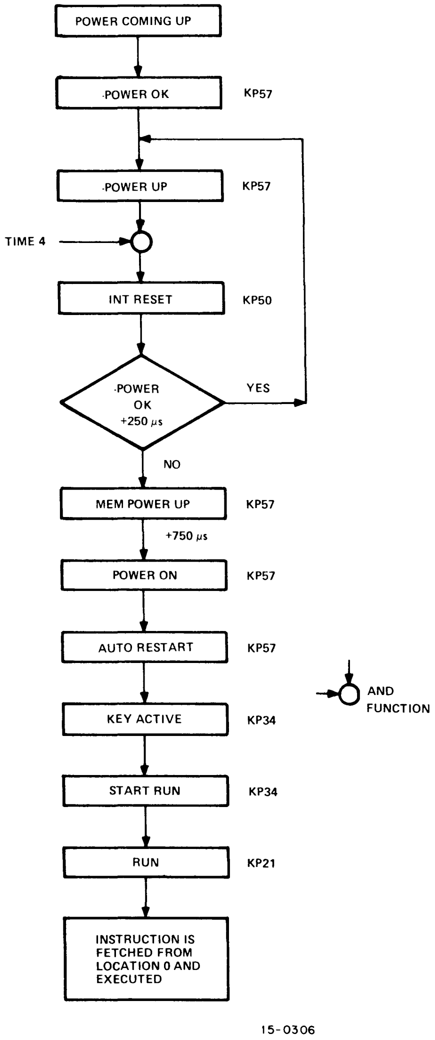


Figure 6-8 Power Restore Sequence With Power Fail Option Enabled, Flow Diagram

The four highest priority levels (0, 1, 2, and 3) have a higher priority than the program interrupt facility and main program operation, but a lower priority than the single-cycle DCH and real time clock operations. The four lower priority levels (4, 5, 6, and 7) have a higher priority than the main program operation, but a lower priority than program interrupts, the four higher API levels, single-cycle, DCH, and real-time clock operation.

Each of the 32 channels (0-37_8) has its own unique entry point to its specific service routine. These entry points are memory addresses $40_{(8)}$ -$77_{(8)}$ in page 0 of bank 0 in the main memory (see Table 6-22)

Table 6-22
Standard API Channel/Priority Assignments

Octal Channel Number	Device Name	Device Mnemonic	Priority Level Assigned	Unique Entry Point Octal Address
0	Software Priority	------	4	40
1	Software Priority	------	5	41
2	Software Priority	------	6	42
3	Software Priority	------	7	43
4	DECtape	TC02, TC15	1	44
5	Magtape	TC59	1	45
6	Not assigned		1	46
7	Not assigned		1	47
10	High Speed Tape Reader	PC15	2	50
11	Real Time Clock	KW15	3	51
12	Power Fail	KF15	0	52
13	Memory Parity	MP15	0	53
14	Display Control	VP15 or VT15	2	54
15	Card Reader	CR03B	2	55
16	Line Printer Control	LP15C or F	3	56
17	A/D Conv.	AD15	0	57
20	Inter Processor Buffer	DB09	3	60
21				61
22	637 Data Phone	DP09	2	62
23	DEC Disk Control	RF15	1	63
24	Disk Pack Control	RP15	1	64
25	Plotter	XY 15	2	65
26				66
27				67
30	Display	340		70
31				71
32				72
33				73
34	Teletype Keyboard	LT19/LT15A	3	74
35	Teletype Printer	LT19/LT15A	3	75
36	DECtape (DCH channel 36)	TC02/TC15*	1	76
37	Dataphone	DP09*	2	77

NOTE: The above table channel assignments should remain fixed for software compatibility, but the suggested priority levels may be changed at the discretion of the user.

*Channel allocated for system with more than one of the above options.

The highest four levels of priority, i.e., 0, 1, 2, and 3 are assigned to hardware devices. The lower four levels, i.e., 4, 5, 6, and 7 are assigned to the four software channels. Of the 28 hardware channels, no more than eight can be multiplexed on any one of the four priority levels. This is strictly a hardware limitation imposed by cable lengths and circuit delays, and attempts to circumvent this restriction will create needless problems.

Each device, when granted service via the API facility, sends its specific entry point address to the computer. This address, which will contain a JMP or JMS instruction to the device service routine, will then be executed by the computer. This type of interrupt service eliminates the need for time consuming flag search routines, and extensive core use for interrupt handling routines, by automatically determining which device requested service and providing immediate entry to the proper service routine.

Higher priority devices will be able to interrupt lower priority routines upon sending and having a request granted. The priority of devices multiplexed on the same priority level is determined by the relative position of the devices on the I/O bus. The first device on the bus having highest priority at that level, the second having second highest priority, etc.

The entire API facility can be enabled or disabled by a single IOT instruction. There is no way to enable or disable specific priority levels or devices. A device may, however, disconnect itself from the API facility by merely clearing its flag in much the same way as it disconnects from the program interrupt facility.

In addition to the above, there are two special features in the API facility. These are:

a. The CAL Instruction - Execution of a CAL instruction with the API facility enabled automatically sets priority level 4 thereby shutting out software requests of a lower priority until this level is released.

b. Program Interrupt - A program interrupt, from any I/O device connected to the computer, sets priority level 3. This occurs whether or not the API facility is enabled. This causes all devices on priority level 3, all software requests and program interrupts to be shut out until the level is released.

Special care must be taken in the programming of the API option to take account of these two features.

6.4.2 Operational Description

There are three sources of request for service to the API facility. They are hardware (I/O devices), software (program requests), and test requests (maintenance aid). The requests are handled in the same way by the API facility regardless of the source (see Figure 6-9).

The only difference in the requests is their level of priority; the hardware having the highest and the software lowest. The test requests are used by the diagnostic program to simulate hardware requests.

When an API request is received, the appropriate RQ flop will be set at I/O processor Time 4, if the API facility is enabled, and if there is no API operation in progress (API SYNC 0). Refer to KA04 and KA01.

The RQ flop is compared against its associated priority level (PL00 through PL07, KA02 and KA03) and, if it is enabled, a RQ SYNC level is generated. The RQ SYNC of all eight priority levels are ORed together and used to generate a CP API RQ signal (KA04 and KP35).

The CP API RQ level is sent to the CPU where it informs the processor that the API facility is requesting service. If it is a hardware request, i.e., level 0-3, the program interrupt facility of the CPU is disabled. The API interrupt is handled by the CP and I/O processors in the same way as a program interrupt (see Figure 6-10).

Once the CPU and the IPU are ready to handle the API interrupt, the API SYNC flop (KP51) in the IPU is set at IPU Time 1.

The API SYNC flop is used to set the enabled API GRANT flop (KA01) which in turn sets the EN A flop in the I/O device to allow the device to send its unique entry address to the IPU via the I/O bus.

API SYNC is ANDed with IPU Time 3 to clear the API GRANT flop and is also ANDed with IPU Time 4 to set the PL flop corresponding to the priority level of the request. This disables this priority level and all lower levels until a DBK or DBR instruction clears out the PL flop.

With API GRANT on a zero, the next I/O SYNC pulse (IPU Time 1) will clear the EN A flop in the device.

The following are IOTs used in the API option. Refer to the PDP-15 Systems User Handbook for IOT descriptions.

IOT	Mnemonic	Name
703304	DBK	Debreak
703344	DBR	Debreak and Restore
705501	SPI	Skip on Priorities Inactive
705504	ISA	Initiate Selected Activity
705512	RPL	Read API Status
707742	RES	Restore

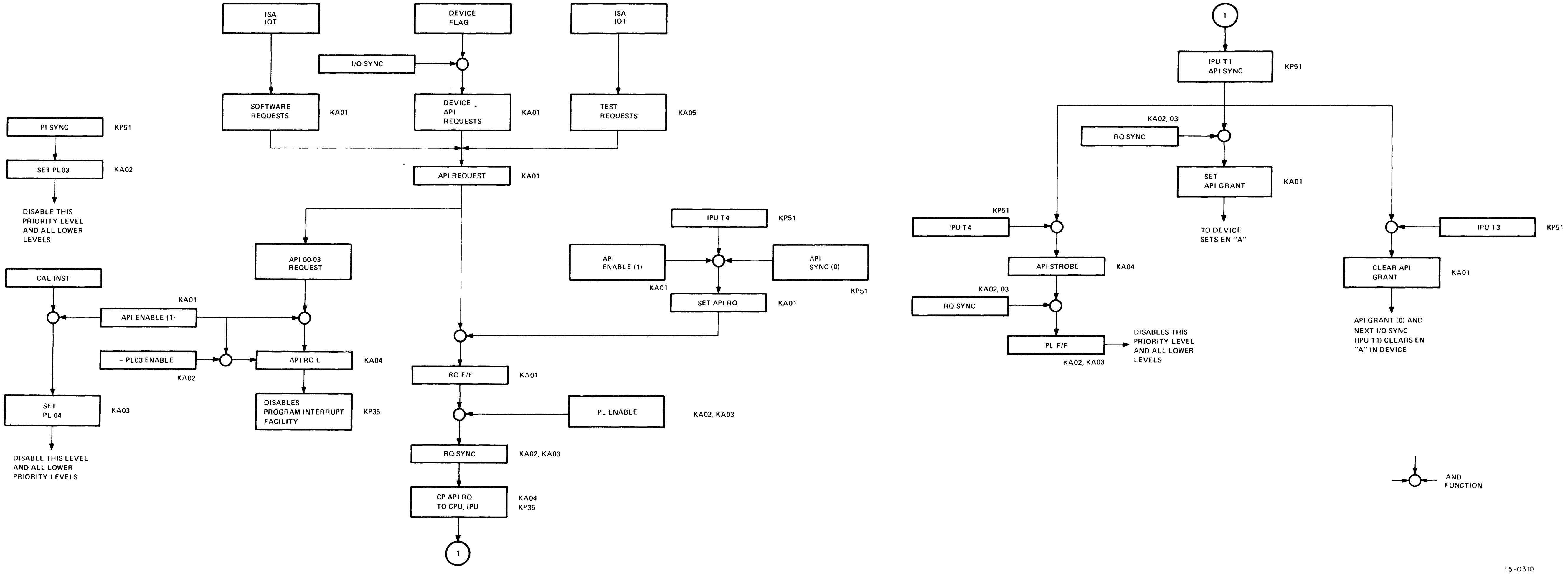


Figure 6-9 API Flow Diagram

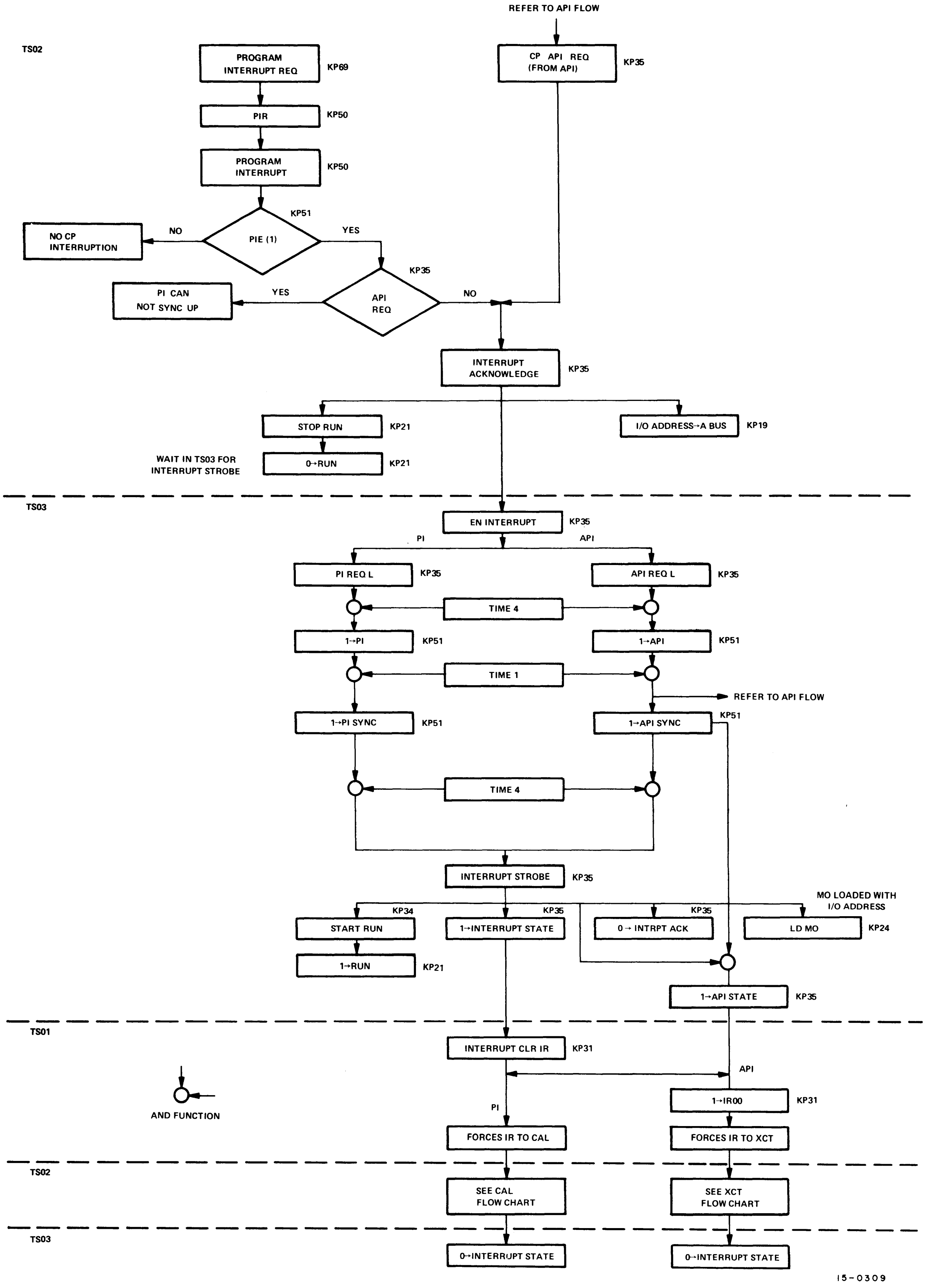


Figure 6-10 API/PI/Central Processor Flow Diagram

6.5 MP15 MEMORY PARITY

The MP15 Memory Parity option provides a continuous check of information being read from core memory, to determine whether bits are being picked up or dropped. It does this first by monitoring all information as it is being sent to the memory for storage. If there are an even number of bits in the data word, the MP15 control causes memory to write the parity bit, thus making the total number of bits odd. If there are an odd number of bits in the data word, the MP15 control inhibits the memory from writing the parity bit. The word is stored with an odd number of bits. When information is read from core, the parity control checks to see that an odd number of bits are read. If, per chance, it finds an even number, then a parity error has occurred. The parity error flag, which is set upon detection of an error, may be used to cause an API interrupt, a program interrupt, a skip request, or an immediate stop of the processor.

To perform the task of writing or not writing the parity bit into core, the MP15 needs time to calculate parity once it knows that data is present on the MDLs. It also needs time to generate the bit, once it has calculated that the bit is needed. Since the only way the MP15 knows that data is present is by MRLS, and since memory will not use the data until it receives MRLS, the MP15 interrupts the MRLS line of the memory bus and inserts a delay. Refer to MP10.

When the delay times out, MRLS is redriven onto the memory. The length of the delay time has been calculated to allow for worst case propogation times in the parity detection and parity bit circuitry.

Time is also required to perform the task of checking parity as information is read from memory. The only way the MP15 can know that data is coming from memory to the CP and is present on the MDLs, is by RD RST; the RD RST line of the memory bus is broken and the signal delayed. When the delay times out, RD RST is redriven to the CP. The delay time has been calculated to allow for worst case propogation time in the parity detection network. Refer to MP10.

The parity error flag, called PAR ERR, may be used in one of four ways. It is connected to the program interrupt facility, the automatic program interrupt facility, the skip facility and the CP stopping network. A switch is provided for the purpose of selecting whether or not the computer should stop upon detection of a parity error. The parity error channel address for the API is address 53 and the level is 0.

For maintenance purposes, the ability to force wrong parity to be written has been provided. An example follows:

```
.LOC    100
        FWP         /SET WRNG PAR FLOP
        DAC 200     /WRITE WRONG PARITY
        CPE         /MAKE SURE PAR ERR CLEAR
```

cleared until the ADR ACK following the write cycle of the interrupt. This is to allow UM (1) to be saved in 20 or 0.

If the violation is a boundary violation, it is detected before M REQ is allowed to go to memory; M REQ is inhibited from going to memory until the CP has synced up for a trap.

TRAP is set as soon as the violation is detected and will start the sync up process at TS03 and PHASE 1. If the violating instruction is a write instruction, TRAP produces a pulse in the processor called TRAP P which clears M REQ. It also produces an ADR ACK so that the processor time states may continue. END OF CP CYCLE is forced by TRAP, START WRITE, and M REQ (0). TRAP and START WRITE also disables CP MEM REQ HOLD. Therefore, another M REQ will not occur until the interrupt.

A non-existent memory violation is detected by timing out a delay started by M REQ. If an ADR ACK is not received within 500 ns, then NEXM is set. This forces PV and TRAP. TRAP and NEXM remove the M REQ from the bus. The functions in the processor are the same as those for a boundary violation on a write instruction as discussed in the preceding paragraph.

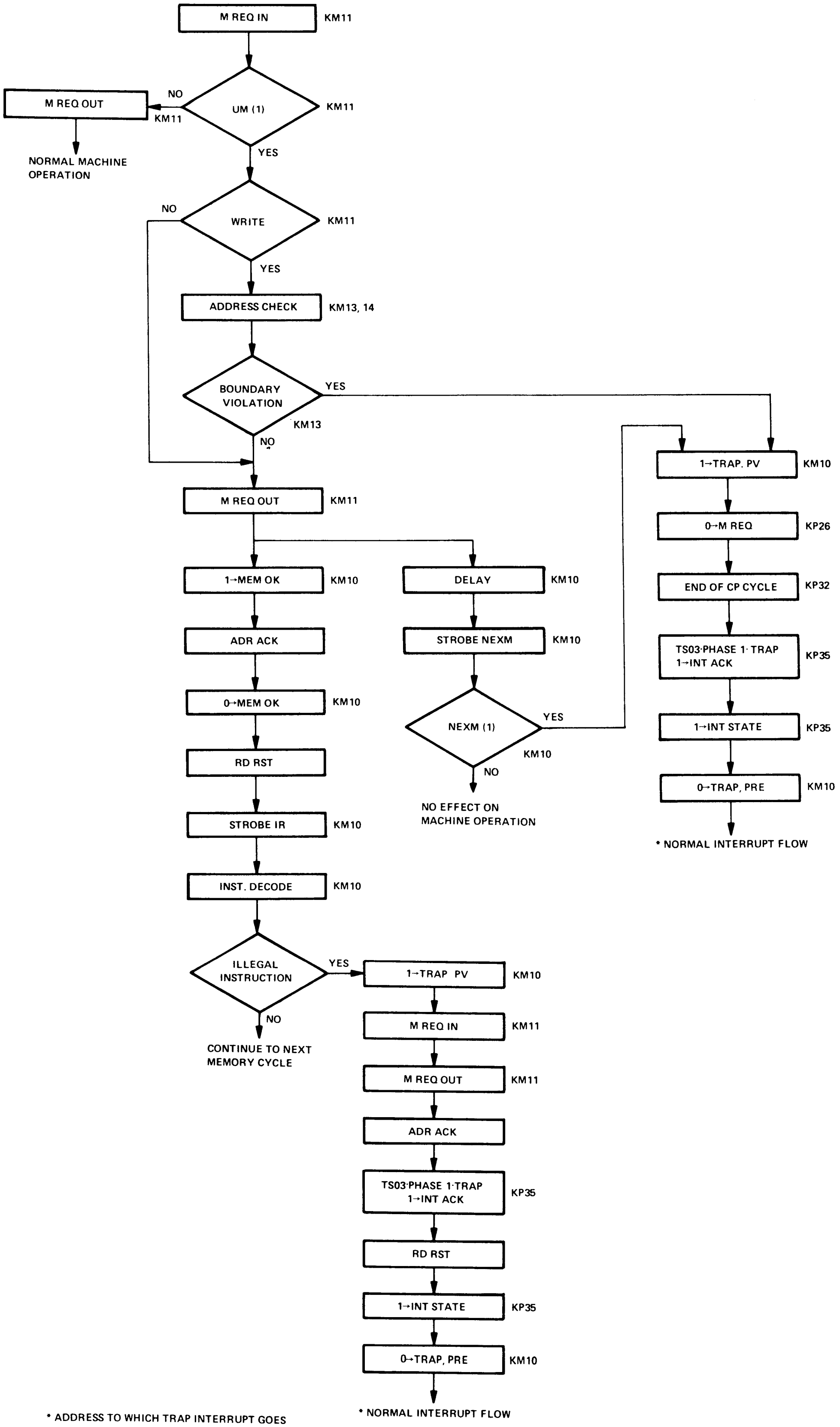
A CAL, PI, or API will cause user mode to be turned off and no violation will occur. The CAL and PI will save the state of user mode. On an API break, the instruction in the break address must be a JMS, JMS I, or CAL if the state of USER MODE is to be saved.

If USER MODE is on and the program tries to address non-existent memory, a TRAP occurs. The non-existent memory flag gets set; the computer restarts in a CAL just like any other violation. If USER MODE is OFF and the program tries to address non-existent memory, the computer hangs up and the non-existent memory flag gets set.

Although the KM15 introduces a delay to the computer, it is only 30 ns when not in USER MODE, when the I/O is active, or when in USER MODE and during any read function, except JMP or ISZ. The only time the cycle time is increased by 175 ns is with USER MODE and doing a write function, a JMP, or an ISZ.

The KM15 has the following instruction set:

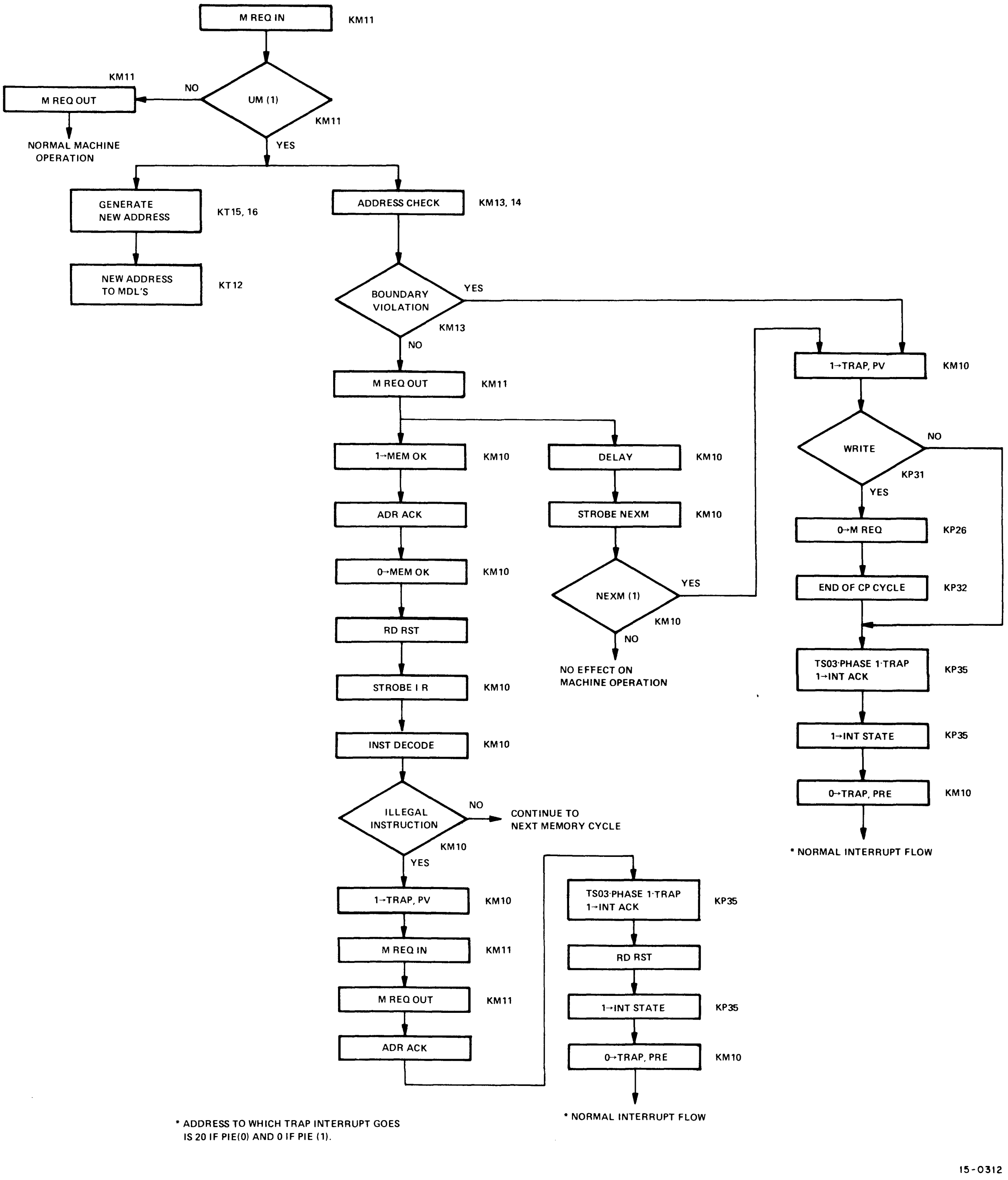
MPSK	701701	Skip on memory protect violation flag.
MPCV	701702	Clear memory protection flag.
MPLD	701704	Load boundary register.
MPSNE	701741	Skip on non-existent memory flag.
MPEU	701742	Enter User Mode.
MPCNE	701744	Clear non-existent memory flag.



* ADDRESS TO WHICH TRAP INTERRUPT GOES IS 20 IF PIE(0) AND 0 IF PIE (1)

Figure 6-11 KM15 Memory Protect Flow Diagram

15-0313



* ADDRESS TO WHICH TRAP INTERRUPT GOES IS 20 IF PIE(0) AND 0 IF PIE (1).

Figure 6-12 KT15 Memory Protect/Relocate Flow Diagram

If the violating instruction is a write instruction, TRAP produces a pulse in the CP called TRAP P which clears M REQ. It also produces an ADR ACK so that the CP time states may continue. END OF CP CYCLE is forced by TRAP, START WRITE, and M REQ (0). TRAP and START WRITE also disables CP MEM REQ HOLD to prevent another M REQ until the interrupt.

A non-existent memory violation is detected by timing out a delay started by M REQ. If an ADR ACK is not received within 500 ns then NEXM is set. This forces PV and TRAP set. TRAP and NEXM removes the M REQ from the bus. The functions in the processor are the same as those for a boundary violation on a write instruction as discussed in the preceding paragraph.

A CAL, PI, or API will cause User mode to be turned off and no violation will occur. The CAL and PI will save the state of User mode prior to clearing the UM flop. On an API break, if the state of User mode is to be saved the break address must contain a JMS, JMS I, or a CAL. If User mode is on and the program tries to address non-existent memory, a TRAP occurs. The NEXM flag gets set, and the computer restarts in a CAL just like any other violation. If User mode is off and the program tries to address non-existent memory, the computer hangs up and the NEXM flag gets set.

The KT15 has the following instruction set:

MPSK	701701	Skip on memory protect violation flag.
MPCV	701702	Clear memory protect violation flag.
MPLD	701704	Load core allocation register.
MPSNE	701741	Skip on non-existent memory flag.
MPEU	701742	Enter User Mode
MPCNE	701744	Clear non-existent memory flag.
MPLR	701724	Load Relocation Register.

6.8 PC15 HIGH-SPEED PAPER-TAPE READER/PUNCH

The PC15 High Speed Reader/Punch consists of control logic located in the BA15 peripheral option expander and the PC05 Reader/Punch which contains the Reader Punch mechanics and the tape movement circuitry.

6.8.1 High-Speed Reader

The PC05 Reader has the capability of advancing and reading tape one character at a time up to a maximum of 300 characters per second (refer to PC05 manual for complete details). The control logic in the BA15 interfaces the PC05 to the PDP-15 I/O Bus and also adds two features required by the PDP-15, i.e., character packing and hardware read-in.

The PC15 is able to read tapes in two modes, alpha and binary. A hardware read-in feature is also included.

Table 6-24
High-Speed Reader IOTs

Mnemonic	Octal Code	Operation Executed
RSF	700101	Skip if Reader Flag is set.
RCF	700102	Clear Reader flag, then inclusively OR the contents of the reader buffer into the AC.
RRB	700112	Clear Reader flag, and AC, then transfer contents of reader buffer into AC.
RSA	700104	Select Reader in alphanumeric mode.
RSB	700144	Select Reader in binary mode.

6.8.1.1 Alpha Mode - The reader, in this mode, reads one 8-bit character from tape and then signals the computer that the operation is completed.

The reader control sets the ALPHA flop, signals the reader to read a character, loads tape channels 7 and 8 into the 12-bit buffer in the control and then raises the reader flag (the remaining six bits are stored in the PC05 reader) (see Figure 6-13).

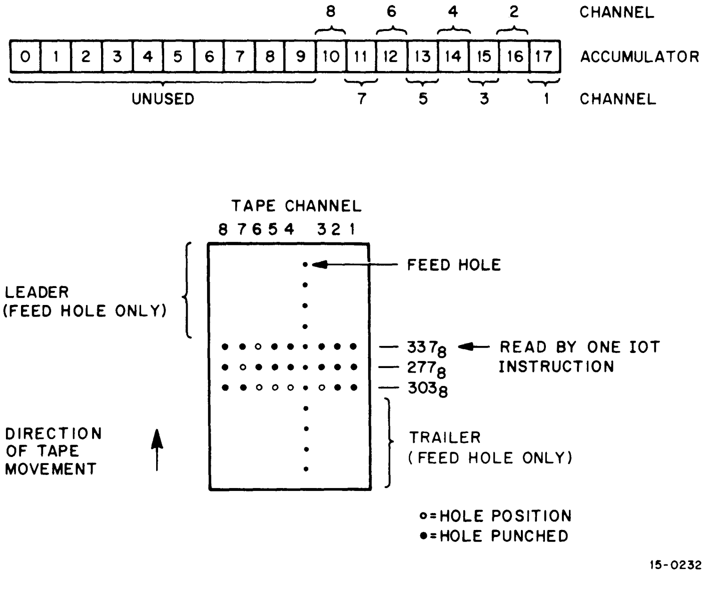


Figure 6-13 Tape Format and Accumulator Bits (Alphanumeric Mode)

6.8.1.2 Binary Mode - The reader advances and reads the tape and loads an 18-bit buffer with three 6-bit characters. Characters without an eight hole punch are ignored. When the buffer is full, an 18-bit word can be transferred to the accumulator of the central processor.

The first two 6-bit characters are loaded into the 12-bit register in the reader control, the remaining are stored in the PC05 until read by a RRB IOT (see Figure 6-14).

6.8.1.3 Read-In - When the console READ IN key is pressed and the PC15 is installed, the PC0 READ IN flop is set (KP66). Refer to the read-in flow diagram (KP75). In read-in mode, 18-bit characters are assembled, as in binary mode. When the central processor has signaled by the character interrupt line that a character is ready, the character is transferred by IOP2 and then another character is read. The storage of the character in memory is described in Paragraph 3.11. When a hole seven punch is detected, the SKIP RQ line is enabled to signal the processor that this is the last word to be transferred and that it should be executed. A I/O PWR CLR pulse then clears out the READ IN mode.

6.8.2 High-Speed Punch

The punch receives data from the I/O bus and punches it on paper tape. Almost all of the punch logic is located in the PC05. The punch is capable of punching in two modes, binary or alpha. In binary mode, a 6-bit character is punched onto tape, bit 7 is never punched and bit 8 is always punched. In alpha mode, 8-bit characters are punched.

Table 6-25
High-Speed Paper Tape Punch

Mnemonic	Octal Code	Operation Executed
PSF	700201	Skip if punch flag is set.
PCF	700202	Clear punch flag.
PSA	700204	Punch a line of tape in alphanumeric mode.
PSB	700244	Punch a line of tape in binary mode.

6.9 LT15A TELETYPE INTERFACE

The interface for the first Teletype in a PDP-15 system is included with the CPU logic (see KP64 and KP65). If a second Teletype is used, it requires an LT15A Teletype interface, which mounts in the BA15 panel. Because the internal and LT15A interfaces are nearly identical, this paragraph describes both.

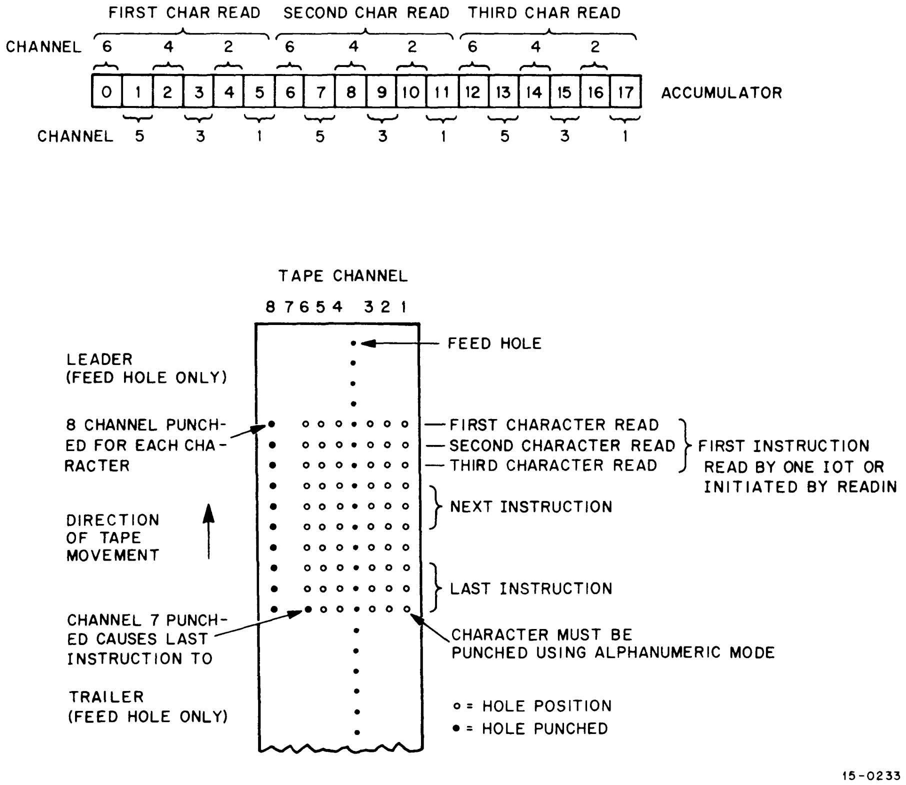


Figure 6-14 HRI Tape Format and Accumulator Bits (Binary Mode)

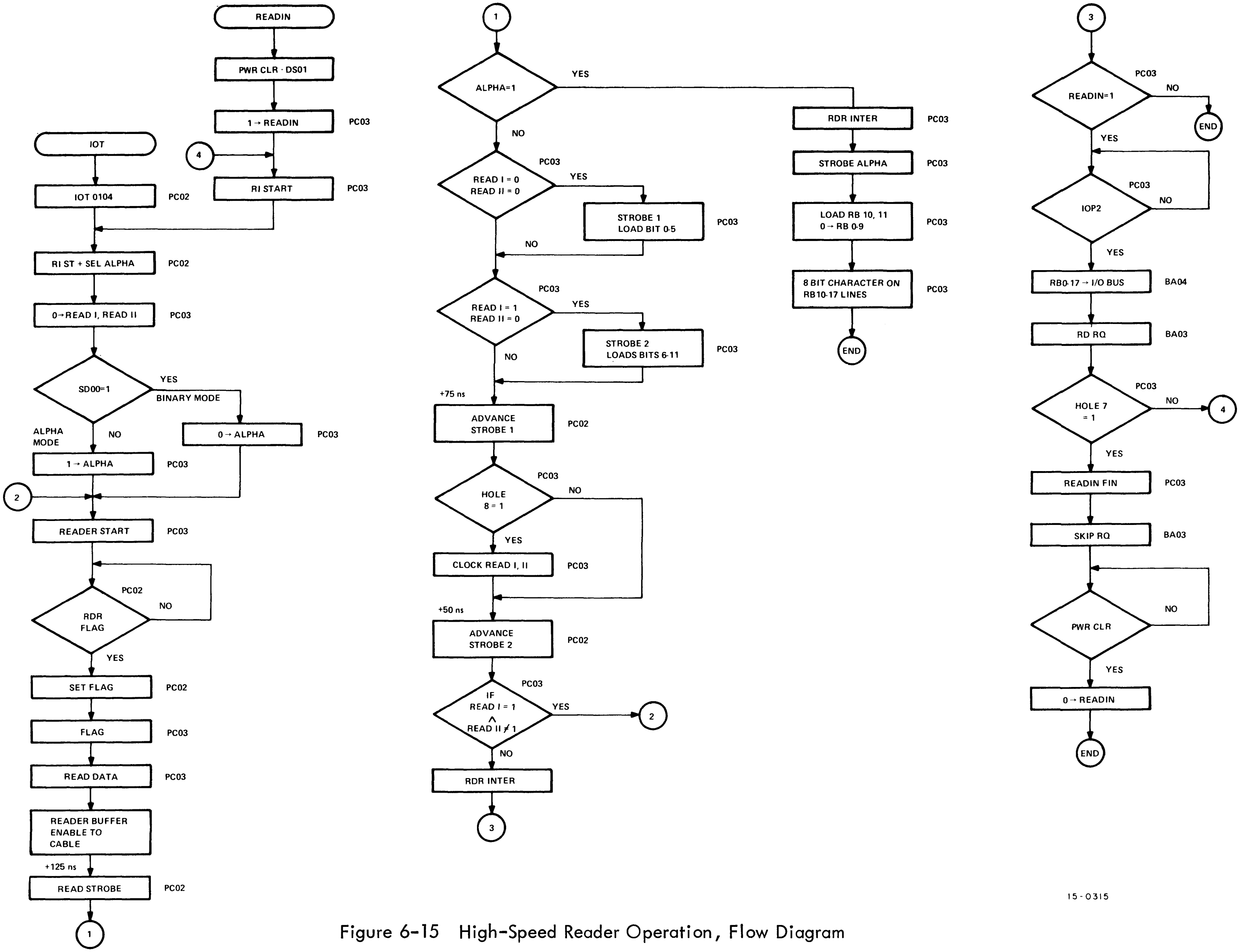


Figure 6-15 High-Speed Reader Operation, Flow Diagram

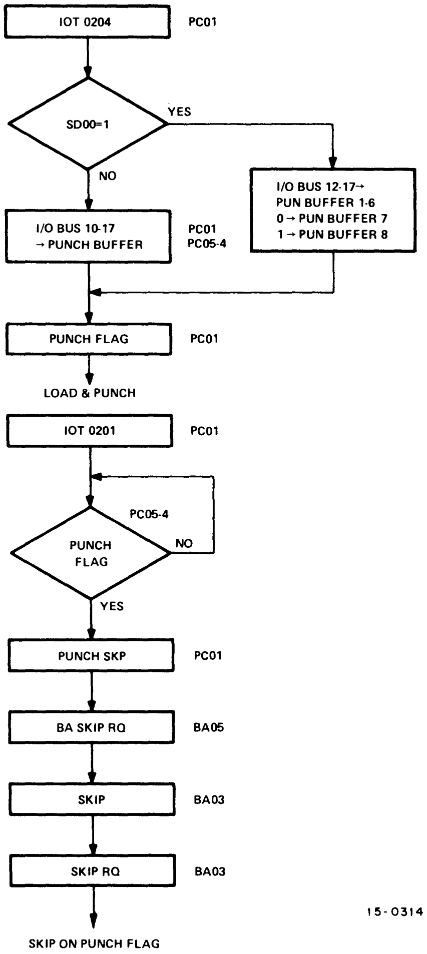
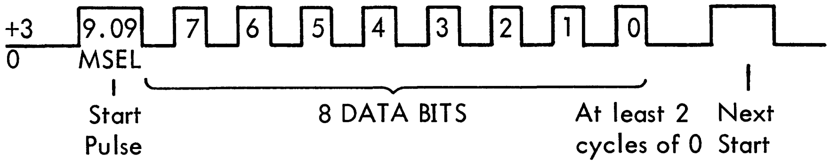


Figure 6-16 High-Speed Punch Operation, Flow Diagram

The Teletypes send and receive 8-bit ASCII codes in the following serial format:



The keyboard and the printer are completely separate systems, and use different serial lines for their data. The corresponding sections of the interface are also separate, but share the 880-Hz clock (LT02).

The receiver portion of the interface (KP64,LT02) uses a 9-bit shift register to store the start pulse and serial data bits as they come in. When the input line is raised by the start pulse, the SPIKE DETECTOR flop is cleared. If the signal is still present a half cycle (4.5 ms) later, IN ACTIVE is cleared and shifting of input data begins. The CLOCK SCALE network causes a shift every 9.09 ms until the start pulse reaches the IN LAST UNIT flop. At this point the keyboard flag is raised, indicating that data is ready to be read into the PDP-15 with an IOT. The IN STOP flops prevent further operation for the full two zero cycles.

The transmitter (KP65,LT03) uses a 9-bit shift register to convert parallel data from the I/O bus to serial form. The print IOT loads TTO00-TTO07 from bus bits 10-17. TTO ENABLE and OUT ACTIVE are set. LINE is also set, causing the start pulse at the output. The bits are shifted, one-by-one into the LINE flop until no 1s remain to the left of TTO07. TTO FIN detects this, stops the shifting by dropping OUT ACTIVE, and sets the TELEPRINT FLAG. The OUT STOP flops produce the two cycles of 0 by preventing further enables until they clear.

In the internal interface, automatic printing of incoming characters is provided by connecting the input line to the output line, and holding off IOT enables while this is in progress.

When executing the KRS instruction, and during read-in operations, the READER RUN flop is set. This inhibits the echo and activates the Teletype paper-tape reader. Only the internal interface uses this feature.

Table 6-26
Primary Teletype IOTs

Mnemonic	Octal Code	Operation Executed
KSR	700301	Skip if keyboard flag set.
KRB	700312	Read keyboard buffer into AC10-17, clear flag.
KRS	700332	Read keyboard buffer, clear flag, select keyboard reader for next character.

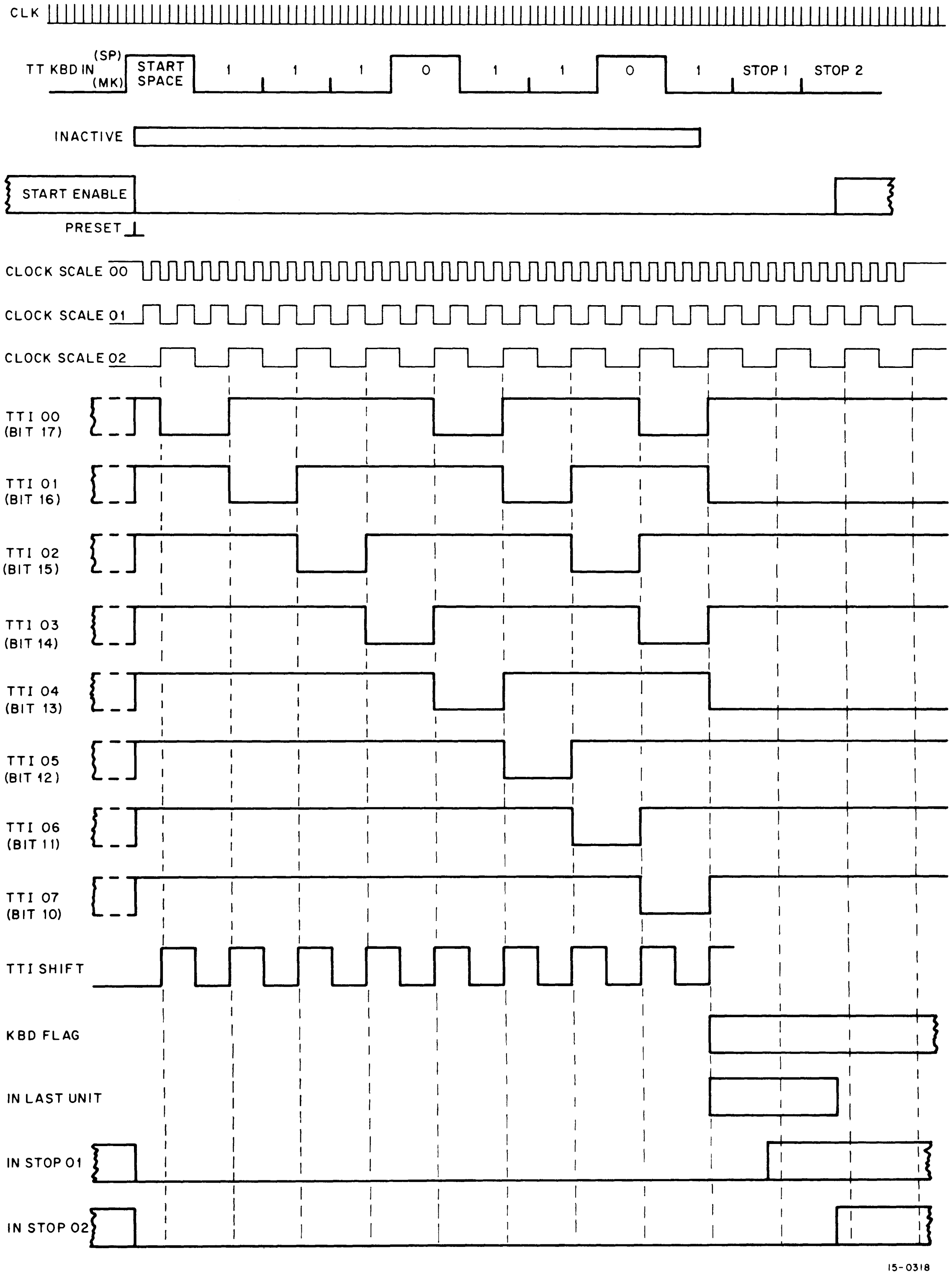


Figure 6-17 Teletype Receiver (Keyboard) Timing

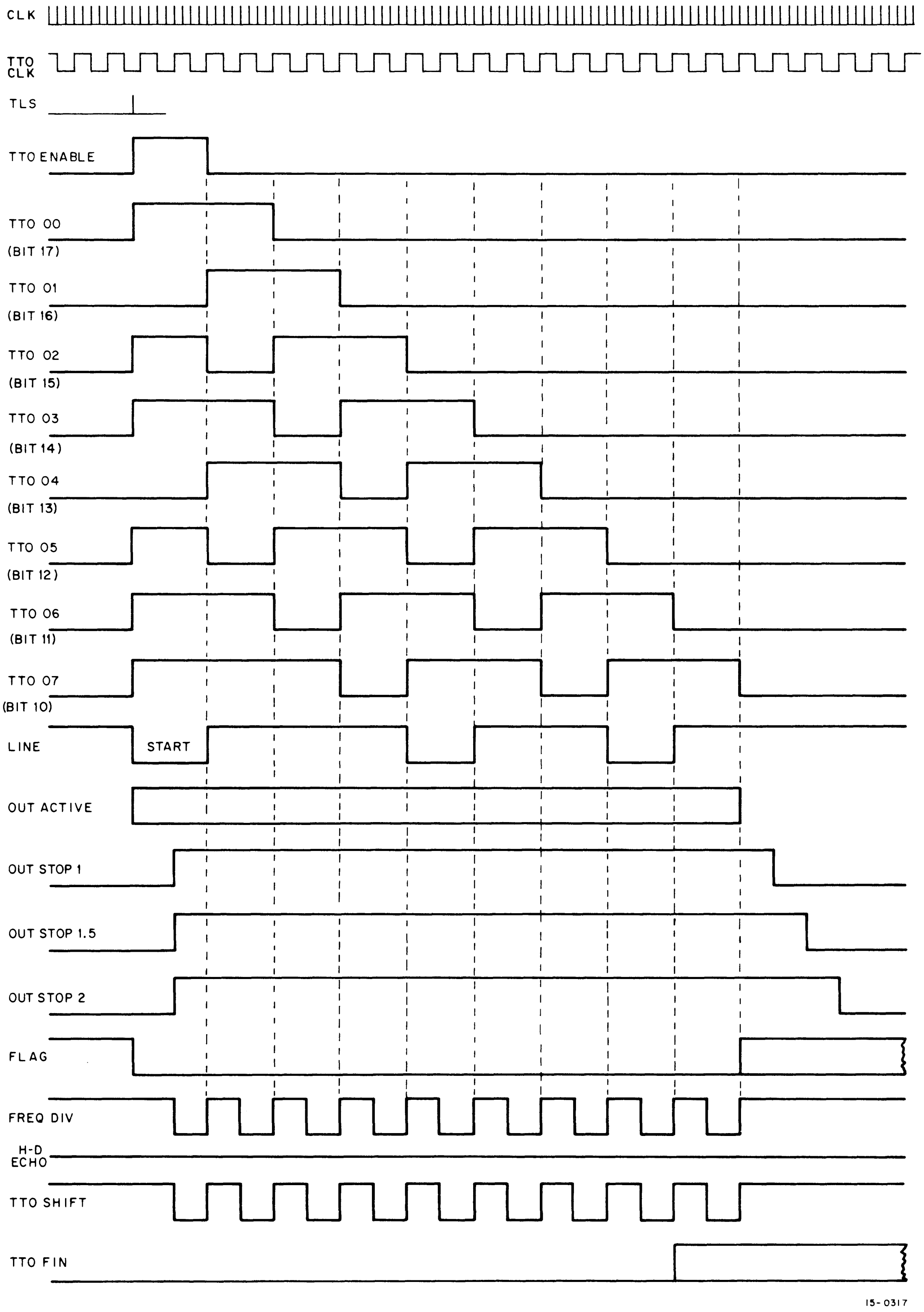


Figure 6-18 Teletype Transmitter (Printer) Timing

Table 6-26 (Cont)
Primary Teletype IOTs

Mnemonic	Octal Code	Operation Executed
TSF	700401	Skip if teleprinter flag set.
TCF	700402	Clear teleprinter flag.
TLS	700406	Load teleprinter buffer from AC10-17, start print operation.

There is no API connection and the internal Teletype uses the PI only.

Table 6-27
The LT15A Instruction Set

Mnemonic	Octal Code	Operation Executed
TSF1	704001	Skip on transmitter (teleprinter) flag.
TCF1	704002	Clear transmitter flag.
TLS1	704004	Load transmitter buffer and transmit.
KSF1	704101	Skip on receiver flag.
KRB1	704102	Clear receiver flag and read buffer.

The LT15A is connected to the API system, when installed.

API for keyboard: Priority level 3, Address 74.
API for teleprinter: Priority level 3, Address 75.

I/O bus receivers and drivers for the LT15A are provided in the BA15 option panel.

6.10 VP15 DISPLAY CONTROL

The VP15 is a point plotting display control which interfaces any one of three displays to a PDP-15: The VP15A uses a Tektronix® 611 storage tube (VT01), the VP15B uses a Tektronix RM503 oscilloscope, and the VP15C uses a DECtype VR12 X-Y Display.

Points are displayed in a 1024 x 1024 bit matrix on an 8-1/4" x 6-3/8" display surface for the 611 tube, an 8 cm x 10 cm display surface for the RM503 scope, and a 6-3/4" x 9" display surface for the VR12 display.

® Tektronix is a registered trademark of Tektronix, Inc., Beaverton, Oregon.

A light pen option is available for both the RM503 and VR12 display systems (VP15 BL and VP15 CL).

The VP15A operates in two modes: Store Mode, where the point plotted is stored on the screen, and Non-Store Mode, where points displayed must be "refreshed" or replotted at least 30 times a second in order to remain visible. A display Done Flag is raised at the completion of the display command. The VT01 storage oscilloscope may be erased by IOT command and requires about 0.5 seconds for completion. The display Done Flag is raised at the end of this operation.

The Display Flag interrupts the computer and a skip on Display Flag is provided. The non-store mode is designed for the display of pointers and small groups of characters. Its intensity may be significantly less than in store mode. It does not use the write-through feature on the VT01.

The VP15B and VP15C are refresh displays and must have the information replotted at least 30 times a second. If a light pen is on the system, a light pen flag is raised whenever light is detected. A timer is provided to limit the occurrence of the light pen flag to about 1 kHz.

The VP15 Display Control includes the display control logic and the digital-to-analog converters.

6.10.1 Display Control

The Display Control provides the timing for point intensification and programmable erasing (VT01) of the display, and a 2-bit brightness register used on the RM503 and VR12 displays.

IOTs initiate the timing sequence for all displays. The VP15A has two IOTs for this purpose, one for store mode and another for non-store. The VP15B and C have only one IOT for intensification.

In the VP15A, a display Done Flag is raised when the display has finished plotting the point previously requested. This flag may be cleared and skipped by an IOT control.

The display may be erased manually by depressing a button on the scope or by an IOT Command (VT01 storage oscilloscope only). Erase takes 0.5 seconds.

The 2-bit brightness register (BR00 and BR01) is loaded from I/O bus bits 16-17 by IOT control. There are four brightness levels for the RM503 and VR12.

6.10.2 Digital-to-Analog Converters (D/A)

The VP15 display contains two digital-to-analog converters (D/A); one for the x-coordinate buffer, and one for the y-coordinate buffer. Coordinate data is loaded into these from the accumulator of the PDP-15 by IOT control.

Upon IOT command, the D/A converters for each coordinate are cleared and loaded with ten bits of information from accumulator bits 08-17. The x buffer (XB) and y buffer (YB) are loaded separately. These 10 bits are converted into a voltage which is used as the input to either the x or y deflection circuitry in the oscilloscope.

6.10.3 VP15A Storage Tube Display IOTs

6.10.3.1 Non-Store Mode

LXDNS	700544	Load the x-coordinate buffer and display (non-stored); the point specified by XB and YB.
LYDNS	700644	Load the y-coordinate buffer and display (non-stored); the point specified by XB and YB.

6.10.3.2 Store Mode

LXBD	700564	Load the x-coordinate buffer and display (stored); the point specified by XB and YB.
LYBD	700664	Load the y-coordinate buffer and display (stored); the point specified by XB and YB.
EST	700724	Erase the storage oscilloscope.

6.10.3.3 Store and Non-Store Mode

CXB	700522	Clear x-coordinate buffer.
CYB	700622	Clear y-coordinate buffer.
LXB	700524	Load x-coordinate buffer from AC08-17.
LYB	700624	Load y-coordinate buffer from AC08-17.
EST	700724	Erase the storage tube.
SDDF	700521	Skip if display Done Flag is set.
CDDF	700722	Clear display Done Flag.

6.10.4 VP15C Display and VP15B Oscilloscope IOTs

DXL	700504	Load x-coordinate buffer from AC08-17.
DXS	700544	Load x-coordinate buffer and display point specified by XB and YB.
DYL	700604	Load y-coordinate buffer from AC08-17.

DYS	700644	Load y-coordinate buffer and display point specified by XB and YB.
DXC	700502	Clear x-coordinate buffer.
DYC	700602	Clear y-coordinate buffer.
DLB	700704	Load brightness register from bits 16-17 of AC.
DSF	700501	Skip if Display (light pen) Flag is set.
DCF	700702	Clear Display (light pen) Flag.

6.10.5 Principles of Operation

A simplified block diagram of the VP15 controller is given in Figure 6-19. This controller consists of three sub-systems:

a. A 10 bit buffer and D/A converted for each x and y deflection circuit.

b. Z axis timing and control circuits.

c. Device decoding, I/O bus receivers and drivers, and interrupt logic (OR API Logic).

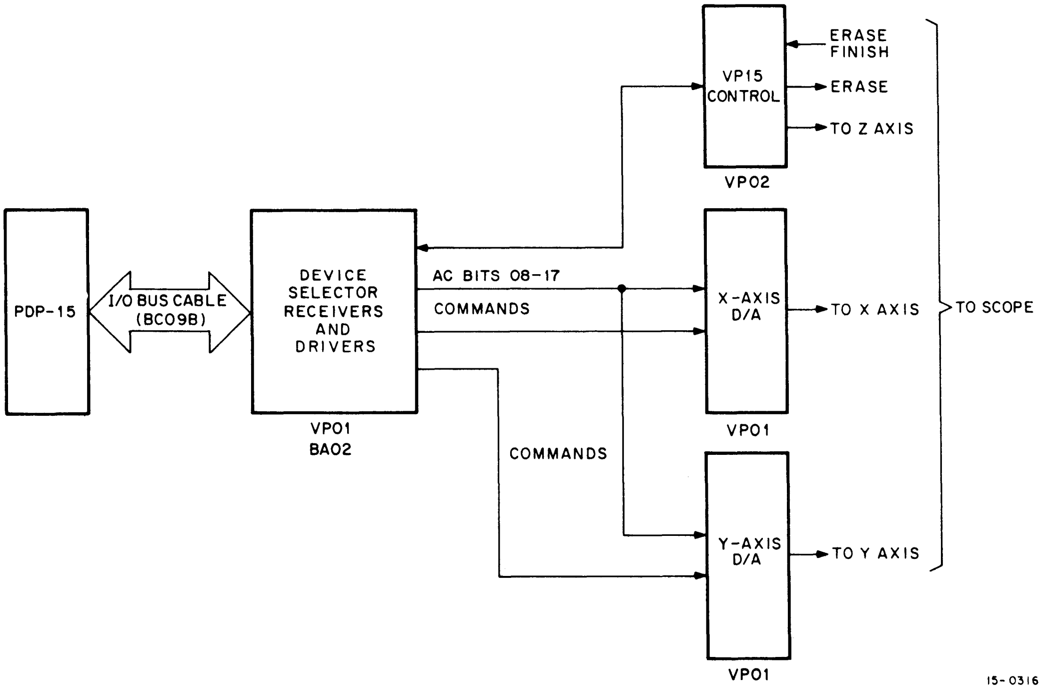


Figure 6-19 VP15 Controller, Simplified Block Diagram

The receivers and drivers interface the device to the I/O bus and to the PDP-15. The device selectors decode the various IOT commands. The 10 bit x and y coordinate buffers, which define a 1024 x 1024 matrix, are cleared and loaded, by IOT commands. After an intensify command has been completed in the VP15A, a display done flag will be set, and the computer interrupt will occur. This condition may be tested by a skip or display done flag.

The VT01 scope may be erased by IOT command and the display flag will be set as above upon the completion of the erase.

An input/output read status, (IORS), instruction will read the status of the display flag into AC bit 05, and light pen flag into AC bit 05; for systems with API, the display is a priority level 2, and its I/O address is 54.

Table 6-28
Settling and Intensification Times

		Deflection Settling Time (μs)	Intensification Time (μs)
VP15A	Store Mode	80	20
	Non-Store Mode	80	1.5-2.2
VP15B		15	1.5-2.2
VP15C		3	1.5-2.2

CHAPTER 7
MAINTENANCE

7.1 INTRODUCTION

It is not the purpose of this chapter to train the reader in the theory of operation of the PDP-15 computer. It is assumed that the reader is already familiar with the PDP-15's operation, and this chapter will act as an aid in the maintenance and troubleshooting of the PDP-15.

It is recommended that maintenance and repairs be performed by qualified service personnel only. Potentially dangerous voltages are present in the power supplies. Safety precautions must be observed.

In this chapter the preventative and corrective maintenance procedures for the PDP-15 computer will be discussed, and a general description of logic troubleshooting is given. The adjustments of PDP-15 Memory, Central Processor, and I/O Processor are shown, and are accompanied by appropriate pictures of the wave forms which can be expected to be seen when the adjustments are properly made.

7.2 SYSTEM MAINTENANCE

System reliability and maintainability are two of the most important criteria in the design of the PDP-15 computer. This fact, however, does not preclude the necessity of setting up a systematic preventative maintenance program. Proper application of such a program will aid both the maintenance personnel and the user through detection and prevention of probable failures that will help keep maintenance and downtime to a minimum.

7.2.1 Maintenance Equipment

Special tools and test equipment required for maintenance are listed in Table 7-1. Except for DEC equipment, suggested commercial brands are given for purposes of specification only; their being mentioned does not constitute exclusive endorsement.

Table 7-1
Maintenance Equipment

Equipment	Specification	Model or Type
Multimeter	10 Kohms/V - 20 Kohms/V	Triplett model 310 or 630-NA
Oscilloscope		Tektronix type 454, 547
Probes	X10 with response characteristics matched to oscilloscope	Tektronix type P6010, P6047
Clip-on current probe	2 mA/mV or 10 mA/mV	Tektronix type P6022 with passive terminator
Recessed probe tip (2 each)		Tektronix
Unwrapping tool	30 gauge	Gardner Denver 505-244-475
Wire-wrap tool	30 gauge	Gardner Denver A-20557-29
30-gauge bit for wire-wrap tool		Gardner Denver 504221
Sleeve for 30-gauge bit		Gardner Denver 500350
Module extender (2)		DEC no. W982
Jumper wires		Assorted lengths affixed with 30-gauge Termi-Points
Screw driver	6-in. non-conductive shaft	
Field service kit		DEC type 142
Diagnostic programs	(Supplied with system)	

7.2.2 Maintenance Test Programs

Table 7-2 lists the diagnostic programs designated MAINDEC. Altogether these programs provide a complete check of the system logic.

These programs are written in such a manner as to check certain circuits or functions of the machine, from a go/no go situation to isolate basic logic faults, through the use of random number generation to help isolate the more difficult random failures that may occur, and finally to programs designed to test the operability of the system under the interaction of various tests coupled with peripheral requests and interrupts.

When an error or failure is detected by the diagnostic it will signify such by either halting or through an error typeout. The reason for the halt or typeout may then be determined through investigation of the console indicators and controls or by use of a scope loop in the diagnostic if one is provided. The diagnostic write ups and listings should be consulted for the type of error failure and for the use of console controls and/or switch settings that will aid in the isolating of the machine fault.

Table 7-2
MAINDEC Diagnostic Programs

Identification Number	Program Name
PDP-15/20 (basic central processor, memory and Teletype)	
MAINDEC-15-D0B0-D	Instruction Test - Document
1-PH	Part 1
2-PH	Part 1A
MAINDEC-15-D0BB	Instruction Test, Part 2
MAINDEC-15-D0AB	Hardware Index Register Test
MAINDEC-15-D0DA	JMP - Self Test
MAINDEC-15-D0EA	JMP-Y Interrupt Test
MAINDEC-15-D0FA	JMS-Y Interrupt Test
MAINDEC-15-D0KA	ISZ Test
MAINDEC-15-D1A0-D	Basic Memory Checkerboard Document
1-PB	Low
2-PB	High
MAINDEC-15-D1BB	Extended Memory Checkerboard
MAINDEC-15-D0CA	Memory Address Test
MAINDEC-15-D1CD	Extended Memory Test
MAINDEC-15-D1FA	Extended Memory Address Test
MAINDEC-15-D1GA	Memory Address Timing Test
MAINDEC-15-D7AA	4K Basic Exerciser
MAINDEC-15-D7BC	8K Basic Exerciser (if 8K of core)
MAINDEC-15-DZAA	ASR33/35 Teletype Test (Part 1)
MAINDEC-15-DZBA	ASR33/35 Teletype Test (Part 2)
MAINDEC-15-D2G3	Binary Count Pattern Test Tape (for above)
PDP-15/20 (same as 8K 15/10, plus the following)	
MAINDEC-15-D0GA	EAE Test, Part 1
MAINDEC-15-D0HA	EAE Test, Part 2
MAINDEC-15-DZDA	High Speed Punch Test
MAINDEC-15-DZCA	High Speed Reader Test
MAINDEC-15-DZC0	Reader Test Tape (for above)
MAINDEC-15-D3BA	DECtape Basic Exerciser, Part 1
MAINDEC-15-D3CA	DECtape Basic Exerciser, Part 2
MAINDEC-15-D3RA	DECtape Random Exerciser
PDP-15/30 (same as 15/20, plus the following)	
MAINDEC-15-D0IB	I/O Test (API)
MAINDEC-15-D1EA	Memory Protect Test
MAINDEC-15-D8CA	LT09/LT19 Teletype Control Test

Table 7-2 (Cont)
MAINDEC Diagnostic Programs

Identification Number	Program Name
PDP-15/40 (same as 15/30, plus the following)	
MAINDEC-15-D5AA	RF15 Disk Data Test
MAINDEC-15-D5BB	RF15 Multi-Disk Test
MAINDEC-15-D5CA	DISKLESS
Peripherals and Options for PDP-15/20/30/40	
1) With CR03B Card Reader	
MAINDEC-15-DZUA	CR03B GDI Card Reader Test
MAINDEC-15-DZAZ-C	Binary Cards (for above)
2) With VP15A Display	
MAINDEC-15-D6BA	Display Diagnostic
3) With TU20 or TU20A Magnetic Tape	
MAINDEC-15-D4CB	Magnetic Tape Control Drive Function Timer
MAINDEC-15-D4DA	7 Track Data Reliability
MAINDEC-15-D4EA	9 Track Data Reliability
MAINDEC-15-D4GB	Random Exerciser
4) KT15 Memory Relocate	
KT15 D1JA	Memory Relocate Test
5) KT15 D1K0-D	Memory Parity Test Document
1-PB	Low
2-PB	High
6) KF Power Fail	
KF D0JA	Power Fail Test

7.3 PREVENTIVE MAINTENANCE

7.3.1 Introduction

This section provides information for performing preventive maintenance inspections. This information consists of visual, static, and dynamic tests that provide better equipment reliability. Preventive maintenance consists of procedures that are performed prior to the initial operation of the computer and periodically during its operating life. These procedures include visual inspections, cleaning, mechanical checks, and operational testing. A log should be kept for recording specific data which indicates the performance history and rate of deterioration. This information can then be used to determine the need and time for performing corrective maintenance on the system.

Scheduling of computer usage should always include time for scheduled preventive maintenance checks. Careful testing during this scheduled time may turn up faults that occur intermittently during normal operation or catch problems before they occur resulting in an overall saving of computer usage time for a line operation.

7.3.2 Scheduled Maintenance

The PDP-15 must receive certain routine maintenance attention to ensure maximum life and reliability of the computer system. DEC recommends the following schedule:

1000 hrs:	Electrical Inspection
500 hrs:	Mechanical Inspection or at least once every 3 months

Daily Maintenance

Once a day the Basic Exerciser Program MAINDEC should be run for several minutes to ensure general overall system operation.

Weekly Maintenance

Time should be scheduled each week to run the MAINDEC Programs listed in Table 7-2. Each program should be run for a minimum of five minutes. Take any corrective action needed at this time and log the results. The external cleanliness of the system should also be maintained on a weekly basis.

Computer downtime can be minimized by a rigid adherence to a preventive maintenance schedule. A dirty, clogged air filter can lead to machine failure due to overheating. All filters should be cleaned periodically. The procedure for cleaning filters is described under Preventive Maintenance Tasks.

Preventive Maintenance Tasks

The following tasks should be performed at least once every three months.

a. Clean both the exterior and interior of the computer cabinet, using a vacuum cleaner and/or clean cloth moistened in a non-flammable solvent.

b. Clean all air filters. Use a vacuum cleaner to remove the dirt and dust or wash the filters in clean warm water.

c. Lubricate all slide mechanisms, door pins and castors with a light machine oil. Wipe off any excess oil.

d. Inspect all wiring, cables and harnesses for cuts, breaks, fraying, wear, deterioration, kinks, strains and mechanical security. Replace or repair any defects found.

e. Inspect the following for both proper operation and/or mechanical security. Repair, replace, or tighten as required all lamps, switches, knobs, connectors, fuses, fans and covers.

f. Inspect all module mounting panels to ensure that all modules are firmly seated in their sockets. Remove and clean any modules that may have collected excess dirt or dust.

g. Inspect the power supply for leaky capacitors, overheated resistors, relay operation, etc.; replace or repair any defective items found.

h. Check the outputs of the 715 Power Supply without disconnecting the load. The outputs of the 715 Power Supply are non-adjustable; therefore, if any voltage is not within specification then corrective maintenance must be performed to remedy the defect.

i. Check the voltage outputs of all voltage regulators and make any necessary adjustments of the regulators to bring the voltages into specification. If the voltage cannot be adjusted to meet the specification the corrective maintenance must be performed.

j. Run all MAINDEC Programs to verify proper machine operation. Each program should be run for a minimum of 5 minutes.

k. Perform all preventive maintenance procedures for each peripheral device connected to the PDP-15 system as directed by the individual instructions supplied with each option.

l. Check for circuit deterioration by varying the machine timing in accordance with the PDP-15 engineering specification.

m. Enter the results of the preventive maintenance in the log book.

7.4 CORRECTIVE MAINTENANCE

7.4.1 Introduction

The PDP-15 is constructed of reliable TTL M-series modules. Proven reliability of this circuitry ensures relatively little equipment downtime due to logic failure. If a malfunction occurs, maintenance personne! should analyze the condition and correct it as indicated in the following procedures. The best corrective maintenance tool is a thorough understanding of the physical and electrical characteristics of the equipment. Personnel responsible for maintenance should be familiar with the system concept, the logic drawings, the theory of operation of the specific module circuits, and location of mechanical and electrical components.

The first step in repairing a reported malfunction is to isolate the problem. In a hardware-software system environment such as the PDP-15, the first step is to determine whether the problem lies in the hardware, the software, or both. The only practical way of doing this is by maintaining good communications between the operator, programmer, and maintenance personnel.

Until the problem is isolated to either hardware or software, the cooperation of all parties concerned is essential. A step-by-step procedure should be used to trace the problem until a point is reached where all the inputs (conditions) to an element (of the hardware) are correct, but the output is not correct. The faulty element thus located should be repaired. Where necessary, the element itself may be subjected to step-by-step fault location (from output to input) until the source of the problem is found.

It is virtually impossible to outline all specific procedures for locating faults within digital systems such as the PDP-15. However, diagnosis and remedial action for a faulty condition can be undertaken logically and systematically in the following phases:

a. Preliminary investigation
b. System troubleshooting
c. Logic troubleshooting
d. Circuit troubleshooting
e. Repairs and replacement
f. Validation tests
g. Recording

7.4.2 Preliminary Investigation

Before beginning troubleshooting procedures, explore every possible source of information. Gather all available information from those users who have encountered the same problem and check the system log book for any previous references to the problem or to a similar one.

Do not attempt to troubleshoot by using only complex system programs. Run the MAINDEC programs and select the shortest, simplest program available which exhibits the error conditions. MAINDEC programs are carefully written to include program loops for assistance in system and logic troubleshooting.

7.4.3 System Troubleshooting

Once the problem is understood and the proper program is selected, the logical section of the system at fault should be determined. Obviously, the program which has been selected gives a reasonable idea of what section of the system is failing. However, faults in equipment which transmits or receives information, or improper connection of the system, frequently gives indications similar to those caused by computer malfunctions.

Reduce the program to its simplest scope loop and duplicate this loop in a dissimilar portion of memory to verify, for instance, that an operation failure is not dependent upon memory location. This process can aid in distinguishing memory failures from processor failures. Use of this technique often pinpoints the problem to a few modules.

System troubleshooting is the first step towards isolating and repairing a machine malfunction. If the machine cannot be started, refer to the section on console checks (Paragraph 7.4.4). If the machine is running, determine that hardware, not software, is causing the problem. If the problem is occurring with DEC software (Compact, system monitor, etc.), obtain a certified copy of the program that is in good condition and attempt to repeat the malfunction. Generally, a recurring problem indicates a logic failure. If the problem occurs only with the user's software, an analysis of the failing program must be made. Use of the single time and single step is recommended. Some other items to check in the user's software are special bit assignments and functions. Are bank mode, page mode, etc., being used properly? Are memory fields being used properly? Is the program making unwarranted assumptions; i.e., assuming that the accumulator will be clear on start-up, etc?

In general, attempt to isolate the problem to a major system; then exercise that system with the MAINDEC diagnostics. Then, if necessary, proceed to logic troubleshooting and repair.

7.4.4 Console Checks

Assuming the problem is not clearly defined, the following checks should be made in an attempt to isolate basic machine faults. Any malfunctions observed at this time should be remedied before going further. Refer to Paragraph 7.5 for adjustments.

a. Power On — Does the power indicator on the console come on? Is the power up to specification? (Refer to Paragraph 7.3.2, steps i and j.)

b. Reset — Does the exec major state lamp light as well as the time state 3 lamp? Are the IR, MB, AC, PC, LINK and MO Registers cleared?

c. Deposit — Do the data switches transfer to the MO and MB? Do the address switches transfer to the OA? Are the fetch and time state 3 lamps on?

d. Examine — Do the address switches transfer to the OA and MO register? Do the contents of the location addressed appear in the MB register?

e. Deposit Next — Do the data switches transfer to the MO and MB? Do the contents of the OA register increment properly?

f. Examine Next — Do the OA and MO registers increment? Do the contents of each sequential memory location appear in the MB?

g. Deposit all 1s through memory using the repeat deposit next feature. Does the OA increment through TO the final available memory address +1? When the first non-existent memory address is accessed, does the computer stop in the fetch state at time state 2? Is the run light on and the non-existent address in the OA and MO? Is the MB cleared? Does the repeat speed vary with the on/off speed switch? With the machine hung with the run light on, the stop and reset keys should have to be depressed together to clear this condition. Either one by itself should have no effect.

h. Examine all of memory for bit loss.

i. With the address switches equal to 0, and with all 1s in memory, depress reset and start. Does the run light come on? Do the PC and MO registers increment up through 7777 (for 4K systems) or 17777 (for 8K or greater)? Does the AC equal all 1s? Do the time states 1, 2, and 3 lamps light?

j. Depress stop — Does the run light go out? Does the computer stop in fetch, time state 3?

k. Depress continue — Are the indications as in step i?

l. Deposit a Lac 100 in location zero, a jump to zero in location 1, and a 525252 in location 100. Depress start key. Does the machine cycle properly between the fetch and execute states? Does 525252 appear in the AC?

m. Using the execute switch try performing several instructions set in the data switches. Does the processor react properly to each instruction? Can the instructions be repeated by the use of the repeat function?

If the computer performs the above operations successfully, the timing of the CP, Memory, console and I/O are approximately correct and troubleshooting at a processor level can be commenced.

7.4.5 Processor Troubleshooting

Memory, I/O and CP processor troubleshooting is best achieved through the use of the MAINDEC diagnostic programs listed in Table 7-2. They provide the most rapid method of exercising these system areas.

Since the use of the MAINDECs requires the Readin function of the Teletype or high-speed reader to be operational this section of the processor must be checked first.

If the Readin section of the computer is not working, the operation of the reader IOTs should be checked by use of the execute repeat function of the console or by toggling in small read routines and comparing the processor operation against the appropriate timing and flow charts.

The extensive use of indicators (including two maintenance positions on the console indicator switch) in the PDP-15 and its peripheral devices was purposely included in the design. These indicators can, and should be used as aids in troubleshooting problems. You will probably find that many problems can be found and repaired through the use of these indicators without the need of any other test equipment.

Since the three main sections of the PDP-15 computer rely upon a request-grant system of operation most basic problems should be fairly easy to locate. As an example, the CP requests memory and waits for memory to acknowledge with an address acknowledge signal. If memory fails to answer, the processor will hang up with memory request still set, thus almost immediately pointing out the source of the trouble.

More subtle problems should be investigated through the use of the appropriate diagnostic program. Timing margins and module vibration often will point up intermittent problems more rapidly although care should be exercised in the use of module vibrating so as not to introduce more problems.

When troubleshooting peripheral devices a check of the signals on the I/O bus should be performed first, i.e., IOP pulses, device and subdevice select lines, etc., for proper levels and timing. Then the appropriate peripheral diagnostic should be run and the problem investigated by use of the diagnostic write-up.

If the diagnostic will not run, check the operation of the individual IOTs for the peripheral. Can the command register be loaded? Can data be transferred to and from the devices data register? Can the status registers of the device be transferred to the processor? In the case of a 3-cycle device a small routine to transfer blocks of data with +1 CA inhibit set will prevent wiping out core memory if there is a word count or current address problem. Once the fault has been isolated to a particular section, the next logical step is logic troubleshooting.

7.4.6 Logic Troubleshooting

Logic troubleshooting in the PDP-15 is best accomplished using the technique of reverse signal tracing. That is, when a malfunction has been isolated to a section of logic, some type of failure loop using either the reset-start switch combination, a small program, or a scope loop option in a diagnostic MAINDEC should be used. Then, by comparing the logic engineering drawings with the machine status, that portion of logic which is causing the failure becomes evident.

NOTE

An unconnected input to a gate, if not tied to a +3V pullup high, floats at approximately +1.9V.

Before attempting to troubleshoot the logic, make sure that proper and calibrated test equipment is available. Always calibrate the vertical preamplifier and probes of the oscilloscope before using. Make certain that the oscilloscope has a good ac ground, and keep the dc ground from the probe as short as possible.

Use the oscilloscope to trace signal flow through the suspected logic element. Oscilloscope sweep can be synchronized by control pulses or by level transitions which are available at individual module terminals on the wiring side of the logic. Care should be exercised when probing the logic to avoid shorting between pins. Shorting of signal pins to power supply pins can result in damaged components. Within modules, unused gate inputs are held at +3V.

WARNING

Standard safety practices should be observed when working with energized equipment. Remember that peripherals are not always connected to the mainframe power control and may be energized when the PDP-15 is off.

7.4.7 Module (Circuit) Troubleshooting

Engineering schematic diagrams of each module are supplied with each PDP-15 system in the Module Manual and should be referred to for detailed circuit information. Engineering block schematic diagrams are contained in Volume 2 of the Maintenance Manual.

Visually inspect the module on both the component side and the printed-wiring side to check for overheated or broken components, etc. If this inspection fails to reveal any signs of trouble or fails to confirm a fault condition observed, use the multimeter to measure resistance.

CAUTION

> Do not use the lowest or highest resistance ranges of the multimeter when checking semiconductor devices. The X10 range is suggested. Failure to heed this warning may result in damage to components.

Measure the forward and reverse resistance of diodes. Diodes should measure approximately 20 ohms forward and more than 1000 ohms reverse. (Front-to-back ratio should always be greater than 10 to 1.) If readings in each direction are the same and no parallel circuit paths exist, replace the diode.

Measure in both directions the emitter-collector, collector-base, and emitter-base resistances to transistors. Short circuits between collector and emitter or an open circuit in the base-emitter path cause most failures. A good transistor indicates an open circuit in both directions between collector and emitter. Normally 50 to 100 ohms exist between the emitter and the base, or between the collector and the base in the forward direction; an open circuit exists in the reverse direction. To determine forward and reverse directions, consider a transistor as two diodes connected back to back. In this analogy, PNP transistors would have cathodes connected together to form the base, and both the emitter and collector would assume the function of an anode. In NPN transistors, the base would be a common-anode connection; and both the emitter and collector would be the cathode.

Multimeter polarity must be checked before measuring resistance because many meters apply a positive voltage to the common lead when in the resistance mode.

Since integrated circuits contain complex circuits with only the input, output, and power terminals available, static multimeter testing is limited to continuity checks for shorts between terminals. Integrated circuit checking is best done under dynamic conditions and using a module extender to make terminals readily accessible. Using PDP-15 engineering drawings and the M-series module schematics, an integrated circuit may be located on a circuit board in the following ways:

a. Hold the module with the handle in your left hand (the component side facing you).

b. Integrated circuits are numbered starting at the contact end of the board in the upper right corner.

c. The numbers increase toward the handle.

d. When a row is complete, the next integrated circuit is located in the next row at the contact end of the board (see Figure 7-1).

e. The pins on each integrated circuit are located as shown in Figure 7-2.

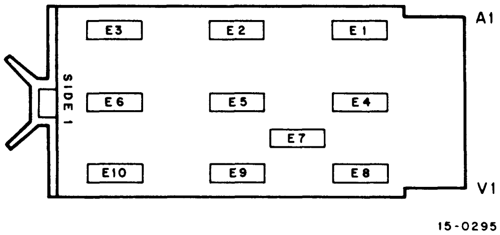


Figure 7-1 Integrated Circuit Location

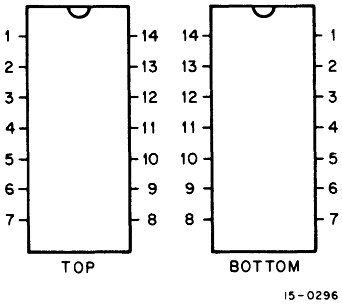


Figure 7-2 Integrated Circuit Pin Location

7.4.8 Repairs and Replacements

NOTE

DEC recommends replacing defective modules with modules of known quality on a one-for-one basis and returning the suspect module to a DEC field office for subsequent repair and/or replacement. If, however, for expediency, field repairs must be performed, it is imperative that the following procedure be strictly adhered to.

When soldering semiconductor devices (transistors, diodes, rectifiers, or integrated circuits, any of which may be damaged easily by heat, physical shock, or excessive electrical current), take the following special precautions:

a. Make sure the equipment is turned off.

b. Use a heat sink, such as a pair of pliers, to grip the lead between the nearest joint and device soldered.

c. Use a 6V iron with an isolation transformer. Use the smallest iron adequate for the work. Using an iron without an isolation transformer may result in excessive voltages present at the iron tip.

d. Perform the soldering operation in the shortest possible time to prevent damage to the component and delamination of the module-etched wiring.

e. Integrated circuits may be removed by using a solder puller to remove all excessive solder from contacts. Then, by straightening the leads, lift the integrated circuit from its terminal points. If it is not desirable to save the defective integrated circuit for test purposes, the terminals may be cut at the integrated body and each terminal removed from the board individually.

CAUTION

Never attempt to remove solder from terminal points by heating and rapping modules against another surface. This practice usually results in module or component damage. Always remove solder with a solder-sucking tool.

When removing any part of the equipment for repair and replacement, make sure that all leads or wires which are unsoldered, or otherwise disconnected, are legibly tagged or marked for identification with their respective terminals. Replace defective components with parts of equal or better quality and tolerance.

In all soldering and unsoldering operations in the repair and replacement of parts, avoid placing excessive solder or flux on adjacent parts or service lines. When the repair has been completed, remove all excess flux by washing junctions with a solvent such as trichlorethylene. Be very careful not to expose painted or plastic surfaces to this solvent.

7.4.9 Validation Tests

Always return repaired modules to the location from which they were taken. If a defective module is replaced by a new one during a repair period, tag the defective module, noting the location from which it was taken and the nature of the failure. When repairs are complete, return the repaired module to its original location and determine whether or not the repairs have corrected the problem.

To confirm the fact that repairs have been completed, run all tests which originally showed up the problem. If modules were moved during the troubleshooting period, return them to their original positions before running the validation tests.

Any time that a module is replaced by one from spares, return the module to its original location to confirm its defectiveness before initiating a repair procedure.

7.4.10 Recording (Log Book)

A log book is supplied with each PDP-15 system. Corrective maintenance is not complete until all activities are recorded in the log book. Record all data, indicating the symptoms displayed by the fault, the method of fault detection, the component at fault, and any comments which would be helpful in maintaining the equipment in the future. The log should be maintained on a daily basis, recording all operator usage and preventive maintenance results.

7.5 ADJUSTMENT PROCEDURE

The PDP-15 computer system has been purposely designed to eliminate numerous delay and timing adjustments which have proved to be frequent sources of trouble in other machines. The aim of this design is to aid the serviceman in troubleshooting and maintaining the PDP-15 system by eliminating time wasted in the adjustment of a large number of delays and timing chains. This feature is probably best pointed up in the PDP-15 memory where the only adjustment, other than the voltages, is the memory strobe delay.

NOTE

> All timing adjustments are taken at +1.5V point on the appropriate wave form.

7.5.1 DC Voltage Adjustments

a. +5V Memory, CP, I/O

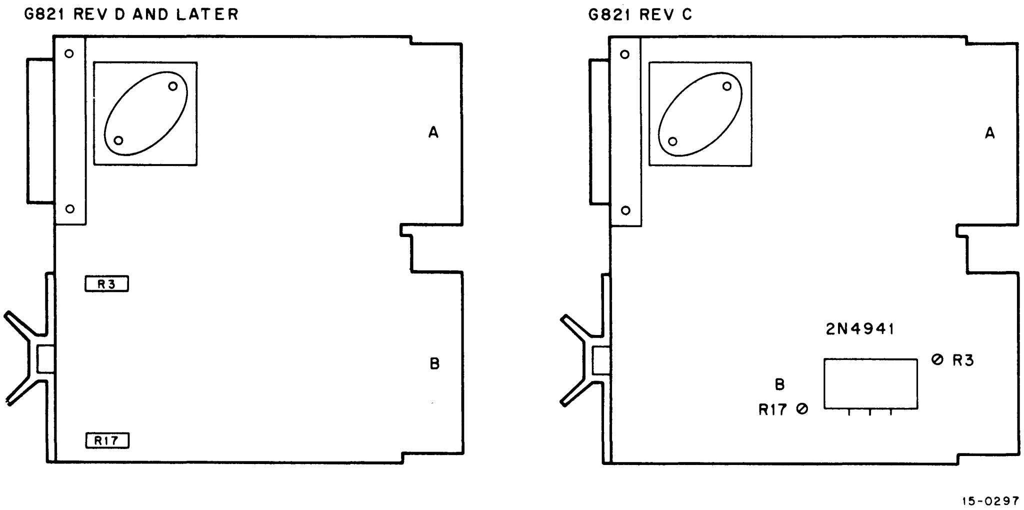
There is one G821 +5V regulator in each MM15 Memory and four in the CP, I/O processor section of the PDP-15 Computer System. Each one must be properly set up to supply the correct voltage to its associated logic (refer to MM15-20).

b. Each G821 module also has a low voltage detector adjustment R17, (see Figure 7-3), and this adjustment should be made at this time to ensure that the logic will not have to operate at a marginal voltage setting, which could induce errors. To set the low voltage detector, adjust the +5V output with R3 for 4.75 volts. With the regulators now providing the lower I.C. voltage limit, adjust R17 on each G821.

While observing the panel light (grain of wheat bulb on wire wrap panel) associated with the G821 being adjusted, turn R17 counterclockwise until the panel light goes out. Then turn it clockwise until it just comes back on. Then proceed with the +5V adjustment described below.

c. +5V Adjustment

Look at pins A01A2 in each memory bank; E01A2, H01A2, K01A2, and M01A2 in the CP/IO and adjust R3 on the appropriate G821 for +5V to ground with your multimeter. (See Figure 7-3 for potentiometer position on module.)



G821 REV D AND LATER

G821 REV C

15-0297

Figure 7-3 G821 Module Adjustment

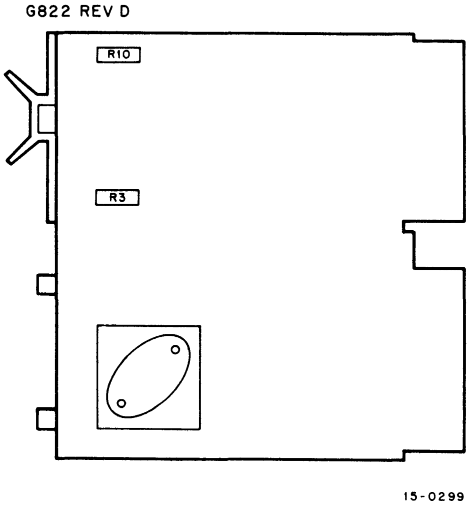
d. -6V Memory Slice Voltage

Each MM15 memory contains a G822 slice and threshold voltage regulator.

To adjust the -6V slice look at pin C01B2 in each memory and adjust R3 (see Figure 7-4) until -6V to ground is measured on the multimeter.

e. Memory Threshold Voltage

To adjust the memory threshold voltage with the multimeter, measure between pin C01V2 in memory and ground. Adjust POT R10 on the G822 module (see Figure 7-4) until a reading of 4V is obtained.



G822 REV D

15-0299

Figure 7-4 G822 Module Adjustment Locations

f. -24V Memory Voltage

To adjust the -24V memory voltage in each memory, measure between pin C17R1 and ground. Adjust POT R14 on the G823 module (see Figure 7-5) until a reading of -24V is obtained on the meter. This is an approximate setting.

This voltage should give you approximately 400 mA of current when a current probe is used on the current loops in memory (see Figure 7-6). Further adjustment of R14 may be necessary to obtain exactly 400 mA.

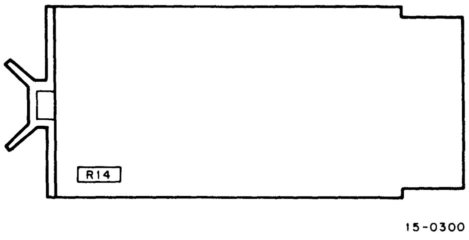


Figure 7-5 G823 Module Adjustment Locations

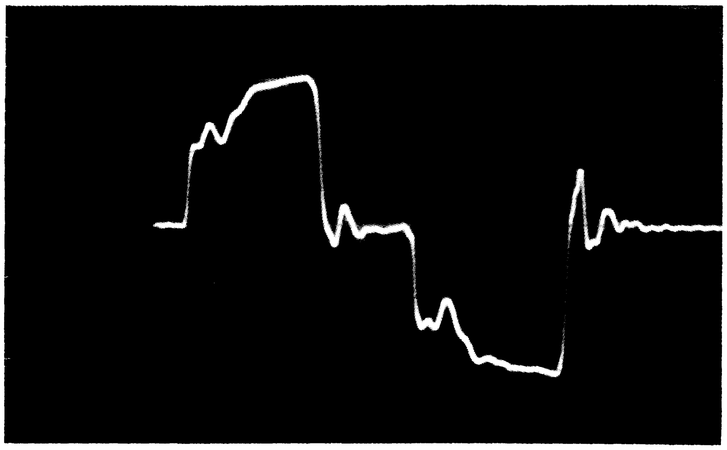


Figure 7-6 MEM Current

7.5.2 Memory Timing Adjustments

7.5.2.1 Memory Strobe — As stated before, memory strobe is the only variable timing adjustment in memory. To adjust memory strobe use an oscilloscope with a dual trace. Sync on chan #1 and with current probe #1 look at the RDX current loop B04S2-B05T2. Trigger the scope positive.

With probe #2 look at C04C1 (memory strobe H). Set the sweep speed to 50 ns/cm and use a chopped trace. Toggle in a JMP 0 in location 000000 of the memory being adjusted and depress the start key. Adjust the POT on A17 until the leading edge of memory strobe occurs 115 ns after the leading edge of 10% point of RDX current (see Figure 7-7).

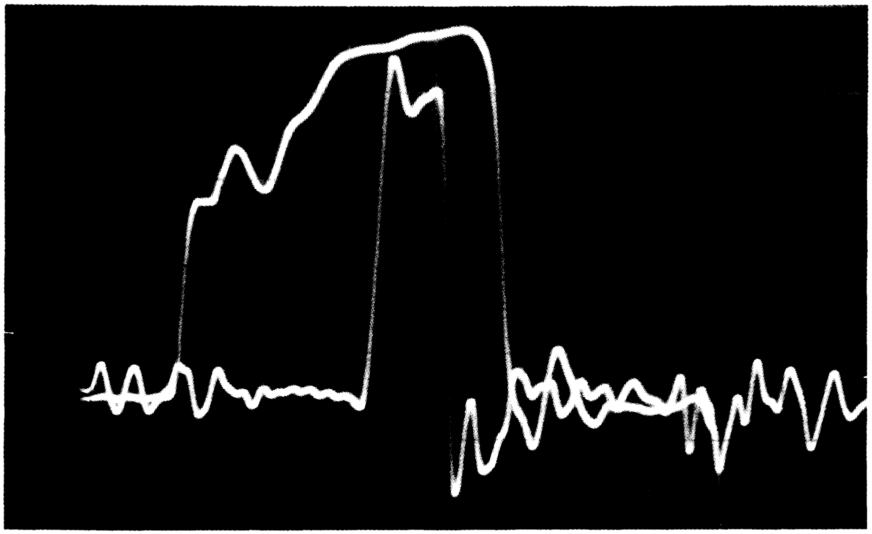


Figure 7-7 MEM Strobe Delay

Deposit 777777 in Loc 0 and a JMP 0 in Loc 1 of the memory being adjusted with probe #1. Look at C04C1. With probe #2 look at C05A1, a sense amp output test point. The wave forms should look like Figure 7-8. Use scope settings as described in Figure 7-8.

7.5.3 CP Timing Adjustment

7.5.3.1 CP Clock — With probe #1 sync on, and Look at pin E30F2 in the CP (HS clock H), adjust the pot on E30 to give approximately a 65 ns cycle time (85 ns for machines with parity) (see Figure 7-9).

Deposit the following program in memory and loop on it.

```
Loc     100/     ISZ     103     440103
        101/     JMP     100     600100
        102/     JMP     100     600100
```

With probe #1 sync on, and look at E30K2 (TS01 H) (see Figure 7-10). Readjust the pot on E30 to give a 260 ns duration (350 ns for parity machines).

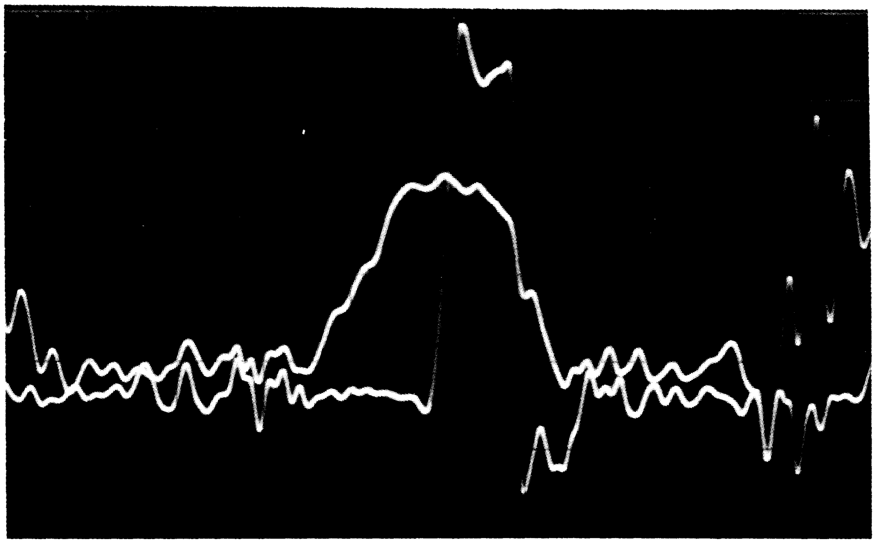


Figure 7-8 MEM Strobe/Sense Amp Output

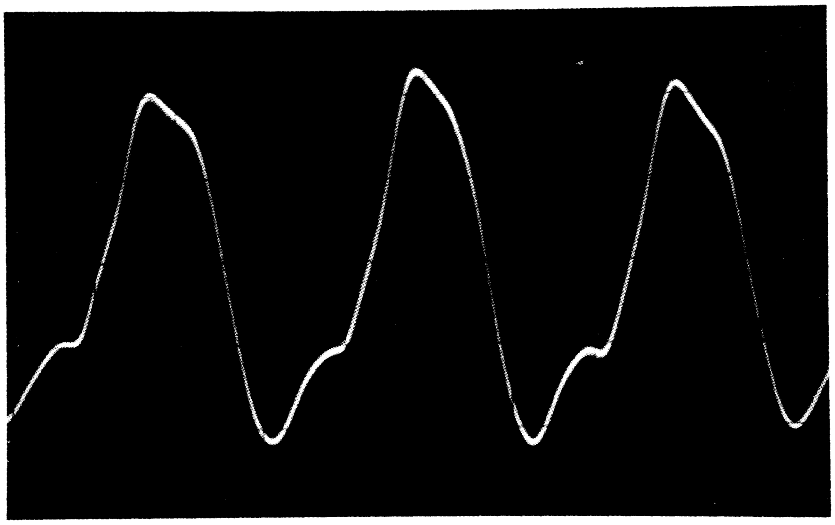


Figure 7-9 CP Clock

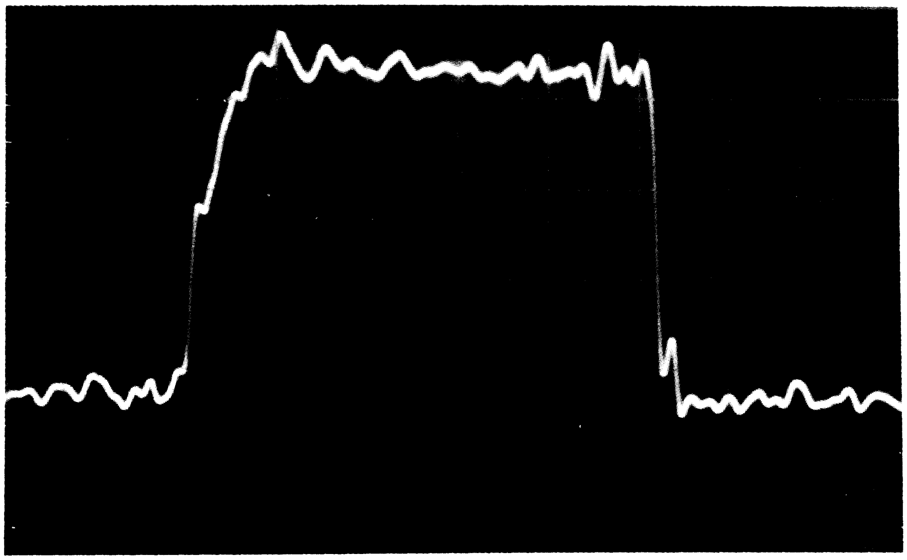


Figure 7-10 TS01 Duration

Now change the time base of the scope and measure the time between the leading edge of one TS01 to the leading edge of the next (see Figure 7-11). Typically, this time should be 780-810 ns. (For machines with Memory Protect Option 800-850 ns; for machines with memory relocate option 800-950 ns; for machines with parity option 1.02-1.08 ns.) The variations in time are due to synchronization circuitry, cable and circuit delays.

The CP timing can be varied as per Table 7-3 to test the operation of the synchronization circuitry and as a margin test to aid in troubleshooting. Data path speeds that are deteriorating may show up as the processor speed is increased, and they can be repaired before a problem at normal speed occurs.

7.5.4 I/O Timing

7.5.4.1 I/O Clock — The I/O clock is adjusted by observing with probe #1 pin N21D2 (I/O Clock) (see Figure 7-12). Adjust the pot on N21 so that the pulses occur every 250 ns.

7.5.4.2 Console Clock — With probe #1 sync on and look at pin N28D2, Clk in H (see Figure 7-13). Adjust the pot on N28 so that the pulses occur every 27.5 ns.

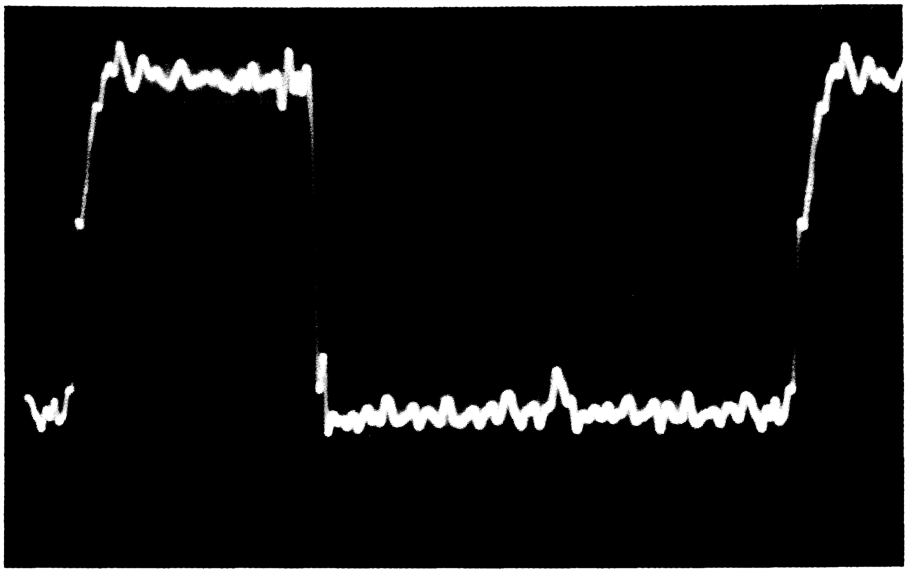


Figure 7-11 Machine Cycle Time

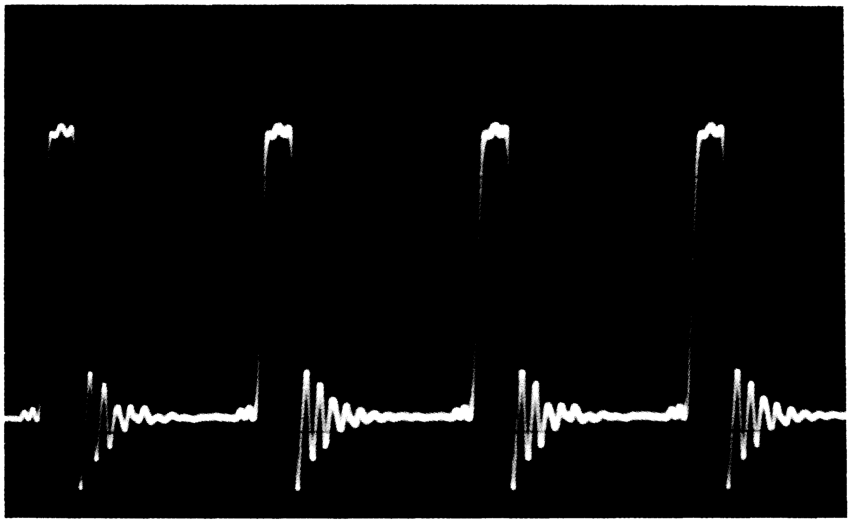


Figure 7-12 I/O Clock

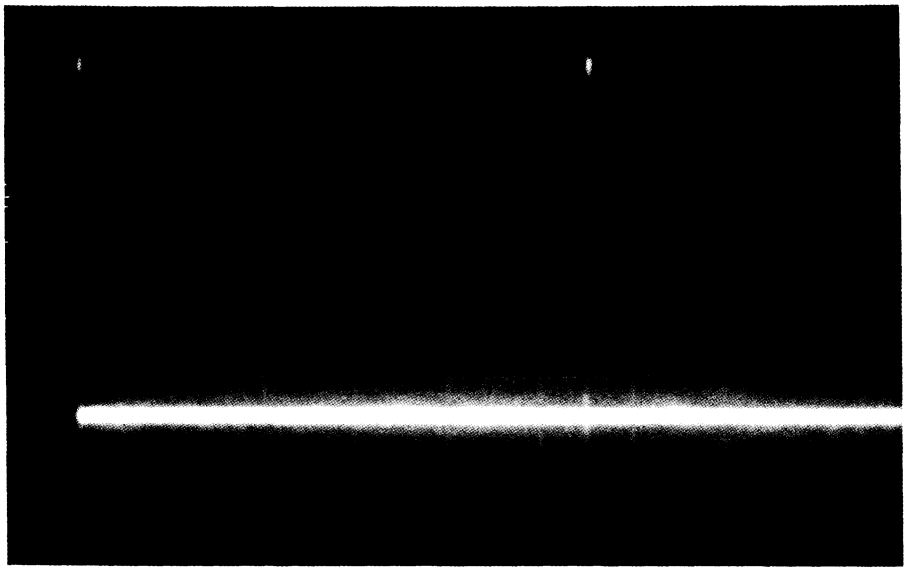


Figure 7-13 Console Clock

7.5.4.3 Teletype Clock — With probe #1 sync on and Look at pin M28K2 - T to Clock L (see Figure 7-14), adjust the pot on M28 so that the clock pulse occurs every 4.5 ms.

This adjustment is fairly critical and may need to be slightly readjusted if type-out errors are noted when running the Teletype diagnostics.

7.5.5 Timing Adjustment Summary

Table 7-3 gives a summary of all the PDP-15 timing adjustments and the margin variations which can be obtained under normal conditions.

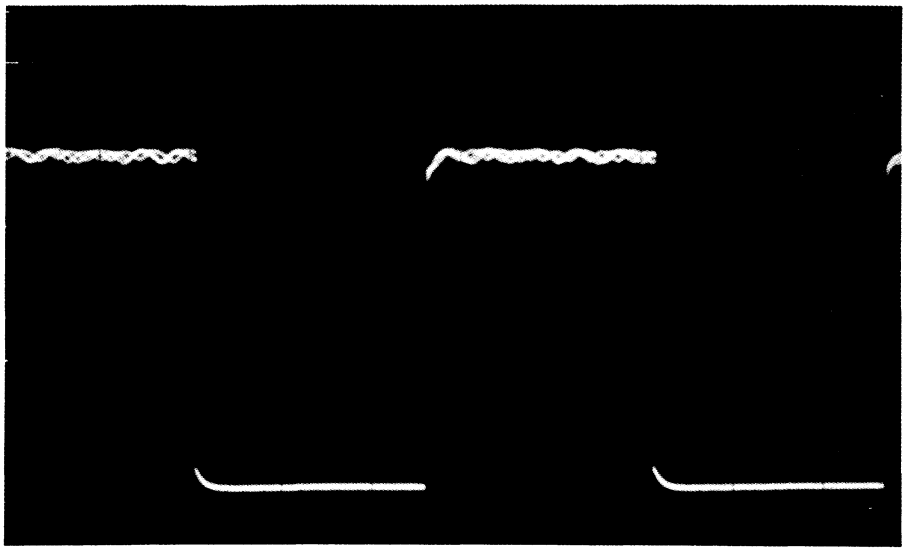


Figure 7-14 Teletype Clock

Table 7-3
PDP-15 Timing Adjustment Summary

Adjustment	Monitor Point	Nominal Setting	Minimum High Margin	Maximum Low Margin	Machine Cycle Time at Nominal Setting	Conditions
Console	N28D2	27.5 μs	N/A	N/A	N/A	
CP Clock	TS01 E30K2	260 ns	350 ns	230 ns	790-820 ns	No KM15, KT15 or MP15
CP Clock	TS01 E30K2	260 ns	350 ns	250 ns	800-850 ns	With KM15 and no MP15
CP Clock	TS01 E30K2	260 ns	350 ns	250 ns	800-900 ns	With KT15 and no MP15
CP Clock	TS01 E30K2	350 ns	375 ns	320 ns	1.05 μs	With MP15 and no KT15 or KM15
CP Clock	TS01 E30K2	350 ns	375 ns	320 ns	1.0-1.2 μs	With MP15 and either KT15 or KM15

Table 7-3 (Cont)
PDP-15 Timing Adjustment Summary

Adjustment	Monitor Point	Nominal Setting	Minimum High Margin	Maximum Low Margin	Machine Cycle Time at Nominal Setting	Conditions
I/O Clock	N21D2	250 ns	N/A	N/A	1.0 μs I/O time	
Teletype Clock	M28K2	4.5 ms (220 Hz)	N/A	N/A		Double check by running Teletype diagnostic

Digital Equipment Corporation
Maynard, Massachusetts

printed in U.S.A.