PCM-12

OMEGA

System Operating Manual

PC/M, Inc.

Dublin, CA

PCM

SYSTEM
OPERATING MANUAL
for
PCM-12 OMEGA
MICROCOMPUTER

PC/M, Inc.
Dublin, CA USA

PC/M, Inc.
Dublin, CA

June, 1980
Document #12890

TABLE OF CONTENTS

		Page Number
1.0	INTRODUCTION	1-1
2.0	FRONT-PANEL OPERATION	2-1
3.0	BUS STRUCTURE	3-1
4.0	SOFTWARE CONSIDERATIONS	4-1
5.0	FUNDAMENTAL MACHINE OPERATION	5-1
6.0	12020 STATIC MEMORY MODULE	6-1
7.0	12540 MEMORY EXTENDER/STACK/REAL-TIME CLOCK MODULE	7-1
8.0	12070 HIGH-SPEED PAPER TAPE READER/PUNCH INTERFACE	8-1
9.0	POWER SUPPLY CONSIDERATIONS	9-1
10.0	12080 AUDIO CASSETTE RECORDER INTERFACE	10-1
11.0	12560 TTY/CRT SERIAL INTERFACE	11-1
12.0	12390 4K NON-VOLATILE CMOS EPROM/RAM MEMORY MODULE	12-1
13.0	12210 4K NON-VOLATILE CMOS RAM MEMORY MODULE	13-1
14.0	12310 12-BIT PARALLEL I/O INTERFACE MODULE	14-1
15.0	12440 DEC-COMPATIBLE FLOPPY-DISK INTERFACE	15-1
16.0	12340 POWER-FAIL & NON-VOLATILE CLOCK/CALENDAR MODULE	16-1
17.0	12410 8K DYNAMIC RAM MEMORY MODULE	17-1
18.0	12360 LETTER-QUALITY PRINTER INTERFACE MODULE	18-1
19.0	12490 POSITIVE BUS CONVERTER MODULE	19-1
20.0	12460 MATRIX PRINTER INTERFACE MODULE	20-1
21.0	12430/12435 DEC-COMPATIBLE HARD DISK CONTROLLER	21-1
22.0	12170 OPTICALLY-ISOLATED PARALLEL INPUT MODULE	22-1
23.0	12180 OPTICALLY-ISOLATED PARALLEL OUTPUT MODULE	23-1
24.0	12480 12-BIT ANALOG-TO-DIGITAL CONVERTER MODULE	24-1
25.0	12470 12-BIT DIGITAL-TO-ANALOG CONVERTER MODULE	25-1
26.0	12120 4K CONTROL-PANEL MEMORY EXPANSION MODULE	26-1
27.0	12110 8K EPROM/RAM MEMORY MODULE	27-1
28.0	12520 4K 3-PORT DMA RAM MEMORY MODULE	28-1
29.0	12140 SERIAL INTERFACE WITH MODEM CONTROL	29-1
30.0	12260 4-PORT SERIAL INTERFACE MODULE	30-1
31.0	12240 RELAY-TYPE PARALLEL OUTPUT MODULE	31-1

SECTION 1.0

INTRODUCTION

		Page Number
1.1	ORGANIZATION OF MANUAL	1-2
1.2	COMPARISON OF PCM-12 WITH PDP-8/E	1-2

SECTION 1.0

INTRODUCTION

The purpose of this manual is to serve as a general guide to the PCM-12 Omega microcomputer system. The PCM-12 is a small, general-purpose computer employing a 12-bit word length, utilizing the Intersil IM6100 microprocessor for its Central Processor Unit (CPU). The IM6100 device employs the same binary instruction set as the PDP-8 series of minicomputers manufactured by Digital Equipment Corporation of Maynard, Massachusetts (see note below). Therefore, the PCM-12 "microcomputer" is software-compatible with the entire PDP-8 computer family, including the PDP-8, PDP-8/S, PDP-8/L, PDP-8/M, PDP-8/F, PDP-8/E, PDP-8/A and the recently-introduced VT-78 DECstation, as well as those variants of the VT-78 such as the WS-78 and WD-78 systems.

For the user wishing to become familiar with PDP-8 (or PCM-12) software, reference should be made to Section 4.0 of this manual, where a list of pertinent DEC documentation will be found.

One of the most important virtues of a computer designed around a microprocessor is that, perhaps for the first time for most users, the computer's hardware operations become understandable. Rather than being a maze of MSI and SSI logic, the CPU is reduced to a single LSI package with a modest number of inputs and outputs. Once the user familiarizes himself with the operation of the microprocessor (e.g., by reading Section 5.0 of this manual), the operation of the entire computer becomes simple to understand. The microprocessor itself, although a complex device embodying all the functionality of a powerful CPU, can be treated as a "black-box" - simply stimulate it with a binary-coded instruction, and it performs the indicated operation. How it accomplishes this feat, what lurks behind the "silicon curtain", is of no concern to the user. If the processor "chip" should malfunction (a very rare occurence), one simply discards it and plugs another

---

NOTE - The following are registered trademarks of Digital Equipment Corporation, Maynard, Massachusetts: DEC, DDT, DECtape, DIBOL, DIGITAL, EDUSYSTEM, FOCAL, OMNIBUS, OS-8, and PDP.

into the same socket. The user is able to concentrate on applying the computer to his job, comfortable in the feeling that he understands his hardware down to the throw-away module level.

This manual discusses the IM6100 device as it is applied in the PCM-12 system. No attempt has been made to explain the complete, detailed internal operation of the "chip" itself, since Intersil has already provided adequate documentation on the IM6100 and its family of support devices. The would-be hardware "expert" is referred to the data sheets on the IM6100 family as supplementary material to this manual. Requests for Intersil documentation can be made by writing them at: Intersil, Inc., 10900 N. Tantau Avenue, Cupertino, California 95014, or by calling (408) 996-5000.

1.1 ORGANIZATION OF MANUAL

This manual is organized into a series of sections. Each section covers a topic of major interest to the user. Each manual provided by PC/M gives all the required documentation for the hardware purchased by the user. When the user acquires additional accessory modules for his system, additional sections for the operating manual are also provided, which document the new hardware. (For example, while the basic operating manual contains six major sections, a user with the 12560 TTY/CRT Interface, 12540 Memory Extender/Stack/Real-Time Clock module and 12440 Floppy-Disk Interface would also receive Sections 11, 7 and 15, which document those modules, respectively.)

1.2 COMPARISON OF PCM-12 WITH PDP-8/E

To the PDP-8/E user, the PCM-12 will seem quite familiar. This section compares the two machines in several areas of interest.

The PCM-12 is designed around a microprocessor CPU. Therefore, all its major operational characteristics are defined by the microprocessor. Using a 4.0 MHz system-clock crystal, the IM6100 has a basic cycle time of 2.5 microseconds. Basic cycle time for the PDP-8/E is 1.2 to 1.3 microseconds.

The binary instruction sets for the PCM-12 and PDP-8/E are indentical, which is why they are mutually software-compatible. Software written for either machine will run properly on the other, with the one exception that the PCM-12 cannot be used in "time-share" applications. Any other software incompatibility would result from running software on the PCM-12 which required the higher operating speed of the PDP-8, or attempting to use

software which was written for hardware not existing in the user's system (two obvious limitations). Happily, considering these few exceptions, the two computers will be found to have an extremely large overlap in their respective software "appetites".

The hardware design of the PCM-12 and the PDP-8/E are quite different, particularly since the PCM-12 is microprocessor based. Therefore, the bus structures on the two machines are different, and modules from one system cannot be directly plugged into the other. However, this is not a major consideration for the PCM-12 user, since PC/M makes available (at very attractive prices) nearly every interfacing and memory module the user could require in his system. For example, for the user who wishes to use older DEC-style positive-bus peripherals in his PCM-12 system, these can be interfaced through the 12490 Positive Bus Converter module.

Finally, the memory system in the PCM-12 is dissimilar to that in the PDP-8. The latter uses core memory, while the PCM-12 employs modern semiconductor memory exclusively.

SECTION 2.0

PCM-12 OMEGA FRONT-PANEL OPERATION

		Page Number
2.1	HARDWARE-DRIVEN CONTROLS AND INDICATORS	2-1
2.1.1	POWER Switch, Power Receptacle and Fuse	2-1
2.1.2	AUX Switch	2-2
2.1.3	SW Switch	2-2
2.1.4	RESET Switch	2-2
2.1.5	RUN/HALT and CONT/SNGL INST Switches	2-3
2.1.6	XTAL CLK/SNGL CLK Switches	2-3
2.1.7	RUN, LINK, IFETCH and XTA Lamps	2-3
2.2	SOFTWARE-DRIVEN CONTROLS AND INDICATORS	2-3
2.2.1	MEMORY ADDRESS Display	2-4
2.2.2	Display Lamps	2-4
2.2.3	SWITCH REGISTER	2-5
2.2.4	Rotary Switch	2-5
2.2.5	TIMER Switch	2-5
2.2.6	Function Switches	2-7
2.2.6.1	ADDR LOAD Switch	2-7
2.2.6.2	DECR ADDR Switch	2-7
2.2.6.3	EXTD ADDR Switch	2-7
2.2.6.4	DEP Switch	2-8
2.2.6.5	EXAM Switch	2-8
2.2.6.6	BOOT Switch	2-9
2.3	FRONT PANEL OPERATION WITH EXTENDED MEMORY	2-10
2.4	COMPARISON OF PCM-12 AND PDP-8/E FRONT PANELS	2-10
2.4.1	RESET Switch	2-11
2.4.2	SNGL CLK Switch	2-11
2.4.3	DEP Switch	2-11
2.4.4	EXAM Switch	2-11
2.5	SPECIAL 12530 INSTRUCTIONS	2-12
2.6	12530 LOGIC OPERATION	2-13
2.6.1	REAL-TIME DISPLAYS	2-16
2.6.2	TIMER OPERATION	2-16
2.7	CONTROL PANEL SOFTWARE ROUTINE	2-17

SECTION 2.0

PCM-12 OMEGA FRONT PANEL OPERATION

The 12530 front panel for the PCM-12 Omega microcomputer is an improved version of the original 12030 front panel for the PCM-12. The 12530 includes real-time operation of the address and memory-data display lamps, full decoding of all control-panel memory addresses, two-speed timer operation, a user-addressable set of display lamps, and built-in bootstraps for several mass-storage units as well as a multi-field paper-tape loader.

Operation of the PCM-12 front panel is essentially very similar to the front panel on DEC's PDP-8/E minicomputer. However, there are some differences caused by the fact that many of the PCM-12's front-panel operations are software driven, and by the fact that the PCM-12 front panel includes some functions that the PDP-8 does not. The following paragraphs describe the PCM-12's front-panel switches and indicators in detail. For the reader who is already familiar with the PDP-8/E, Section 2.4 summarizes the major differences bwtween front-panel operations on the two machines.

2.1 HARDWARE-DRIVEN CONTROLS AND INDICATORS

The controls and indicators on the PCM-12 front panel can be roughly classified into two groups - those which are hardware driven, and those which are software driven, requiring interaction with the 12530 control-panel software routine for their proper operation. The hardware-driven controls and indicators are dRscussed in this Section 2.1, while the software-driven devices are described in Section 2.2. Some controls and indicators are both hardware and software driven and so are touched upon in both Section 2.1 and Section 2.2.

2.1.1 POWER Switch, Power Receptacle and Fuse

The POWER Switch, power receptacle and fuse for the PCM-12 Omega microcomputer are actually located on the rear panel of the machine. The power receptacle contains both the fuse and a programming card for selection of the proper line voltage.

The programming card allows selection of four different nominal line voltages: 100 volts, 120 volts, 220 volts or 240 volts, at either 50 or 60

Hz. The input line voltage should be within plus/minus 10 percent of the nominal value. Set-up of the power receptacle for the desired line voltage is accomplished by correctly positioning the programming card behind the clear plastic window in the receptacle. The voltage for which the card is set can be read through the window. As shipped by PC/M, this will normally be "120" for domestic units and "240" for units shipped internationally. To change to another line voltage, simply slide the window to the side, remove the fuse by pulling the fuse-removal lever, and pull the programming card out with a pliers. Then turn the card and re-insert it into the receptacle. Be sure the proper line voltage is now readable through the window, and replace the fuse.

DO NOT ATTEMPT OPERATION WITHOUT FIRST SETTING THE POWER RECEPTACLE FOR THE PROPER LINE VOLTAGE. PERMANENT DAMAGE MAY OTHERWISE RESULT.

A 3.0 ampere medium-blow fuse should be used for 100- or 120-volt operation. For 220- or 240-volt operation, use a 1.5 ampere fuse.

2.1.2 AUX Switch

This is an auxiliary control switch which, in the "up" position, grounds line 42 on the PCM-12 bus (otherwise line 42 just "floats"). It may be used to manually control any function; for example, activating a peripheral device.

2.1.3 SW Switch

This switch is identical in operation to the AUX switch, except that in the "up" position it grounds line 41 on the PCM-12 bus.

2.1.4 RESET Switch

This switch, when raised to the "up" position, grounds line 18 on the PCM-12 bus. On the CPU module this causes the IM6100 Accumulator to be cleared, the Program Counter to be set to 7777_8 and the CPU to go to the HALT state. Line 18 is also used on most other modules to clear flip-flops, initialize control logic, etc.

2.1.5 RUN/HALT and CONT/SNGL INST Switches

The RUN/HALT switch operates in conjunction with the CONT/SNGL INST switch. In the "down" position the RUN/HALT switch causes the CPU to go to the HALT state. With this switch in the "down" position, however, activation of the CONT/SNGL INST switch causes the machine to fetch and execute the next sequential instruction, and then halt once more. This "single-instruction" mode of operation can be used to check out a program one instruction at a time.

When in the RUN ("up") position, the RUN/HALT switch enables the CPU to enter the RUN mode. The machine will start running when the CONT/SNGL INST switch is momentarily activated.

2.1.6 XTAL CLK/SNGL CLK Switches

This pair of switches controls the CPU clock source. When the toggle switch is in the XTAL CLK position, the CPU receives a continuous stream of clock pulses from a crystal oscillator on the CPU module. When the toggle switch is in the SNGL CLK position, a single clock pulse is developed each time the push-button switch is activated, advancing the CPU by one-half T-state. This "single-clock" operation is normally used in the RUN mode to examine program operation one half-state at a time.

If the machine is in the single-instruction mode (RUN/HALT switch "down"), the SNGL CLK push-button can also be used to "micro-examine" operation of a single instruction. This is done by putting the RUN/HALT and XTAL CLK/SNGL CLK toggle switches "down", pushing the CONT/SNGL INST push-button once, then clocking the instruction through with the SNGL CLK push-button.

2.1.7 RUN, LINK, IFETCH and XTA Lamps

These four lamps constantly monitor the status of the corresponding lines on the PCM-12 bus. When any of these lines is asserted, the corresponding lamp is lighted.

2.2 SOFTWARE-DRIVEN CONTROLS AND INDICATORS

The remaining front-panel controls and indicators depend, for their operation, on a control-panel software routine stored in read-only memory

devices on the 12530 module. The TIMER and the six Function Switches are each capable of generating a control-panel interrupt to the CPU. During the execution of the control-panel interrupt service routine, normal I/O-device interrupt requests are ignored, as are DMA requests. A control-panel interrupt request is granted even if the CPU is in the HALT state; the machine is forced into the RUN state for the duration of the control-panel routine, then returns to the HALT state.

The control-panel interrupt routine and several bootstrap routines are stored in three 512x4 PROM chips. These devices, and three 16x4 RAM chips which the routines use as scratchpads, are located on the 12530 front-panel printed-circuit board. See the front-panel schematic diagram, Drawing 12532.

2.2.1 MEMORY ADDRESS Display

When the machine is running a program in main memory under XTAL CLK control and the TIMER is not on (the normal mode of operation), these 12 lamps display the address of the current instruction. As the instruction is fetched from memory, its address is latched into these lamps. Therefore, the display runs in "real time". This can be handy for determining the approximate location in memory where the program is currently running.

When in the SNGL CLK mode of operation, these lamps display the address of the current instruction, whether the program is running from main memory or control-panel memory.

Loading of the MEMORY ADDRESS lamps with address data, as described in this section, is accomplished by hardware logic on the 12530 module; it does not utilize any part of the control-panel software. For operation of this display in XTAL CLK mode under control of the control-panel software, see discussion of the TIMER and Function Switches below.

2.2.2 Display Lamps

When the machine is running a program in main memory under XTAL CLK control and the TIMER is not on , these 12 lamps normally display the current instruction. As the instruction is fetched from memory, it is latched into the Display lamps. Thus, the display runs in "real time". (An exception to this mode of operation for the Display lamps takes place when the lamps are used, under program control, as a user-data display through use of the memory-mapped 0170-0172 control-panel memory locations. See Section 2.5 below.)

In the SNGL CLK mode of operation, these lamps are effectively directly connected to the bus DX lines (lines 29-40), and therefore constantly display the state of those lines. Note, however, that the DX lines only carry valid data during some T-states. See the timing diagram shown in Figures 5-1 and 5-2 to ascertain when DX data is valid.

For operation of the Display lamps in XTAL CLK mode under control of the control-panel software, see discussion of the TIMER and Function Switches below.

2.2.3 SWITCH REGISTER

These switches are used in conjunction with the ADDR LOAD, DEP and EXTD ADDR Load switches to load data into the CPU Program Counter, main memory, and the 12540 Memory Extender IF and DF registers (when present in the system), respectively.

2.2.4 Rotary Switch

This switch selects the data to be displayed in the Display lamps when the TIMER is on or one of the six Function Switches is depressed. The MD position selects Memory Data, AC position selects the CPU Accumulator contents, MQ position selects the CPU's MQ register contents, and the Status bits are selected by the STATUS position. The Status bits are:

- 0: CPU Link flip-flop state.
- 1: not used.
- 2: CPU Interrupt Request line (lamp lighted when interrupt request active and not inhibited by bus line 28).
- 3: Interrupt Inhibit Flip-Flop (this f/f is located in the IM6102 device on the 12540 Memory Extender module.) Lamp is lighted when the f/f is set, i.e., inhibiting device interrupt requests.
- 4: The CPU's Interrupt Enable Flip-Flop. Lamp is lighted when interrupts are enabled.
- 5: not used.
- 6-8: The currently-selected Instruction Field.
- 9-11: The currently-selected Data Field.

2.2.5 TIMER Switch

This switch, in the "up" position, activates a timer on the front-panel printed-circuit board. The timer then generates control-panel interrupts at a

rate determined by the position of the RUN/HALT toggle switch. When the RUN/HALT switch is in the RUN position, CP interrupts are generated at a rate of one or two per second. When the switch is in the HALT state, interrupts are generated at a rate of about 25 per second. [The timer is unable to generate CP interrupts, however, when the CPU has started execution of an I/O device interrupt routine (INTGNT, bus line 74, high) or has recognized a Direct Memory Access request (DMAGNT, bus line 73, high), when timer-generated control-panel interrupts are barred.]

The timer has two primary purposes. The first is to interrupt the normal operation of the machine when it is running (RUN/HALT switch "up"), to take a "snapshot" of the machine state and display it on the front-panel indicators. The second purpose is similar - to show the microcomputer's state immediately after each instruction when the user has the machine in the single-instruction mode (RUN/HALT switch "down"). [NOTE: In single-instruction mode, the TIMER will not show the machine state after each instruction when either the INTGNT or DMAGNT bus line is high, because control-panel interrupts are barred when either of these lines is asserted. Likewise, the TIMER will not show the state of the machine after a ION or RTF instruction, because no CP interrupt can take place for one instruction after a ION or RTF.]

Whenever the timer generates a control-panel interrupt, the control-panel software routine causes the current state of the Program Counter to be displayed in the MEMORY ADDRESS lamps and, if the Rotary Switch is in the MD position, the next instruction itself is shown in the Display lamps. If the Rotary Switch is in the AC, MQ or STATUS positions, the Display lamps show the current state of the Accumulator, MQ register or Status bits, respectively.

Note that when the user has the machine in single-instruction mode with the TIMER off, the real-time action of the front-panel displays will always show the address of the last instruction executed in the MEMORY ADDRESS lamps, and the last instruction itself in the Display lamps. If the TIMER is on, however, the MEMORY ADDRESS lamps will show the address of the next instruction, and the Display lamps will show the next instruction itself if the Rotary Switch is in the MD position. With the TIMER on, the state of the Accumulator, MQ register or Status bits can also be determined simply by turning the Rotary Switch.

The TIMER is disabled in the SNGL CLK mode of operation, so that it cannot "steal" all the manually-produced clock pulses.

2.2.6 Function Switches

Six momentary-action function switches on the front panel are capable of generating a control-panel interrupt when the machine is in the HALT state. These are the EXAM, DEC ADDR, BOOT, ADDR LOAD, EXTD ADDR and DEP switches. When the machine is running these switches are disabled. They are normally used with the machine halted and the TIMER off.

2.2.6.1 ADDR LOAD Switch

Activation of the ADDRess LOAD switch causes the CPU Program Counter to be loaded with the contents of the front-panel SWITCH REGISTER, and this new address to be displayed in the MEMORY ADDRESS display. The Display lamps are then updated with either the data in the main-memory location indicated by the MEMORY ADDRESS display (MD position), CPU Accumulator contents (AC position), CPU MQ register contents (MQ position), or the Status bits (STATUS position), depending on the position of the Rotary Switch. The Program Counter is used as the pointer for deposits to, and examinations of, main memory locations from the front panel.

2.2.6.2 DECR ADDR Switch

Operation of the DECRement ADDRess switch is identical to operation of the ADDR LOAD switch, except that rather than loading the Program Counter, it is simply decremented by one state.

2.2.6.3 EXTD ADDR Switch

The EXTendeD ADDRess Load switch causes bits 6-8 and 9-11 of the SWITCH REGISTER to be loaded into the Instruction Field and Data Field registers, respectively, on the 12540 Memory Extender module, when this board is present in the system. (When this module is not present in the system, this load, of course, does not occur and both Instruction Field and Data Field stay set to 000_2.) The current state of the Program Counter is then displayed in the MEMORY ADDRESS display and, if the Rotary Switch is in the STATUS position, the new Status bits are shown in the Display lamps. If the Rotary Switch is in one of the other three positions, the extended-address load still

takes place, but the new Status bits are not shown in the Display lamps. Rather, the latter display shows the contents of the CPU Accumulator (AC position), the CPU MQ register (MQ position), or the memory location pointed to by the MEMORY ADDRESS lamps (MD position).

2.2.6.4 DEP Switch

Activation of the DEPosit switch causes the data set up in the SWITCH REGISTER to be deposited into the main-memory location (of the currently-specified data field) whose address is shown in the MEMORY ADDRESS display. (Note that for all main-memory references to/from the front panel, the CPU Program Counter is used as the pointer.) Then, if the Rotary Switch is in the MD position, the new memory data is shown in the Display lamps, and the CPU Program Counter is incremented. If the Rotary Switch were in any other position, the memory load would take place as indicated, but the data shown in the Display lamps would correspond to the position of the Rotary Switch, and the Program Counter would not be incremented.

Since the software routine increments the CPU Program Counter after a memory load (if the Rotary Switch is in the MD position), when loading a series of sequential memory locations it is only necessary to manually set up the first address with the ADDR LOAD switch. Thereafter each sequential address is automatically set up and it is only necessary to set the memory data for each location into the SWITCH REGISTER, and activate the DEP switch. But, as is the case with the EXAM switch, it is important to remember that after the the DEP switch is pushed, the Program Counter is actually one state higher than that shown in the MEMORY ADDRESS display (if the Rotary Switch is in the MD position).

2.2.6.5 EXAM Switch

Activating this switch, with the Rotary Switch in the MD position, causes the execution of a software routine which displays the current state of the CPU Program Counter in the MEMORY ADDRESS display, then (in the Display lamps) shows the contents of the (data-field) memory location indicated by the MEMORY ADDRESS display, then increments the CPU Program Counter.

This switch is convenient for examining the contents of a series of memory locations. This may be accomplished by loading the first address to be examined into the MEMORY ADDRESS display by using the ADDR LOAD switch, which

will also cause the contents of that (data-field) address to be displayed in the Display lamps (Rotary Switch assumed in the MD position). Activating the EXAM switch once will then cause no change in the displays, but will increment the CPU Program Counter, the "pointer" for the EXAM operation. Each time the EXAM switch is activated thereafter it will cause the address of the next higher memory location to be displayed in the MEMORY ADDRESS lamps and the contents of that location to be shown in the Display lamps.

Note that the two displays now show the address and the contents of one specific (data-field) memory location. The PCM-12 always produces this correspondence between the two displays; there is never a case where the MEMORY ADDRESS display points to one memory location and the Display lamps show the contents of a different location. This correspondence between the two displays differs from PDP-8/E front-panel operation. See Section 2.4.

2.2.6.6 BOOT Switch

This switch activates a selected bootstrap or loader routine, which will bootstrap a mass-storage operating system, or load a paper-tape (or audio-cassette tape) into memory. The selected bootstrap or loader must be set up in switches 10 and 11 of the SWITCH REGISTER before the BOOT switch is depressed, as indicated below:

BOOTSTRAP/LOADER	SW(10)	SW(11)
Paper Tape	0	0
Floppy Disk	0	1
TU58 DECtape II	1	0
RK05 Hard Disk	1	1

The paper-tape loader is identical in operation to DEC's BIN loader for the PDP-8. It will read a binary-format program into memory from an external source such as a low-speed paper-tape reader, high-speed paper-tape reader, audio-cassette recorder, or other compatible paper-tape reader. While the paper-tape loader is running, the MEMORY ADDRESS display shows the address of each main memory location as it is loaded, and the Display lamps show the data loaded into that address. The latter data is displayed regardless of the position of the Rotary Switch. However, at the end of the load the MEMORY ADDRESS display shows the address of the last location loaded, and the Display lamps again follow the Rotary Switch, displaying the contents of the last

location loaded only if the Rotary Switch is in the MD position. If the Rotary Switch is in the AC position (the normal mode of operation), the memory load will end with the CHECKSUM result shown in the Display lamps. If the Rotary Switch is in the STATUS position the load will end with the Status bits shown in the Display lamps, including the last data-field specified by the program loaded. If the Rotary Switch is in the MQ position, the Display lamps will show the contents of the CPU's MQ register. See Section 4.0 for paper-tape load procedure.

Note that when using the paper-tape loader [SW(10) and SW(11) down], SW(0) must be down to use a high-speed reader and up for a low-speed reader.

The three mass-storage bootstraps are used simply by setting up the SWITCH REGISTER appropriately, as indicated above, then activating the RESET switch once, then pushing the BOOT switch followed by the CONT switch (assuming the RUN/HALT and XTAL CLK/SNGL CLK toggle switches are "up"). The mass-storage operating system should then come up and running, as indicated by a monitor character (typically a dot) being typed on the system console.

The bootstrap routine for the floppy disk will boot both single-density (RX01-compatible) and double-density (RX02-compatible) floppy-disk peripherals.

2.3 FRONT PANEL OPERATION WITH EXTENDED MEMORY

When the 12540 Memory Extender is present in the system, the selected Instruction Field (IF) and Data Field (DF) may, of course, differ. The memory data displayed in the Display lamps (when the Rotary Switch is in the MD position) is from the DF if the control-panel interrupt was generated by one of the six Function Switches, and from the IF if the CP interrupt was generated by the TIMER.

2.4 COMPARISON OF PCM-12 AND PDP-8/E FRONT PANELS

For the user who is already familiar with the PDP-8/E, the PCM-12 front-panel functions will seem quite straightforward. The primary differences between the front-panel operations of the two machines are discussed in the following paragraphs.

2.4.1 RESET Switch

This switch on the PCM-12 is similar to the CLEAR switch on the PDP-8/E. However, RESET sets the PCM-12 Program Counter to 7777_8, while CLEAR does not affect the PDP-8/E Program Counter. Therefore, to start a program:

PCM-12	PDP-8/E
1. Lift RESET.	1. Set up starting address.
2. Set up starting address.	2. Depress CLEAR.
3. Push CONT.	3. Depress CONT.

2.4.2 SNGL CLK Switch

This switch (or switch-pair) on the PCM-12 might be compared to the SING STEP switch on the PDP-8/E. However, SNGL CLK advances the PCM-12 by one clock cycle (one-half T-state), while SING STEP advances the PDP-8/E by one full machine cycle.

2.4.3 DEP Switch

The DEP switches on the two machines act very similarly, except in the way memory data is displayed in response to an action by this switch. On the PDP-8/E activation of DEP causes the SWITCH REGISTER data to be transferred to the memory location indicated by the MEMORY ADDRESS display just before DEP was raised, the new memory data to be shown in the Display lamps (assuming the Rotary Switch is in the MD position), and then the Memory Address Register to be incremented and shown in the MEMORY ADDRESS display. Thus, the address and the data shown in the two displays do not correspond; the address shown is one higher than that in which the displayed data resides. In the PCM-12, however, the two displays do correspond; the data shown in the Display lamps resides in in the address shown in the MEMORY ADDRESS display. When depositing data to memory with the Rotary Switch in the MD position, the PCM-12 does increment its pointer after each deposit, but does not display the incremented state.

2.4.4 EXAM Switch

The comments about the manner in which address and memory-data are displayed in response to the DEP switch also apply to the EXAM switch.

2.5 SPECIAL 12530 INSTRUCTIONS

The 12530 control-panel module provides five special (memory-mapped) instructions which the user, at his option, can activate. Three of these instructions are associated with use of the Display lamps for display of user data. The other two instructions are used for selecting whether indirectly-addressed operands will go-to/come-from main memory or control-panel memory. Refer to Table 2-2 and the associated discussion in Section 2.6 below.

For routines residing in control-panel memory, the Display lamps can be used for display of user data by first executing a ENDISP instruction to activate this capability, and then writing data into the lamps with a STROBE instruction. During the time the user is making use of the Display lamps, the TIMER is barred from over-writing the user's displayed data. The only way the displayed data can be lost (while the machine is running), is if the XTAL CLK/SNGL CLK toggle is thrown "down"; in that mode the Display lamps are forced to follow the DX bus state.

To deactivate use of the Display lamps for user data, execute a DISDIS instruction. The TIMER routine then again has access to the Display lamps.

To latch user data into the Display lamps from a program running in main memory, it is first necessary to assert the CPMEM bus line by means of a CPEN (6121_8) instruction, then access control-panel memory location 0172 by means of a DCA I 0172 to enable the Display lamps for display of user data. The data is then strobed into the Display lamps with a DCA I 0170 instruction. Finally, a MEMEN instruction (6122_8) is executed to de-assert the CPMEM bus line. See Section 5.1.2 for discussion of the CPEN and MEMEN instructions on the 12510 CPU module.

The remaining two special instructions (INDCP and INDMN) are intended for use only in routines running in control-panel memory, and are used to give the user the ability to indirectly address locations in control-panel memory. The INDCP and INDMN instructions are operational only with the 12510 CPU module (not the older 12010 CPU). These instructions are used to determine whether operands of indirect AND, TAD, ISZ and DCA instructions will come-from/go-to main memory or control-panel memory. The normal (default) mode for the machine is that all "indirects" come-from/go-to main memory. However, if a INDCP instruction is executed, the CPMEM bus line is asserted low and all subsequent indirects will come-from/go-to control-panel memory

until a INDMN or CAF instruction is executed. This, for example, enables the user to load a portion of control-panel RAM with an auto-indexed DCA I loop. Note that indirect JMS and JMP instructions aren't affected by the INDCP/INDMN logic; indirect JMS's and JMP's always go to control-panel memory if they were fetched from control-panel memory and to main memory if they were fetched from main memory (the one exception, of course, is the JMP I ZERO which is executed at exit from control-panel memory - this instruction, fetched from control-panel memory, always causes a jump to main memory).

2.6 12530 LOGIC OPERATION

A schematic diagram for the logic on the 12530 front-panel module is given in Drawing 12532. This drawing should be reviewed in relation to the the front-panel software routine given in Section 2.7, since most front-panel operations are closely inter-related with the software resident in the PROM's (U21, U22 and U23) on the 12530 module. Reference should also be made to Section 5.6 of this manual, which discusses the machine's provisions for control-panel interrupts. (In fact, a review of all the material in Section 5.0 is recommended before attempting to understand the detailed circuitry on each printed-circuit module in the PCM-12 microcomputer.)

Electrical communication between the 12530 module and the PCM-12 bus is accomplished via a 50-conductor ribbon-cable. This ribbon-cable terminates at P1 on the 12530 printed-circuit module. The pin designations for P1 are given in Table 2-1 below:

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
1	GND	14	DX9	27	GND	40	CPREQ L
2	RUN L	15	DX8	28	AUX L	41	GND
3	+5V	16	DX7	29	GND	42	SWSEL L
4	LINK L	17	DX6	30	CPSEL L	43	GND
5	+5V	18	DX5	31	GND	44	FREERUN H
6	RESET L	19	DX4	32	IFETCH H	45	GND
7	+5V	20	DX3	33	GND	46	DMAGNT H
8	RUN/HALT L	21	DX2	34	LXMAR H	47	GND
9	+5V	22	DX1	35	GND	48	INTGNT H
10	SNGL CLK H	23	DX0	36	XTA H	49	GND
11	GND	24	CPMEM L	37	GND	50	GND
12	DX11	25	GND	38	XTC H		
13	DX10	26	SW L	39	GND		

TABLE 2-1

Pin Assignments for Front-Panel Ribbon-Cable Connector, P1

Each address and each instruction is latched into the hex-D flip-flops U5 and U6 by the trailing edge of the LXMAR pulse acting through the inverter U31C. The decoder devices U14, U17, U18 and U19, acting in conjunction with the gates U15A, U16B, U15B and U15D, are used to decode each address or instruction to determine if it is intended for the control-panel logic, and if so what action is intended.

All the software-driven functions, switches and indicators on the 12530 module are memory-mapped; that is, they are stimulated by addressing specific addresses (0170_8-0177_8) in the control-panel memory space. The eight control-panel memory locations from address 0170 through 0177 have been reserved for this purpose. The function-switch array is read with a FNSW (TAD 0173) instruction. All the other functions are stimulated with a DCA to the appropriate address, leaving the Accumulator in the clear state. The instruction set for the 12530 module is shown in Table 2-2, below:

MNEMONIC	OCTAL	OPERATION
STROBE	3170	Strobe user data into Display lamps.
ENDISP	3171	Enable user access to the Display lamps.
DISDIS	3172	Disable user access to the Display lamps.
FNSW	1173	Add the Function-Switch array to the Accumulator.
MASEL	3174	Load Accumulator contents into the MEMORY ADDRESS lamps.
DISPLA	3175	Load Accumulator contents into the Display lamps.
INDCP	3176	Indirects to control-panel memory.
INDMN	3177	Indirects to main memory (default mode).

TABLE 2-2

Instruction set for 12530 Control-Panel Module

Any of the six Function Switches (EXAMine, DECRement ADDRess, BOOT, ADDRess LOAD, EXTendeD address LOAD and DEPosit) can cause a control-panel (CP) interrupt by pulling an input low on U40 when the CPU is halted. Whenever this happens, the R/S flip-flop formed by U40 and U41D is set, asserting pin 6 of U34C high. This assertion is passed through U27A and U42D to cause the CPREQ bus line (line 66) to be asserted low, thus creating a CP interrupt. By executing a FNSW instruction (TAD 0173) in the control-panel routine, it is then possible to read the Function Switch array through U39 to discover which of the six possible Function Switches caused the CP interrupt.

When the FNSW instruction is executed, pin 12 of U18 is asserted low at CPSEL time, which enables the Function Switch array through inverter U38A and gate U16D. Simultaneously, the drivers U39 and U28 are enabled to drive the Function Switch array data onto the 12530 module's internal DX bus, and the bus drivers U3 and U4 are enabled by gate U36B and inverter U31D to drive this data onto the PCM-12 bus. The R/S flip-flop formed by U40 and U41D gets reset as a result of the ION (6001_8) instruction which is executed at the end of the CP routine. Execution of the ION causes pin 14 of U19 to be asserted low during the ION's IOTA cycle. This sets flip-flop U37A. Then during the next IFETCH cycle, flip-flop U43B is clocked set by the CPSEL signal, which resets both the R/S flip-flop and flip-flop U37A. U43B is reset during the "indirect" cycle of the JMP I ZERO instruction which follows the ION.

Memory addresses are strobed into the MEMORY ADDRESS lamps through the quad-D flip-flops U7, U9 and U11 by executing a MASEL (DCA 0174) instruction in the control-panel routine. Execution of this instruction causes pin 11 of U18 to be asserted at CPSEL time. This signal is passed through gate U13B and inverter U34A to clock the data into U7, U9 and U11.

Memory Data, the Accumulator contents, MQ Register contents or status information is latched into the Display lamps by executing a DISPLA (DCA 0175) instruction. When this instruction is executed, pin 10 of U18 is asserted at CPSEL time, and this signal is passed through inverter U38F and gates U13D and U13A to latch the appropriate data into the quad-latch devices U8, U10 and U12.

A capability has been designed into the PCM-12 Omega front-panel logic so that the Display lamps can be used to display user-programmed data when the machine is running. To activate this feature, it is necessary to first access control-panel location 0171 to enable the "Data Display". Then location 0170 must be written into to strobe data into the Display lamps. When location 0171 is accessed, pin 14 of U18 is asserted low, which causes flip-flop U30C to go to the set state. With U30C set, gate U13D is disabled and so is the upper input on gate U29A. Thus, all normal functions of the Display lamps are disabled, and the lamps can display data strobed into them through pin 9 of gate U13C by writing into CP location 0170. The Data Display is disabled, and all normal functions of the Display lamps restored, by accessing CP location 0172. This asserts pin 13 of U18, which directly resets flip-flop U30C.

Whenever an indirect AND, ISZ, TAD or DCA is executed from control-panel memory, the operand normally goes-to or comes-from main memory. Logic has

been designed into the 12530 module, however, so that the operand can come-from/go-to either main or CP memory, at the programmer's discretion. The default mode is the normal IM6100 mode; that is, the operands come-from/go-to main memory. However, if a INDCP instruction is executed, flip-flop U20B is taken to the set state by the assertion of pin 9 on U18. This causes the CPMEM bus line (line 47) to be asserted low, which causes all indirect operands to come-from/go-to control-panel memory. When a INDMN instruction is executed, pin 7 of U18 gets asserted at CPSEL time, the flip-flop U20B gets reset again, and indirects return to the default state, i.e., they come-from/go-to main memory.

2.6.1 REAL-TIME DISPLAYS

During each main-memory IFETCH cycle (with TIMER off), the address of the current instruction is latched into the MEMORY ADDRESS lamps, and the current instruction itself is latched into the Display lamps. The LXMAR pulse, acting through inverter U34B, gate U41C, U34F gates U27B and U13B and inverter U34A is used to latch the instruction address into the flip-flops U7, U9 and U11. The XTA pulse at pin 9 of U29A, acting through gate U13A, is used to latch the instruction itself into U8, U10 and U12. When the Data Display is active (flip-flop U30C in set state), the real-time display of instructions is inhibited, so as not to disturb the user data shown in the Display lamps. Real-time display of the instruction addresses in the MEMORY ADDRESS lamps is continued, however.

2.6.2 TIMER OPERATION

When the TIMER switch is in the "up" position, the timer device U44 causes flip-flop U43A to get set several times each second. Each time U43A gets set it causes a CP interrupt request by asserting the CPREQ bus line through pin 9 of U27A and the buffer U42D. The main purpose of the TIMER capability is to allow the MEMORY ADDRESS and Display lamps to be refreshed with the appropriate data, as explained in Section 2.2.5, above. Note that when flip-flop U30C is in the set state, the TIMER is unable to refresh the Display lamps, since gate U13D is disabled.

Note that the RUN/HALT signal is tied to pin 10 of U27A. This guarantees that in the single-instruction mode, every activation of the SINGL INST push-button will cause an instruction to be fetched and executed. An

asynchronous timer-generated CP request cannot block out a single-instruction request by forcing the machine into the RUN mode while the SNGL INST switch is momentarily depressed.

Also note that the gate U16A inhibits TIMER-generated CP requests during the time that either the INTGNT or DMAGNT bus lines is asserted high. A CP interrupt would be inappropriate during a DMA operation, since the whole purpose of DMA operations is to effect a data transfer in the least time possible. CP interrupts are also barred during the time that INTGNT is asserted, because any IOT instruction executed during the CP routine would cause INTGNT to be reset to the low (unasserted) state before the main-memory program was ready for this to occur.

2.7 CONTROL PANEL SOFTWARE ROUTINE

The assembly listing on the following pages gives the current control-panel software routine for the PCM-12 Omega. The designation for this program is FP530.PA. The program is heavily commented and therefore should be fairly easy to analyze. The binary loader routine (beginning at address 7600), though, contains many switches and jumps; flowcharting is recommended if modifications to this routine are ever contemplated.

The entire PROM memory space is four pages in length, extending from address 7000_8 to address 7777_8. The front-panel service operations are mainly contained in the address space from 7000 to 7262. The BIN loader begins at address 7600, the floppy boot routine at 7400, the hard disk boot at 7521 and the TU58 boot at 7263. There still remains a small amount of (unprogrammed) space left in the PROM's that the user may wish to make use of at a later date (for example, to support a custom power-fail and re-start function).

The control-panel software routine begins at location 7777 with an immediate jump through 7776 to 7000 where the actual operations begin. The program ends at location 7262 with an indirect jump through CP location 0000, which contains the return address to the main-memory program.

NOTE: The bootstrap routine for the TU-58 device is not included in the current version of this program. When this routine is ready it will be made available at no charge. Un-programmed space in these PROM's can then be used to implement the new bootstrap.

```
                /12530 FRONT-PANEL LISTING - VERSION 1.  25-MAY-80.

                   /PROMS ARE MARKED WITH BROWN DOT, AND AS FOLLOWS:
                   /U21:  12530-U21   (THESE ARE THE MSB'S)
                   /U22:  12530-U22   (THESE ARE THE MIDDLE BITS)
                   /U23:  12530-U23   (THESE ARE THE LSB'S)

          3170          STROBE=3170
          3171          ENDISP=3171
          3172          DISDIS=3172
          1173          FNSW=1173
          3174          MASEL=3174
          3175          DISPLA=3175
          3176          INDCP=3176
          3177          INDMN=3177

                        /RAM LOCATIONS
          0000          *0000
00000     0000  PC,     0000     /HOLDS RETURN ADDRESS.
00001     0000  AC,     0000     /HOLDS STATE OF ACCUMULATOR.
00002     0000  FLAGS,  0000     /HOLDS STATUS BITS.
00003     0000  THREE,  0000     /HOLDS FUNCTION SWITCH ARRAY.

00004     0000          0000
00005     0000  EXEC,   0000     /SUBROUTINE FOR CHANGING FIELDS.
00006     0000          0000

00007     0000  RDRSEL, 0000     /THIS LOCATION, AND NEXT 8, ARE
00010     0000  BEGSW,  0000     /USED BY BOOT ROUTINES.
00011     0000  RUBSW,  0000  /CHAR IS USED BY BIN BOOT ROUTINE, AND
00012     0000  CHAR,   0000  /AS AUTO-INDEXING POINTER BY FLOPPY
00013     0000  RDRSW,  0000  /BOOT ROUTINE.  ADRINC IS USED BY F-P
00014     0000  WORD1,  0000  /ROUTINE FOR ADDRESS INCREMENTING, AND
00015     0000  WORD2,  0000  /ALSO BY BIN BOOT ROUTINE.  IT MUST BE
00016     0000  CHKSUM, 0000  /CLEAR WHEN ENTERING & LEAVING THE
00017     0000  ADRINC, 0000  /BIN BOOT ROUTINE.

                        /PROM PROGRAM
          7000          *7000
07000     3001  START,  DCA AC  /SAVE THE ACCUM (MQ WON'T BE USED).
07001     6004          GTF     /FETCH THE STATUS BITS.
07002     3002          DCA FLAGS  /AND SAVE THEM.
07003     1173          FNSW    /FETCH THE FUNCTION SWITCH ARRAY.
07004     7040          CMA     /SO WE CAN USE 'OR' AFTER RAL.
07005     7004          RAL
07006     7520          SMA SNL  /SKIP IF EITHER TIMER OR FUNCTION
07007     5777'         JMP OTHERS /SWITCH CAUSED INT. - OTHERWISE JUMP.
07010     7630          SZL CLA /SKIP IF TIMER DIDN'T CAUSE INTERRUPT.
07011     5227          JMP XX+2  /JUMP IF TIMER CAUSED INTERRUPT.
07012     1376          TAD (7300
07013     3004          DCA EXEC-1 /SET UP SWITCH-INTERROGATION COUNTER.
```



```
07014     1173  SW,     FNSW  /FETCH THE FUNCTION SWITCH ARRAY.
07015     3003          DCA THREE  /AND SAVE IT.
07016     1003          TAD THREE  /FETCH IT BACK.
07017     0375          AND (1760  /MASK OUT ALL BUT FUNCTION SWITCHES.
07020     7640          SZA CLA    /SKIP IF NO SWITCH CLOSED.
07021     5225          JMP XX  /JUMP IF A SWITCH IS CLOSED.
07022     2004          ISZ EXEC-1  /SHALL WE CHECK SWITCHES AGAIN?
07023     5214          JMP SW    /YES, SO JUMP BACK AND DO IT.
07024     5774'         JMP OUT-1 /NO, THAT'S ENOUGH - GIVE UP.

07025     3172  XX,     DISDIS  /KILL MAIN MEM ACCESS TO DISPLAY.
07026     7410          SKP
07027     3003          DCA THREE  /TO INDICATE TIMER CAUSED INTERRUPT.
07030     3177          INDMN    /INDIRECTS WILL GO TO MAIN MEMORY.
07031     3017          DCA ADRINC  /INIT THE ADDRESS INCREMENTER.
07032     1002          TAD FLAGS  /FETCH THE FLAG BITS BACK.
07033     0373          AND (7700   /AND MASK OUT BITS 6-11.
07034     3002          DCA FLAGS  /AND SAVE BITS 0-5.
07035     6214          RDF        /READ THE DATA-FIELD REGISTER.
07036     7110          CLL RAR
07037     7012          RTR
07040     6224          RIF      /'OR' IN THE INSTRUCTION-FIELD REGISTER.
07041     1002          TAD FLAGS  /ADD IN SAVED STATUS BITS 0-5.
07042     3002          DCA FLAGS /SAVED STATUS BITS NOW INCLUDE IF & DF.
07043     1372          TAD (JMP I EXEC-1  /GET RETURN INST FOR SUBROUTINE.
07044     3006          DCA EXEC+1    /PUT IT INTO SUBROUTINE'S 1ST ADDR.
07045     1003          TAD THREE  /FETCH THE SAVED FUNC SW ARRAY.
07046     7004          RAL
07047     7420          SNL  /SKIP IF TIMER DIDN'T CAUSE INTERRUPT.
07050     5346          JMP TIMER  /JUMP IF TIMER CAUSED INTERRUPT.
07051     7006          RTL
07052     7430          SZL  /SKIP IF 'EXAM' DIDN'T CAUSE INT.
07053     5343          JMP EXAM  /IF 'EXAM' DID CAUSE INT.
07054     7510          SPA  /SKIP IF 'ADDR LOAD' DIDN'T CAUSE INT.
07055     5272          JMP ADDRLD  /IF 'ADDR LOAD' DID CAUSE INT.
07056     7006          RTL
07057     7430          SZL  /SKIP IF 'EXTD ADDR' DIDN'T CAUSE INT.
07060     5307          JMP EXADDR  /IF 'EXTD ADDR' DID CAUSE INT.
07061     7510          SPA  /SKIP IF 'DEP' DIDN'T CAUSE INT.
07062     5341          JMP DEP  /IF 'DEP' DID CAUSE INT.
07063     7006          RTL
07064     7430          SZL  /SKIP IF 'BOOT' DIDN'T CAUSE INT.
07065     5275          JMP BOOT  /IF 'BOOT' DID CAUSE INT.

07066     7240  DECADR, STA  /MUST HAVE BEEN 'DECR ADDR', IF NO OTHER.
07067     1000          TAD PC  /DECREMENT THE MEMORY-ADDRESS IN 'PC'.
07070     3000          DCA PC
07071     5353          JMP BUTTON

07072     7604  ADDRLD, CLA OSR  /FETCH THE SWITCH REGISTER,
07073     3000          DCA PC   /AND PUT IT INTO 'PC'.
07074     5353          JMP BUTTON
```

```
07075  7604  BOOT,   CLA OSR  /FETCH THE SWITCH REGISTER.
07076  7012           RTR
07077  7430           SZL      /SKIP IF BIN OR FLOPPY BOOT DESIRED.
07100  5304           JMP .+4  /JUMP IF TU58 OR RK05 BOOT DESIRED.
07101  7510           SPA      /SKIP IF BIN BOOT.
07102  5771'          JMP FLOP
07103  5770'          JMP BIN
07104  7710           SPA CLA  /SKIP IF TU-58.
07105  5767'          JMP RK05
07106  5766'          JMP TU58

07107  7200  EXADDR, CLA
07110  1002           TAD FLAGS  /FETCH SAVED STATUS BITS.
07111  0376           AND (7300 /MASK OUT IIFF, IF AND DF.
07112  3002           DCA FLAGS
07113  7404           OSR        /FETCH THE SWITCH REGISTER.
07114  0365           AND (0077  /MASK OUT BITS 0-5.
07115  1002           TAD FLAGS  /ADD IN SAVED BITS 0-5.
07116  3002           DCA FLAGS  /SAVE THE NEW STATUS BIT ARRAY.
07117  1002           TAD FLAGS  /FETCH THEM BACK.
07120  0364           AND (0070  /MASK OUT ALL BUT BITS 6-8.
07121  1363           TAD (6202 /FORM 62N2 (CIF) INSTRUCTION.
07122  3005           DCA EXEC   /FORM CIF SUBROUTINE.
07123  4004           JMS EXEC-1  /EXECUTE THE SUBROUTINE.
07124  6254           6254       /IM6102 LIF (IB TO IF) INSTRUCTION.
07125  1005           TAD EXEC
07126  1362           TAD (0005  /FORM 62N7 INSTRUCTION.
07127  3005           DCA EXEC     /FORM 62N7 SUBROUTINE.
07130  4004           JMS EXEC-1   /AND EXECUTE IT (FOR 12040 MODULE).
07131  1002           TAD FLAGS  /FETCH THE SAVED STATUS BITS.
07132  7004           RAL
07133  7006           RTL
07134  0364           AND (0070  /MASK OUT ALL BUT BITS 6-8.
07135  1361           TAD (6201  /FORM 62N1 (CDF) INSTRUCTION.
07136  3005           DCA EXEC   /FORM CDF SUBROUTINE.
07137  4004           JMS EXEC-1   /AND EXECUTE IT.
07140  5353           JMP BUTTON

07141  7604  DEP,    CLA OSR    /FETCH THE SWITCH REGISTER.
07142  3400           DCA I PC   /LOAD THE DATA INTO DATA FIELD ADDR
                                 /INDICATED BY POINTER IN 'PC'.
07143  7301  EXAM,   CLA CLL IAC  /GET 0001,
07144  3017           DCA ADRINC   /AND PUT IT IN ADRINC.
07145  5353           JMP BUTTON

07146  7300  TIMER,  CLA CLL
07147  6224           RIF      /FETCH THE IF REGISTER.
07150  1361           TAD (6201 /FORM 62N1 (CDF) INSTRUCTION.
07151  3005           DCA EXEC  /FORM CDF SUBROUTINE.
07152  4004           JMS EXEC-1  /MAKE DF=IF SINCE TIMER CAUSED INTERRUPT.
07153  1000  BUTTON, TAD PC   /FETCH STORED MEMORY ADDRESS.
07154  3174           MASEL /AND DISPLAY IT.
07155  5760'          JMP ROTSW   /NEXT PAGE.
```



```
07160  7200
07161  6201
07162  0005
07163  6202
07164  0070
07165  0077
07166  7263
07167  7521
07170  7600
07171  7400
07172  5404
07173  7700
07174  7254
07175  1760
07176  7300
07177  7264
       7200           *7200
07200  1173  ROTSW,  FNSW     /FETCH ROTARY SWITCH.
07201  7012           RTR
07202  7510           SPA  /SKIP IF ROTARY SWITCH NOT IN 'STATUS' POSITION.
07203  5212           JMP STATUS  /IF ROT SW IN 'STATUS' POSITION.
07204  7430           SZL  /SKIP IF ROT SW NOT IN 'MQ' POSITION.
07205  5215           JMP MQ  /IF ROT SW IN 'MQ' POSITION.
07206  7010           RAR
07207  7630           SZL CLA /SKIP IF ROT SW NOT IN 'AC' POSITION.
07210  5217           JMP ACC /IF ROT SW IN 'AC' POSITION.
07211  5221           JMP MD  /MUST BE IN 'MD' POSITION IF NO OTHER.

07212  7300  STATUS, CLA CLL
07213  1002           TAD FLAGS  /FETCH THE STORED STATUS BITS.
07214  5230           JMP EXIT

07215  7701  MQ,     CLA MQA   /FETCH THE MQ REGISTER CONTENTS.
07216  5230           JMP EXIT

07217  1001  ACC,    TAD AC    /FETCH THE SAVED ACCUMULATOR.
07220  5230           JMP EXIT

07221  1400  MD,     TAD I PC   /FETCH DATA-FIELD DATA POINTED TO
                                /BY ADDRESS STORED IN 'PC'.
07222  3175           DISPLA      /AND DISPLAY IT.
07223  1000           TAD PC
07224  1017           TAD ADRINC  /INCREMENTS POINTER FOR 'EXAM' & 'DEP'.
07225  3000           DCA PC
07226  7000           NOP  /ALL TIMER PATHS MUST HAVE
07227  5231           JMP EXIT+1  /AN EVEN NUMBER OF INSTRUCTIONS.

07230  3175  EXIT,   DISPLA     /DISPLAY THE SELECTED INFORMATION.
07231  1003           TAD THREE  /FETCH THE SAVED FUNCTION SW ARRAY.
07232  7710           SPA CLA    /SKIP IF TIMER CAUSED C-P INTERRUPT.
07233  5244           JMP FUNSW  /JUMP IF ANY FUNC SW CAUSED INT.
07234  1002           TAD FLAGS  /FETCH STORED STATUS BITS.
07235  7004           RAL
07236  7006           RTL
07237  0377           AND (0070  /MASK OUT ALL BUT BITS 6-8.
```

```
07240  1376          TAD (6201  /FORM 62N1 (CDF) INSTRUCTION,
07241  3005          DCA EXEC   /FORM CDF SUBROUTINE.
07242  4004          JMS EXEC-1  /RESTORE DF REGISTER SETTING.
07243  5255          JMP OUT

07244  1375  FUNSW,  TAD (1760 /SET UP FOR 100 MILLISECOND SWITCH
07245  3004          DCA EXEC-1  /DEBOUNCE (AT 4 MHZ).
07246  1173          FNSW     /FETCH THE FUNCTION SWITCH ARRAY.
07247  0375          AND (1760 /MASK OUT ALL BUT FUNCTION SWITCHES.
07250  7640          SZA CLA   /SKIP IF NO FUNCTION SWITCH CLOSED.
07251  5244          JMP .-5   /GO BACK IF SOME SWITCH STILL CLOSED.
07252  2004          ISZ EXEC-1  /INCREMENT THE DEBOUNCE COUNTER.
07253  5246          JMP .-5  /GO BACK IF DEBOUNCE DELAY NOT OVER.
07254  7402          HLT  /FUNC SW INT MUST END WITH MACHINE HALTED.
07255  1002  OUT,    TAD FLAGS
07256  7004          RAL  /RESTORE THE LINK.
07257  7200          CLA
07260  1001          TAD AC  /RESTORE THE ACCUMULATOR.
07261  6001          ION    /DUMMY ION TO GET OUT OF C-P MEMORY.
07262  5400          JMP I PC  /RETURN TO MAIN MEMORY.

07263  0000  TU58,   0000      /TU-58 BOOTSTRAP ROUTINE GOES HERE.
                               /(THE 0000 JUST HOLDS THIS LOCATION
                               /UNPROGRAMMED.)

07264  0000  OTHERS, 0000      /USE THE REST OF THIS PAGE FOR OTHER
                               /ROUTINES SUCH AS NEW BOOTSTRAPS, OR
                               /POWER-FAIL/START-UP ROUTINES.  (THE
                               /0000 HOLDS THIS LOCATION UNPROGRAMMED.)

07375  1760
07376  6201
07377  0070
       7400          *7400
07400  6201  FLOP,   6201          /BEGIN FLOPPY BOOT ROUTINE.
07401  7300          CLA CLL
07402  1377          TAD (17       /THIS ROUTINE WILL BOOTSTRAP BOTH
07403  3012          DCA CHAR      /RX01 AND RX02 FLOPPY SYSTEMS, WITH
07404  1376          TAD (1061     /THE SYSTEM DISKETTE IN EITHER DRIVE.
07405  3412          DCA I CHAR
07406  1375          TAD (1046
07407  3412          DCA I CHAR
07410  1374          TAD (0060
07411  3412          DCA I CHAR
07412  1373          TAD (3061
07413  3412          DCA I CHAR
07414  1372          TAD (7327
07415  3412          DCA I CHAR
07416  1376          TAD (1061
07417  3412          DCA I CHAR
07420  1371          TAD (6751
07421  3412          DCA I CHAR
07422  1370          TAD (7301
07423  3412          DCA I CHAR
07424  1367          TAD (4053
```



```
07425  3412          DCA I CHAR
07426  1367          TAD (4053
07427  3412          DCA I CHAR
07430  1366          TAD (7004
07431  3412          DCA I CHAR
07432  1365          TAD (6755
07433  3412          DCA I CHAR
07434  1364          TAD (5054
07435  3412          DCA I CHAR
07436  1363          TAD (6754
07437  3412          DCA I CHAR
07440  1362          TAD (7450
07441  3412          DCA I CHAR
07442  1361          TAD (5020
07443  3412          DCA I CHAR
07444  1376          TAD (1061
07445  3412          DCA I CHAR
07446  1371          TAD (6751
07447  3412          DCA I CHAR
07450  1376          TAD (1061
07451  3412          DCA I CHAR
07452  1360          TAD (0046
07453  3412          DCA I CHAR
07454  1357          TAD (1032
07455  3412          DCA I CHAR
07456  1356          TAD (3060
07457  3412          DCA I CHAR
07460  1355          TAD (0360
07461  3412          DCA I CHAR
07462  1367          TAD (4053
07463  3412          DCA I CHAR
07464  1354          TAD (3002
07465  3412          DCA I CHAR
07466  1353          TAD (2050
07467  3412          DCA I CHAR
07470  1352          TAD (5047
07471  3412          DCA I CHAR   /5047 GOES INTO LOCATION 52.
07472  3412          DCA I CHAR   /CLEAR LOCATION 53.
07473  1351          TAD (6753
07474  3412          DCA I CHAR
07475  1350          TAD (5033
07476  3412          DCA I CHAR
07477  1347          TAD (6752
07500  3412          DCA I CHAR
07501  1346          TAD (5453
07502  3412          DCA I CHAR
07503  1345          TAD (0420
07504  3412          DCA I CHAR
07505  1344          TAD (0020
07506  3412          DCA I CHAR
07507  1343          TAD (33
07510  3000          DCA PC       /SET UP STARTING ADDRESS.
```

```
07511  6202  DONE,   6202          /CIF 00.
07512  6254          6254          /IM6102 LIF (IB TO IF).
07513  6207          6207          /FOR 12040 MEMORY EXTENDER.
07514  3001          DCA AC        /START WITH CLEAR ACCUMULATOR.
07515  1002          TAD FLAGS
07516  0342          AND (3300     /TO SET INDICATED IF AND DF TO
07517  3002          DCA FLAGS     /ZERO AND CLEAR LINK AT EXIT.
07520  5741'         JMP BUTTON


07521  6201  RK05,   6201
07522  1340          TAD (6743
07523  3737          DCA I (30
07524  1336          TAD (JMP 31
07525  3735          DCA I (31
07526  1337          TAD (30
07527  3000          DCA PC        /SET UP STARTING ADDRESS.
07530  5311          JMP DONE


07535  0031
07536  5031
07537  0030
07540  6743
07541  7153
07542  3300
07543  0033
07544  0020
07545  0420
07546  5453
07547  6752
07550  5033
07551  6753
07552  5047
07553  2050
07554  3002
07555  0360
07556  3060
07557  1032
07560  0046
07561  5020
07562  7450
07563  6754
07564  5054
07565  6755
07566  7004
07567  4053
07570  7301
07571  6751
07572  7327
07573  3061
07574  0060
07575  1046
07576  1061
07577  0017
```



```
       7600           *7600
07600  6032  BIN,    6032  /INITIALIZE LO-SPD READER.
07601  6014          6014  /SAME FOR HI-SPD RDR.
07602  7200          CLA
07603  6214          RDF  /FETCH THE DF.
07604  1366          TAD C6201  /FORM 62N1 INST (CDF).
07605  3005          DCA EXEC  /FORM CDF SUB-ROUTINE.
07606  7404          OSR      /FETCH THE RDR SELECT CODE.
07607  3007          DCA RDRSEL  /SAVE RDR SELECT CODE.
07610  7040          CMA   /7777 INTO BEGSW WILL INDICATE (AT GO-2)
07611  3010  XBEGG,  DCA BEGSW  /THAT CHKSUM IS TO BE CLEARED.
07612  3011  BEGG,   DCA RUBSW  /0000 INTO RUBSW INDICATES ENTRY
07613  7040          CMA   /FROM XBEGG OR YBEGG, 7777 FROM ZBEGG.
07614  3013          DCA RDRSW  /7777 INTO RDRSW WILL INDICATE (AT
07615  1007  READ,   TAD RDRSEL /SAVE+2) THAT CHAR IS NOT WORD2 (2ND
07616  7700  M7700,  SMA CLA  /6 BITS OF DATA WORD OR ORIGIN).
07617  5224          JMP HI   /USE HIGH-SPEED READER.
```

```
07620  6031  LO,     6031
07621  5220          JMP LO
07622  6036          6036
07623  5227          JMP SAVE
07624  6011  HI,     6011
07625  5224          JMP HI
07626  6016          6016
07627  3012  SAVE,   DCA CHAR  /STORE DATA IN CHAR.
07630  1012          TAD CHAR
07631  2013          ISZ RDRSW /DON'T SKIP IF DATA IS WORD2 (SO
07632  5276          JMP GO+5   /CAN'T BE F/S, ORIGIN OR L/T).
07633  1367          TAD K7402
07634  7750          SPA SNA CLA  /SKIP IF CHAR IS ALL ONES.
07635  5241          JMP NORUB
07636  2011  RUB,    ISZ RUBSW /SKIP IF THIS IS THE 2ND RUBOUT CHAR,
07637  7040          CMA        /OTHERWISE PUT 7777 INTO RUBSW.
07640  5212  ZBEGG,  JMP BEGG
07641  1011  NORUB,  TAD RUBSW
07642  7640          SZA CLA  /SKIP IF NOT PARTWAY BETWEEN
07643  5213          JMP BEGG+1  /TWO RUBOUTS.
07644  1012          TAD CHAR
07645  0364          AND M0300
07646  1325          TAD K7600
07647  7510          SPA   /SKIP IF CH-8 IS PUNCHED.
07650  5266          JMP DAORG /JUMP IF NOT (CHAR IS DATA OR ORIGIN).
07651  7750          SPA SNA CLA  /SKIP IF CH 7 & 8 ARE BOTH PUNCHED.
07652  5263          JMP LT   /JUMP IF ONLY CH-8 IS PUNCHED.
07653  1012  FS,     TAD CHAR
07654  0363          AND M0070
07655  1366          TAD C6201   /FORM 62N1 (CDF) INSTRUCTION.
07656  3005          DCA EXEC
07657  1017          TAD ADRINC
07660  7650          SNA CLA /SKIP IF CDF EXECUTE MUST BE DELAYED UN-
07661  4004          JMS EXEC-1 /TIL AFTER NEXT DATA DEP TO MAIN MEM.
07662  5213          JMP BEGG+1  /GO BACK AND FETCH NEXT CHARACTER.
07663  2010  LT,     ISZ BEGSW /SKIP IF LEADER CODE, NO SKP IF TRLR.
07664  5335          JMP END
07665  5210          JMP BEGG-2
07666  7200  DAORG,  CLA        /CHARACTER IS DATA OR ORIGIN-SETTING.
07667  2010          ISZ BEGSW  /SKIP IF THIS IS 1ST DATA OR
07670  5307          JMP DEPOST  /ORIGIN CHARACTER ON TAPE.
07671  3016  GO,     DCA CHKSUM
07672  1012          TAD CHAR   /BEGIN ASSEMBLY OF NEXT WORD.
07673  3014          DCA WORD1
07674  3013          DCA RDRSW  /CLEAR RDRSW INDICATES NEXT CHARACTER
07675  5215          JMP READ  /TO BE FETCHED WILL BE WORD2.
07676  3015          DCA WORD2 /SAVE CHARACTER.
07677  1014          TAD WORD1 /SEE IF WORD1 IS PART OF ORIGIN WORD.
07700  7102          CLL BSW
07701  7010          RAR
07702  7620          SNL CLA  /SKIP IF ORIGIN-SETTING.
07703  7240          STA
07704  3017          DCA ADRINC
07705  3010          DCA BEGSW /CLEAR BEGSW WILL INDICATE (AT GO-2)
07706  5212  YBEGG,  JMP BEGG  /THAT CHKSUM SHOULDN'T BE CLEARED.
```



```
07707  1014  DEPOST, TAD WORD1  /ASSEMBLE 12-BIT WORD.
07710  7106          CLL RTL
07711  7006          RTL
07712  7006          RTL
07713  1015          TAD WORD2
07714  7430          SZL  /SKIP IF CH-7 OF WORD1 WASN'T PUNCHED.
07715  5330          JMP ORIGIN
07716  3400          DCA I PC  /12-BIT WORD IS DATA; DEP TO MAIN MEM.
07717  1000          TAD PC
07720  3174          MASEL      /DISPLAY ADDRESS JUST DEPOSITED TO.
07721  1400          TAD I PC
07722  3175          DISPLA     /DISPLAY DATA JUST DEPOSITED.
07723  4004          JMS EXEC-1  /EXECUTE THE CDF SUBROUTINE.
07724  2000          ISZ PC   /INCREMENT THE POINTER.
07725  7600  K7600,  7600  /GROUP 2 CLA.
07726  7200          CLA
07727  7410          SKP
07730  3000  ORIGIN, DCA PC   /SET UP NEW POINTER.
07731  1014  CHEX,   TAD WORD1   /CHECKSUM ACCUMULATION ROUTINE.
07732  1015          TAD WORD2
07733  1016          TAD CHKSUM
07734  5271          JMP GO
07735  1014  END,    TAD WORD1   /CHECK CHECKSUM.
07736  0365          AND M0077  /MASK OUT POSSIBLE CHNL 7 PUNCH.
07737  7002          BSW
07740  1015          TAD WORD2
07741  7041          CIA
07742  1016          TAD CHKSUM  /SHOULD MAKE ZERO.
07743  3001          DCA AC  /FOR DISPLAY IF ROT SW IN AC POSITION.
07744  3017          DCA ADRINC /ADRINC IS USED BY PANEL ROUTINE ALSO.
07745  6004          GTF       /ROUTINE FOR DISPLAY OF NEW
07746  0216          AND M7700   /STATUS AND FIELD BITS.
07747  3002          DCA FLAGS
07750  6214          RDF
07751  7110          CLL RAR
07752  7012          RTR
07753  6224          RIF
07754  1002          TAD FLAGS
07755  3002          DCA FLAGS
07756  7240          STA
07757  1000          TAD PC
07760  3000          DCA PC  /TO DECREMENT POINTER SO LAST ADDR
07761  5762          JMP I XBUTTN  /DEPOSITED TO IS SHOWN AT EXIT.
07762  7153  XBUTTN, BUTTON
07763  0070  M0070,  0070
07764  0300  M0300,  0300
07765  0077  M0077,  0077
07766  6201  C6201,  6201
07767  7402  K7402,  7402

       7776          *7776
07776  7000          START
07777  5776          JMP I 7776
                     $
```

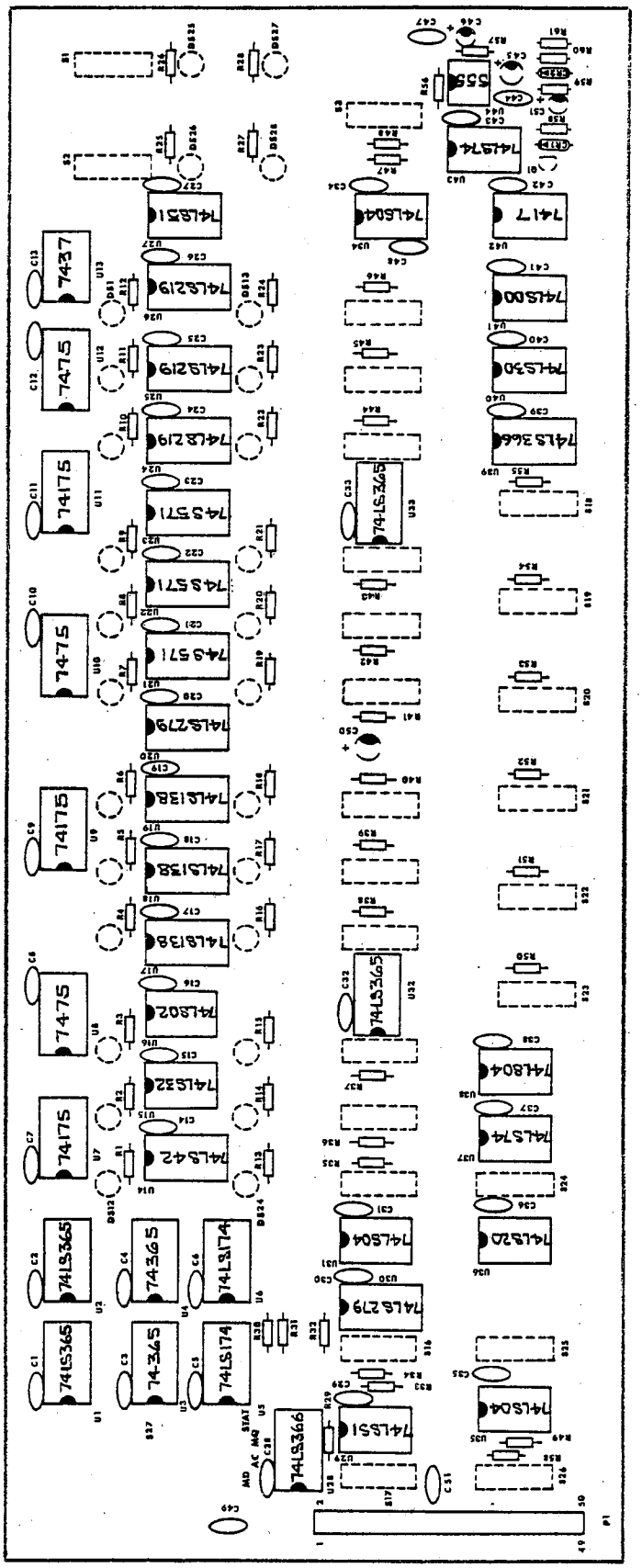


FIGURE 2-1
12530 Printed-Circuit Board Layout

SECTION 3.0

PCM-12 OMEGA BUS STRUCTURE

Page Number

3.1 SUMMARY OF BUS SIGNAL FUNCTIONS 3-2

SECTION 3.0

PCM-12 OMEGA BUS STRUCTURE

The 12530 backplane bus is a double-sided, plated-through fiberglass-epoxy printed-circuit board measuring approximately 16.1" x 6.0". Computer-aided graphic techniques were used to generate the printed-circuit pattern, providing an interlaced ground grid and full shielding for all signal lines. Thus, the performance of a multi-layer bus board is achieved using more economical two-sided printed-circuit fabrication techniques.

The bus board accomodates 18 Texas Instruments H431121-40 80-contact edge connectors. With one position occupied by the CPU module, this leaves 17 positions for memory and interface modules. Connector spacing has deliberately been made non-uniform so that printed-circuit modules having a variety of heights can be accomodated. Low-height boards can be placed to the left in the bus, where edge-connector spacing is just 0.6", and wire-wrapped prototypes can be placed near the center of the bus where the connector spacing is as much as 1.5". Pin-to-pin spacing on the edge connectors is 0.125 inch. The backplane board mounts to the tilt-up card cage in the Omega cabinet, and is supported by the edge connectors to which it is soldered.

The 12530 bus board eliminates all backplane wiring. Every pin in every connector is electrically identical to the corresponding pin in every other connector (except pins 45 and 46, which are used for the DMA and priority-interrupt capability, when implemented in the system, and are serially connected from connector-to-connector). When it is necessary for a given module to be electrically connected to peripheral equipment, these connections are normally accomodated by means of flat ribbon-cable which attaches to the module card by a standard ribbon-cable connector. Although the CPU is normally inserted in the bus at the extreme left end of the card cage (viewing the computer from the front) with memory to its immediate right, any PCM-12 module (CPU, memory or peripheral interface) can plug into any position on the backplane bus.

The PCM-12 bus contains 80 lines. All but the DMA- and interrupt-priority lines (bus lines 45 and 46) are parallel-connected to every edge connector. Table 3-2 summarizes the signals on the bus, information concerning the source of the signal, and the type of driver that asserts the line.

Some of the bus lines are heavily used in the system, and therefore are driven by high-current drivers. Other lines which are used by only a few modules, at most, are driven by TTL gates. A summary of the typical driver capabilities is given in Table 3-1. Each plug-in module usually loads each line with one standard TTL load or less, therefore the drivers in the system will be found to be more than adequate for a fully-loaded PCM-12 bus with 18 plug-in modules in place.

The letter H or L following each signal name in Table 3-2 indicates whether the assertion level for that signal is normally a TTL-logic high or low. H refers to a logic high (2.4 to 5.0 volts). L refers to a logic low (0.0 to 0.8 volts).

3.1 SUMMARY OF BUS SIGNAL FUNCTIONS

AUX L	This line is grounded when the front-panel AUX switch is in the "up" position. Otherwise, it is left to "float", unless asserted by a plug-in module.
BAUDRAT	This line carries a 614.4 KHz square-wave from the CPU module for Baud-rate generation on peripheral interfaces.
BREAK L	Asserted low by serial interface module to attempt interrupting a running program, usually by generating a control-panel interrupt.
C0 L C1 L C2 L	During the execute phase of an IOT instruction these lines are asserted by the device interface logic to define the exact operation that will take place (see Section 4.2.2).
CMOS DIS L	Disables the non-volatile CMOS memory modules when asserted low.
CPMEM L	This line is asserted low by the CPU, front-panel, or other module to force indirect operands to go-to/come-from control-panel memory, rather than main memory (see Section 5.1.2).
CPREQ L	This line is asserted low by the 12530 control-panel, or logic on other modules, to request a control-panel interrupt.
CPSEL L	The CPU module asserts this line low to read data from , or write data into, control-panel memory, switches or indicators during a control-panel interrupt (see Section 5.6 and Figures 5-8 and 5-9.
CPSEL* L	Auxiliary CP-select line driven only by 12120 module.
DATAF H	This line is asserted high when the CPU wishes to read from, or write into, the currently-specified data-field. The distinction between the instruction-field and data-field is only defined, and therefore the DATAF signal is only meaningful, when a memory extender module is present in the system (see Section 5.1.1).
DEVSEL L	During an IOTA execute cycle this line is asserted low by the CPU to read from, or write into, a peripheral device interface.

DMAREQ L — A peripheral device requests a direct-memory access by asserting this bus line low (see Section 5.5).

DMAGNT H — When the CPU grants a DMA "cycle-steal" request, it asserts this line high (see Section 5.5).

DX0-DX11 — These are the 12 multiplexed bi-directional lines that carry address, instruction and data between the CPU, memory and device interfaces (and front-panel). DX0 is the most significant bit, and DX11 the least significant bit.

EMA0 H
EMA1 H
EMA2 H — When the system employs more than 4K words of memory, these lines are driven by the memory extender module to develop the required effective 3-bit extension to the CPU's Program Counter and Memory Address Register (see Section 7.0.)

EXCPR L — This line can be asserted low by a peripheral device to request a control-panel interrupt (see Section 27.0)

FREERUN H — This line is controlled by the front-panel XTAL CLK/SNGL CLK toggle switch. When this switch is in the "up" position, the FREERUN line is high, and the CPU is driven by the crystal-controlled clock oscillator on the CPU module. When the switch is "down", FREERUN is low and the CPU is driven manually by the SNGL CLK pushbutton.

IFETCH H — This line is asserted high by the CPU throughout an instruction-fetch cycle (see Figures 5.1 and 5.2).

INTDIS L — The memory extender module, when present in the system, prevents peripheral devices from being granted an interrupt (when appropriate) by asserting this line low (see Section 7.0).

INTGNT H — When the CPU grants a device-interrupt request, it asserts this line high (see Section 5.4).

INTGNT* H — Driven high by 12510 CPU module at execution of VCTR instruction to indicate an interrupting PIE device should generate its interrupt vector to CPU.

INTREQ L — A I/O device interface requests an interrupt by asserting this line low (see Section 5.4).

LINK L — This line is asserted low whenever the CPU Link flip-flop is set.

LXMAR H — This line is asserted by the CPU early in most cycles. The falling edge of this pulse is used by memory and device interfaces to latch addresses and instructions into each module (see Figures 5.1 and 5.2).

MEMSEL L — This line is asserted low by the CPU to read from, or write into, main memory (see Figures 5.1 and 5.2).

MEMSEL* L — Similar to MEMSEL bus line, but not controllable by the CPMEM bus line (see Section 5.1.2).

MEMSEL INH L — This line is asserted low by the 12540 stack module to suppress writing subroutine and interrupt return addresses into memory.

PRIN L PROUT L	These two pins on each edge connector are used to implement the hardware-priority-interrupt and DMA-priority functions. For further discussion, see documentation on specific modules making use of these lines.
PWROK H	When asserted high, indicates that power-fail logic on 12330 or 12340 module has not detected any abnormality in AC power.
RESET L	When this line is asserted low by the front-panel RESET switch, it forces the CPU to its reset state (i.e., halted, with Program Counter set to 7777_8), and also initializes logic on many of the system's plug-in modules. This line can also be asserted low by the CAF instruction; for operational description see Table 4-3.
RUN L	The CPU asserts this line low when it is in the RUN state (causing the front-panel RUN lamp to light); when halted this line is high.
RUN/HALT L	This line is pulsed low by the control-panel logic to invert the run/halt state of the CPU.
SKIP L	A device interface causes the Program Counter to be incremented by asserting this line during an IOTA execute cycle (see Section 4.0).
SNGL CLK H	In the SNGL CLK mode of operation, this line is asserted high by the SNGL CLK pushbutton to advance the CPU by one clock cycle (one-half T-state). See discussion in Section 2.1.6.
SW L	This line is grounded when the front-panel SW switch is in the "up" position. Otherwise, it is left to "float", unless asserted by a plug-in module.
SWSEL L	This line is asserted low by the CPU during the execute phase of an OSR or LAS instruction to read the front-panel Switch Register (see Section 4.2.3 and Figure 4-2).
XTA H XTB H XTC H	These three lines are asserted by the CPU during various parts of each cycle. The respective states of the three lines indicate the instantaneous state of the CPU (see Figures 5.1 and 5.2).
VDD HI	This is an auxiliary power-supply line available to user. It is not normally driven by the PCM-12 power supply.
WAIT L	This line can be asserted low by a slow memory module or I/O device interface to cause the CPU to pause for an integral number of clock cycles while the memory module or device interface "catches up" with the CPU.
4 MHZ	This line carries a 4.0 MHz square-wave of identical phase to the square-wave that normally drives pin 14 on the IM6100 microprocessor.

NOTES

When selecting unused bus lines for customer use, the user should choose the higher-numbered bus lines. Additional bus lines required for PC/M-designed modules will use lower-numbered lines.

The 12510 CPU module should normally be plugged into the extreme left-hand bus slot, to place it electrically farthest from the 12530 front-panel module. This provides maximum system noise immunity.

Driver Type	Standard TTL-load Drive Capability	LSTTL-load Drive Capability
74365/366/367/368	20	80
74LS365/366/367/368	10	40
74LS00/02/04/08/86	5	20
74LS242/243/257A	15	60
7407,7417	25	100
74LS03/05/09	5	20
NPN transistor	80	320

TABLE 3-1

PCM-12 Bus Driver Characteristics

Line #	Signal Name	Description	Source	Driver
1	GND	System Ground	-	-
2	GND	System Ground	-	-
3	+5V	+5-volt supply line	Power supply	-
4	+5V	+5-volt supply line	Power supply	-
5	+5V	+5 volt supply line	Power supply	-
6	+5V	+5 volt supply line	Power supply	-
7	4 MHz	4 MHz square wave	CPU module	74LS86
8	RUN L	Indicates "run" or "halt"	CPU module	74LS04
9	LINK L	Indicates state of Link f/f	CPU module	74LS04
10	PWROK H	Indicates state of AC power	12330 or 12340 module	7407
11	INTGNT* H	Forces PIE's to vector	12510 CPU module	74LS08
12	EXCPR L	External control-panel request	external device	
13	EMA0 H	Extended memory address MSB	12540 EMA module	74LS367
14	-----	not used as of 1-FEB-80		
15	EMA1 H	Extended memory address	12540 EMA module	74LS367
16	CMOS DIS L	Disables CMOS memory modules	12330 or 12380 module	relay
17	EMA2 H	Extended memory address LSB	12540 EMA module	74LS367
18	RESET L	Resets CPU and I/O flags logic on I/O interfaces	Front-panel switch or CPU module	toggle switch or transistor
19	+12V	+12 volt supply line	Power supply	-
20	+12V	+12 volt supply line	Power supply	-
21	RUN/HALT L	Toggles Run/Halt f/f in CPU	12330 or 12530 module	7407 or 7417
22	DMAREQ L	Requests DMA action by CPU	Hi-speed I/O module	LS gate
23	MEMSEL INH L	Inhibits MEMSEL pulse	12540 stack logic	74LS03
24	INTREQ L	Requests I/O device interrupt	Any I/O interface	74LS03
25	MEMSEL* L	Un-gated main-memory select	12510 CPU module	74LS08
26	CPSEL* L	Auxiliary CP select pulse	12120 module	LS gate
27	SNGL CLK H	Single clock pulse to CPU	12530 Front panel	74LS04
28	INTDIS L	Prevents device interrupts	12540 EMA module	74LS03
29	DX11	Bi-directional address/data	CPU, memory, I/O	74365/74366
30	DX10	same	same	same
31	DX9	same	same	same
32	DX8	same	same	same
33	DX7	same	same	same
34	DX6	same	same	same
35	DX5	same	same	same
36	DX4	same	same	same
37	DX3	same	same	same
38	DX2	same	same	same
39	DX1	same	same	same
40	DX0	same (MSB)	same	same
41	SW L	Auxiliary control line	12530 Front-panel	toggle switch
42	AUX L	Auxiliary control line	12530 Front-panel	toggle switch
43	CPSEL L	Control-panel memory SELect	12510 CPU module	74365

(next page)

Line #	Signal Name	Description	Source	Driver
44	BAUDRAT	614.4 KHz square wave	12510 CPU module	74LS04
45	PRIN L	Priority input line	I/O interface	LS gate
46	PROUT L	Priority output line	I/O interface	LS gate
47	CPMEM L	MEMSEL/CPSEL switch control	12510/12530/12330	7407/74LS03
48	IFETCH H	Indicates instruction-fetch	12510 CPU module	74LS365
49	-----	not used as of 1-FEB-80		
50	C0 L	CPU control line	I/O interface	74LS03
51	C1 L	CPU control line	I/O interface	74LS03
52	C2 L	CPU control line	I/O interface	74LS03
53	DATAF H	Selects data-field	12510 CPU module	74365
54	-----	not used as of 1-FEB-80		
55	LXMAR H	Latches address/instruction into memory and I/O modules	12510 CPU module	74365
56	-----	not used as of 1-FEB-80		
57	-12V	-12 volt supply line	Power supply	-
58	-12V	-12 volt supply line	Power supply	-
59	XTA H	Indicates CPU state	12510 CPU module	74LS365
60	DEVSEL L	I/O device SELect line	12510 CPU module	74365
61	WAIT L	Causes CPU to pause	Memory or I/O module	74LS03
62	BREAK L	Break from terminal		
63	-----	not used as of 1-FEB-80		
64	XTC H	Indicates CPU state	12510 CPU module	74365
65	XTB H	Indicates CPU state	12510 CPU module	74LS365
66	CPREQ L	Requests CP interrupt	Control-panel or CPU	74LS03
67	-----	not used as of 1-FEB-80		
68	SWSEL L	Selects Switch-Register input	CPU module	74LS365
69	VDD HI	Auxiliary power line	User's power supply	-
70	VDD HI	Auxiliary power line	or battery	-
71	FREERUN H	crystal/single-clock switch	Front-panel module	toggle switch
72	SKIP L	Increments CPU Program Counter	I/O interface	74LS03
73	DMAGNT H	Indicates DMA action	12510 CPU module	74LS365
74	INTGNT H	Indicates interrupt granted	12510 CPU module	74LS365
75	MEMSEL L	Main-memory SELect line	12510 CPU module	74365
76	-----	not used as of 1-FEB-80		
77	-----	not used as of 1-FEB-80		
78	-----	not used as of 1-FEB-80		
79	GND	System ground	-	-
80	GND	System ground	-	-

TABLE 3-2

Summary of PCM-12 Bus Signals

SECTION 4.0

SOFTWARE CONSIDERATIONS

		Page Number
4.1	MEMORY ORGANIZATION	4-1
4.2	INSTRUCTION SET	4-2
	4.2.1 MEMORY REFERENCE INSTRUCTIONS	4-3
	4.2.2 INPUT/OUTPUT INSTRUCTIONS	4-4
	4.2.3 OPERATE INSTRUCTIONS	4-8
4.3	FUNDAMENTAL PDP-8 SOFTWARE	4-9
4.4	ADVANCED PROGRAMMING LANGUAGES	4-12
4.5	PCM-12 BINARY BOOTSTRAP ROUTINE	4-13
	4.5.1 PROCEDURE FOR LOADING A BINARY-FORMAT TAPE	4-15

SECTION 4.0

SOFTWARE CONSIDERATIONS

Programming the PCM-12 microcomputer is nearly identical to programming DEC's PDP-8 family of minicomputers. This section provides an overview of PCM-12 software considerations. For a very detailed discussion of PDP-8 (and PCM-12) programming, the user is referred to DEC's Introduction to Programming (ItP) as an indispensable basic supplement to this manual.

4.1 MEMORY ORGANIZATION

Like the PDP-8, the PCM-12 has a basic addressing capacity of 4096 (4K) 12-bit words. This addressing capacity is a natural result of the machine's 12-bit word length, and can be expanded to 32K (or even beyond) by addition of the 12540 Memory Extender module to the system. The 12540 module executes the same memory-extending instruction set as DEC's KM8-E Memory Extender for the PDP-8/E. (The 12540 module also provides a programmable real-time clock and a 256-word hardware push-pop stack for saving interrupt and subroutine return-addresses, as well as general FILO-type data storage.)

The memory system is organized into 4096-word blocks called "fields". The first 4K words are in Field 0. If a full 32K of memory is installed, the uppermost memory field is numbered 7. In any given memory field every location has a unique 4-digit octal (12-bit binary) address, 0000_8 to 7777_8 (0000_{10} to 4095_{10}). Each memory field is divided into 32 pages of 128 words each. Memory pages are numbered sequentially from page 00_8, containing addresses 0000_8-0177_8, to page 37_8, containing addresses 7600_8-7777_8. The first five bits of a 12-bit memory address denote the page number and the low-order 7 bits specify the address of the memory location within the page, called the Page Address.

To select the proper memory field from among the eight that may be present in the system, the 12540 Memory Extender module provides a three-bit extension to the memory-addressing word generated by the CPU's Program Counter and Memory Address Register. Normally these three bits come from the Instruction Field register on the 12540 module. However, during the execute cycle of an indirectly-addressed AND, TAD, ISZ or DCA instruction, when the DATAF line (line 53) on the PCM-12 bus is asserted by the CPU, the three-bit extension is derived from the Data Field register on the 12540 module.

4.2 INSTRUCTION SET

The PCM-12 (and PDP-8) instruction set is divided into three categories: Memory Reference Instructions (MRI's), Operate Instructions (OPI's) and Input-Output Transfer Instructions (IOT's). The high-order three bits (on bus lines DX0-DX2) denote the instruction type. MRI's begin with 0_8, 1_8, 2_8, 3_8, 4_8 or 5_8. All IOT's start with 6_8, and all OPI's start with 7_8. This first octal digit in the instruction code is called the "opcode".

Table 4-1 details the required machine cycles, and the T-states required in each cycle, for each type of instruction:

INSTRUCTION TYPE	OPCODE	REQUIRED CYCLES first T-states	second T-states	third T-states	fourth T-states
AND	0_8				
directly addressed		5 [IFETCH]	5 [execute]		
indirectly addressed		5 [IFETCH]	5 [indirect]	5 [execute]	
auto-indexed		5 [IFETCH]	6 [auto-index]	5 [execute]	
TAD	1_8				
directly addressed		5 [IFETCH]	5 [execute]		
indirectly addressed		5 [IFETCH]	5 [indirect]	5 [execute]	
auto-indexed		5 [IFETCH]	6 [auto-index]	5 [execute]	
ISZ	2_8				
directly addressed		5 [IFETCH]	6 [increment]	5 [execute]	
indirectly addressed		5 [IFETCH]	5 [indirect]	6 [increment]	5 [execute]
auto-indexed		5 [IFETCH]	6 [auto-index]	6 [increment]	5 [execute]
DCA	3_8				
directly addressed		5 [IFETCH]	6 [execute]		
indirectly		5 [IFETCH]	5 [indirect]	6 [execute]	
auto-indexed		5 [IFETCH]	6 [auto-index]	6 [execute]	
JMS	4_8				
directly addressed		5 [IFETCH]	6 [execute]		
indirectly addressed		5 [IFETCH]	5 [indirect]	6 [execute]	
auto-indexed		5 [IFETCH]	6 [auto-index]	6 [execute]	
JMP	5_8				
directly addressed		5 [IFETCH]	5 [execute]		
indirectly addressed		5 [IFETCH]	5 [indirect]	5 [execute]	
auto-indexed		5 [IFETCH]	6 [auto-index]	5 [execute]	
IOT	6_8	5 [IFETCH]	6 [IOTA]	6 [IOTB]	
OPI	7_8				
2-cycle OPI		5 [IFETCH]	5 [execute]		
3-cycle OPI		5 [IFETCH]	5 [execute]	5 [execute]	

TABLE 4-1

Required Machine Cycles and T-States for Each Instruction Type

4.2.1 MEMORY REFERENCE INSTRUCTIONS

The memory reference instructions operate on the contents of a memory location or use the content of a memory location to operate on the CPU's Accumulator or Program Counter.

Operation of each of the MRI's is detailed in Table 4-2. Each of these instructions may be directly-addressed, indirectly-addressed, or auto-indexed. When one of the MRI's is to be directly addressed, the absolute address of the operand is embedded in the instruction itself, so only two cycles are required for a complete fetch and execution. (For more information, see ItP page 2-8.)

When a MRI is indirectly addressed, the second machine cycle is an "indirect" (called "defer" by DEC) cycle, which is used to pick up the desired absolute address of the operand from memory. Execution of the instruction takes place in the third (and fourth for an ISZ) cycle. This mode of addressing must be used when the desired address of the operand is not on the current page or on page 00_8. (See page 2-15 in ItP for more on indirect addressing.)

MNEMONIC	OPCODE	OPERATION
AND	0_8	Logical AND. Operand is AND'ed with contents of Accumulator. Result remains in the Accummulator.
TAD	1_8	Binary ADD. Operand is added to Accumulator contents; result remains in the Accumulator. Carry out complements the Link. Commonly used for Accumulator load if Accumulator is initially clear.
ISZ	2_8	INCREMENT, AND SKIP IF ZERO. Operand is incremented and the next instruction is skipped if the result was zero. Accumulator is unaffected.
DCA	3_8	DEPOSIT TO MEMORY, AND CLEAR ACCUMULATOR. Contents of Accumulator are deposited in operand address, then the Accumulator is cleared.
JMS	4_8	JUMP TO SUBROUTINE. Contents of Program Counter are deposited in operand address. Then Program Counter is incremented. Accumulator is unaffected.
JMP	5_8	UNCONDITIONAL JUMP. Program Counter is set to the operand address. Accumulator is unaffected.

TABLE 4-2

Operation of Memory Reference Instructions

By extending an "indirect" cycle to six states, an instruction can be auto-indexed. Whenever a memory location from 0010_8-0017_8 is used as a pointer in an indirectly-addressed MRI, its contents are incremented before being used as the absolute address of the operand. This provides a convenient means for indexing applications. (See page 3-27 in ItP for complete instructions on auto-indexing.)

4.2.2 INPUT/OUTPUT INSTRUCTIONS

The second category of instructions is the Input/Output Transfer (IOT) instructions. These all have an opcode of 6_8 and are used to initiate the operation of peripheral devices, and to transfer data between peripherals and the CPU. Using IOT's all device data movements are programmed data transfers; device data can also be moved to/from memory and CPU by means of interrupt-initiated transfers, or by direct-memory-access. Programmed data transfers are the simplest way to move data to/from peripheral devices, but are also the slowest.

In an IOT instruction, bits 0-2 are always set to 110_2 (6_8). Unless the selected device interface employs the 6101 PIE device, bits 3-8 are the device selection code used to specify the peripheral device, and bits 9-11 specify the operation to be performed with the selected peripheral. [When the PIE device is used on the device interface module, bits 3-7 specify the device and bits 8-11 the operation to be performed.]

The device selection code 000_2 000_2 (00_8) in bits 3-8 is reserved for "Processor IOT's". There are eight of these: 6000_8-6007_8. They are used by the CPU for certain "housekeeping" functions associated with the interrupt system. The operation of each of the Processor IOT's is detailed in Table 4-3.

A programmed data transfer begins when the CPU fetches an instruction from memory and recognizes it as an IOT instruction. The CPU sequences the IOT instruction through a 2-cycle execute phase referred to as IOTA and IOTB. See Figure 4-1. The instruction is latched into the device interface during IOTA, using the trailing edge of the LXMAR pulse. DEVSEL is the active SELect line for all IOT instructions. The selected peripheral device controls the CPU during the data transfer by means of the CO, C1, C2 and SKIP lines on the

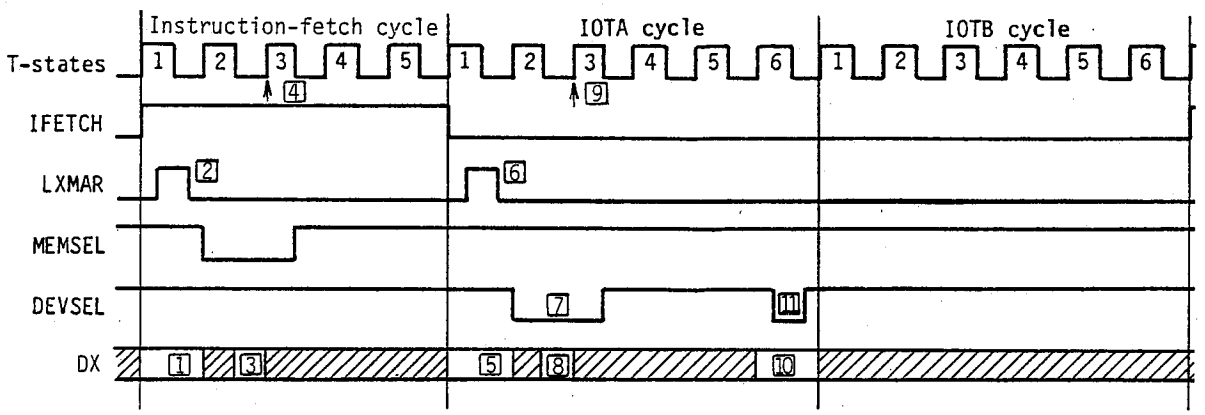
MNEMONIC	OCTAL	OPERATION
SKON	6000	SKIP IF INTERRUPT ON. The next instruction is skipped if the CPU's Interrupt-Enable Flip-Flop is set, then this flip-flop is reset (cleared).
ION	6001	INTERRUPTS ON. The CPU Interrupt-Enable Flip-Flop is set immediately after fetching the next instruction. (At least one more instruction will be executed after the ION before the first interrupt is recognized.)
IOF	6002	INTERRUPTS OFF. Immediately resets (clears) the CPU's Interrupt-Enable Flip-Flop, so no further interrupts will be allowed.
SRQ	6003	SKIP IF INTERRUPT REQUEST. If the INTREQ pin (pin 8) on the 6100 CPU device is asserted low, the next instruction will be skipped.
GTF	6004	GET FLAGS. The following flag bits are jammed into the Accumulator: AC(0): Link flip-flop [AC(0)=1 if Link f/f set]. AC(2): INTREQ pin (pin 8) on 6100 [AC(2)=1 if pin 8 asserted low]. AC(3): Interrupt-Inhibit Flip-Flop on 12540 Memory Extender module [AC(3)=1 if IIFF set]. AC(4): CPU's Interrupt-Enable F/F [AC(4)=1 if IEFF is set]. AC(6-11): Save-Field Register on 12540 module.
RTF	6005	RETURN FLAGS. Link is set by AC(0). Interrupt-Inhibit Flip-Flop on 12540 Memory Extender is unconditionally set until the next JMP or JMS instruction is executed. CPU Interrupt-Enable Flip-Flop is unconditionally set, after next instruction, as in ION. Instruction Buffer register (IB) on 12540 module is loaded from AC(6-8), and Data Field (DF) register is loaded from AC(9-11). IB register will be transferred to IF register as next JMP or JMS is being executed. Accumulator not cleared.
SGT	6006	Not used by PCM-12 (used for KE8-E module on PDP-8).
CAF	6007	CLEAR ALL FLAGS. Accumulator and Link are cleared. Interrupt-Enable Flip-Flop is reset (cleared). This instruction is also used by most device interfaces to clear flag flip-flops and set their individual interrupt-enable flip-flops.

TABLE 4-3

Processor IOT Instructions

bus. The type of data transfer is specified by the peripheral device interface by asserting the control lines as shown in Table 4-4. Also see note 3 on Figure 5-2.

Except for Processor IOT's, all IOT instructions are non-specific in that, unlike all other instructions, the operations that they perform are not "known" by the CPU. Rather, the hardware designer specifies what each of these instructions does by the logic he builds into the interface for each specific peripheral device. The IOT instructions work in conjunction with the



1 - Instruction address driven onto DX lines by CPU.
2 - Instruction address latched into memory on trailing edge of LXMAR.
3 - Memory module drives instruction onto DX lines.
4 - IOT instruction read into CPU from memory.
5 - CPU drives instruction onto DX lines.
6 - IOT instruction latched into device interfaces on trailing edge of LXMAR.
7 - Device-read SELect pulse.
8 - Selected device interface drives DX lines with "read" data.
9 - CPU reads DX lines as well as C0, C1, C2 and SKP bus lines at this time.
10 - CPU drives "write" data onto DX lines.
11 - Device-write SELect pulse.

FIGURE 4-1
Basic IOT Instruction Timing

CO, C1, C2 and SKIP bus lines to the CPU. For example, for a PDP-8 Teletype interface, it is necessary that instruction 6034_8 cause the TTY keyboard data to be OR'ed into the Accumulator. Referring to Table 4-4, it is seen that in order to cause device data to be OR'ed into the Accumulator, it is necessary to assert control line C1 low while CO and C2 remain high at DEVSEL time. The interface logic, then, must recognize the arrival of the 6034_8 instruction ands assert C1 low when DEVSEL is asserted low at T3 time. Similarly, instruction 6031_8 must cause the next instruction to be skipped if the Keyboard Data Ready Flag is set on the device interface. To accomplish this, the interface logic must, upon arrival of the 6031_8 instruction, test the TTY Data Ready Flag, and then if (and only if) it is set, assert the SKIP bus line low.

CONTROL LINES CO	C1	C2	OPERATION
H	H	H	Accumulator contents written into device. AC not cleared.
L	H	H	AC contents written into device, then AC cleared.
H	L	H	Device data OR'ed into Accumulator.
L	L	H	Device data jam-transferred into Accumulator.
X	H	L	Device data added to contents of Program Counter (relative jump).
X	L	L	Device data jam-transferred into Program Counter (absolute jump).

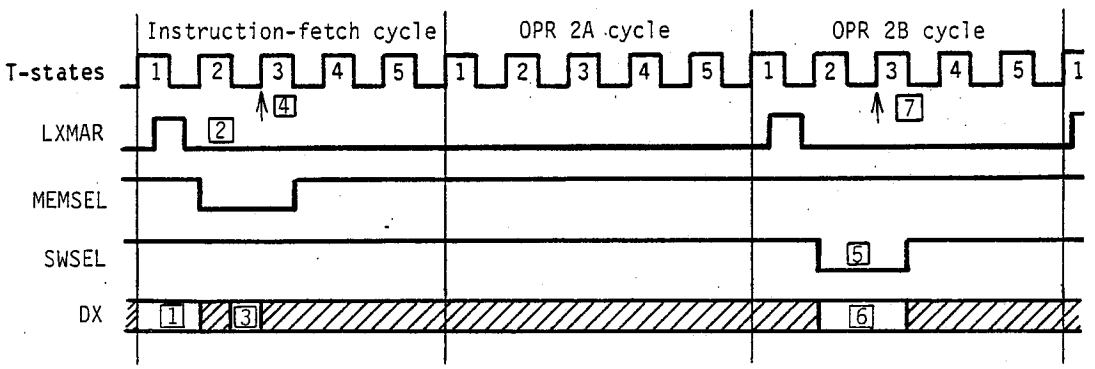
X = irrelevant

TABLE 4-4
Control Lines [CO, C1, C2] Operation

The system designer has nearly complete freedom with the IOT instructions. He first decides what he wants a given IOT to do, then builds the necessary "interpretive" logic into his peripheral interface. To retain DEC software compatibility, though, the designer must use the IOT instructions as already defined in DEC's Small Computer Handbook and Introduction to Programming, and assure that the interface logic "understands" those instructions.

4.2.3 OPERATE INSTRUCTIONS

The third category of instructions are termed the Operate Instructions, all of which have the opcode of 111_2 (7_8). These instructions are all used for CPU internal operations, such as conditional and unconditional skips, Accumulator/Link rotates (either right or left, and one-bit or two-bit shifts), clearing and setting the Accumulator and Link, transferring data between the MQ Register and Accumulator, etc. These instructions use bits 3-11 in the instruction (after the opcode 111_2 in bits 0-2) to specify the exact operation to be performed. All these bits are available, of course, since nearly all operations specified are internal to the CPU itself and do not require specification of a memory address or device code.



1 - CPU drives instruction-address onto DX lines.
2 - Address latched into memory modules on trailing edge of LXMAR.
3 - Memory module drives instruction onto DX lines.
4 - CPU reads instruction from DX lines.
5 - SWSEL line is asserted, allowing Switch Register to drive DX lines.
6 - Switch Register data driven onto DX lines.
7 - Switch Register data read from DX lines by CPU.

Figure 4-2
OSR Instruction Timing

A complete listing of the OPI's is given on pages D-3 to D-5 in ItP (and a better listing is given in the Intersil and Harris data sheets for the 6100 device). It should be pointed out that these instructions are actually termed "micro-instructions", since by setting or not setting given bits in the instruction word, they can be combined with one another. This cuts down the number of individual steps necessary in a program. It is possible, for example, to use a single instruction to produce 0006_8 in the Accumulator by the combination CLA CLL CML IAC RTL. This first clears the Accumulator and Link, then complements the Link, then increments the Accumulator and finally rotates the Accumulator (through the Link) two bits to the left. The instruction would be coded as 7327_8 by the assembler. A complete discussion of the operation of the OPI's is given on pages 2-18 to 2-28 in ItP.

There is one unique OPI which is particularly noteworthy, since it acts somewhat like an IOT instruction. This is the OSR instruction, which OR's the state of the front-panel Switch Register into the CPU's Accumulator. The timing diagram for this instruction is shown in Figure 4-2.

4.3 FUNDAMENTAL PDP-8 SOFTWARE

Since the PCM-12 and PDP-8 are software-compatible, the general rule is that software written for either machine will run on the other. This software compatibility specifically includes the basic paper-tape system software that DEC has long offered for the PDP-8 series of minicomputers.

For the PCM-12 user without software, DEC's PDP-8/E Extended Software Kit is recommended. This kit contains exhaustive documentation on the PDP-8 "system", and several very useful, and nearly essential, paper tapes. The software kit is designated number QF081-CB. It is a copyrighted system of software available from DEC for less than $200.00. No licensing agreement is required. The kit may be ordered from one of DEC's three Technical Documentation Centers located at 2525 Augustine Drive, Santa Clara, CA 95051; 712 West Algonquin Road, Arlington Heights, IL 60005; or Cotton Road, Nashua, NH 03060. (A copy of the PDP-8 Software Catalog is available free, upon request, from the same address.)

The PDP-8/E Extended Software Kit contains the following very useful manuals:

PDP-8/E Small Computer Handbook
Introduction to Programming
Logic Handbook (late edition)
PDP-8 Family Utility Routines
4K Assemblers PAL III and MACRO-8
Self-Starting Binary Loader
PDP-8 Pocket Reference Card

The Small Computer Handbook is the "bible" of PDP-8 system hardware. It is very useful for the PCM-12 user who intends to develop his own custom interfaces for peripheral devices, and for the user wanting to gain a thorough understanding of PDP-8 hardware organization. It may be ordered individually as catalog number EB-02546-76.

The PDP-8/A Minicomputer Handbook (catalog number EB-06219-76) is also very useful for anyone contemplating add-on hardware design for the PCM-12 system.

Introduction to Programming (ItP) has been mentioned previously in this manual, and is indispensable. It may be ordered individually as catalog number DEC-08-XINPA-A-D.

The Logic Handbook is a thorough treatment of DEC's available hardware for interfacing the PDP-8 to the user's unique equipment. As a companion to the Small Computer Handbook it is a useful aid in learning the PDP-8 "system". Catalog number EB-06036-76.

The titles of the remaining handbooks adequately describe their contents.

The PDP-8/E Extended Software Kit kit also contains the following binary paper tapes:

PAL III Assembler
Dynamic Debugging Technique (DDT)
Octal Debugging Technique (ODT), high and low
Symbolic Editor
Self-Starting Binary Loader
RIM-format Punch Routine
Octal Memory Dump Routine
Binary Punch Routine
23-Bit Floating Point Package

Additionally, a RIM-format binary loader tape is included. (Normally this tape is not needed with a PCM-12, since the machine's front panel contains its own binary-format loader in control-panel read-only memory.)

The Symbolic Editor is used to create and modify symbolic (source) program tapes from the Teletype keyboard, eliminating the tedious task of preparing such tapes off-line. The Editor is fully interactive. The editing changes may be verified and re-corrected, if necessary. The Editor includes a search feature to scan the text for occurences of a specific character or string. Other commands permit blocks of data to be inserted, deleted, appended, listed, moved or changed. The Editor is documented in Chapter 5 of ItP.

PAL III is a three-pass assembler designed for the PDP-8 family of computers. It operates with just 4K words of memory. During the first pass of the assembly, all user symbols are defined and placed in the assembler's symbol table. During the second pass, the binary equivalent of the input source language is generated and punched. The assembler's third pass, which is optional, produces a printed assembly listing of the program instructions, with the location and generated binary and source code listed side by side on each line. The binary tape output of the second pass can be loaded into the PCM-12 for execution, using the ROM-stored binary loader in the front panel.

The DEC manual entitled 4K Assemblers (listed above) contains descriptions of two PDP-8 4K assemblers, the most basic of which is PAL III. In addition to PAL III, the document also discusses the MACRO-8 assembler, which is similar to PAL III, with some additional features such as user-defined macros, Boolean operators, double-precision integers and floating-point constants, text facilities, etc. MACRO-8 has a smaller symbol table capacity, however, than PAL III.

Dynamic Debugging Technique (DDT) and Octal Debugging Techhnique (ODT) are two debugging programs for the PDP-8. These two service programs allow the user to run his program on the computer and use the Teletype keyboard to control program execution, examine registers, change their contents and make alterations to the program. With DDT, the user can debug the program using the symbolic language of the source listing, with DDT performing all translations to and from the binary representation. ODT has the same capabilities as DDT, except the programmer must use octal representation instead of the mnemonic symbols. Chapter 5 of ItP discusses the features of both of these

de-bugging routines.

The RIM and BIN punch programs provide the means for punching information contained in selected memory locations in RIM or BIN format via the ASR-33 Teletype paper-tape punch.

A detailed description of the various loaders, verifiers, duplicators, conversion and printing routines is given in the PDP-8 Family Utility Routines handbook, listed above.

The 23-Bit Floating Point Package (FPP) provides an easy means for performing basic arithmetic operations such as addition, subtraction, multiplication and division using floating-point numbers. It also provides extended function capabilities for the computation of natural logarithms, exponential functions, basic trigonometric functions and the like. The 23-bit FPP maintains a high degree of precision and greatly facilitates I/O operations in floating-point notation. Chapter 8 of ItP describes the functional features of the 23-bit FPP.

4.4 ADVANCED PROGRAMMING LANGUAGES

DEC provides many high-level-language software packages for the PDP-8. The most popular are FOCAL, BASIC and FORTRAN, although several others are also listed in the software catalogs. 4K ALGOL is available from DECUS for about ten dollars. DEC's DIBOL is an advanced business-applications language somewhat similar to COBOL, but much easier to learn and use.

8K and 16K SPL are two very powerful high-level FOCAL-like languages available directly from PC/M, Inc.

The easiest high-level language to learn on the PDP-8 or PCM-12 is FOCAL (or better yet, SPL). DEC's FOCAL-8 is an interactive language (algebraic) developed specifically for the PDP-8/E. FOCAL's desk-calculator mode of operation makes the full computational power of the computer available to the user in response to simple sentence-structured keyboard commands. FOCAL is similar to BASIC and FORTRAN in many respects, but is more easily and quickly learned. The dynamic combination of computational capability and simplicity makes FOCAL-8 an ideal language for on-line problem solving without having to master a complex programming language. FOCAL-8 requires only 4K words of memory, yet it offers a full range of mathematical functions, extendable I/O and versatile self-editing capabilities.

PC/M's 8K SPL is an extended version of FOCAL requiring a minimum of 8K words of memory. 16K SPL is intended to run under DEC's OS-8 mass-storage operating system, with at least 16K words of memory.

From DEC, the 4K FOCAL-8 binary paper tape is available as catalog number DEC-08-LFL8A-A-PB for about ten dollars. The manual on the language is catalog number DEC-80-LFL8A-A-D. Using other tapes which are readily available, this 4K FOCAL system can be extended for use with 8K or more words of memory (though for users with at least 8K of memory 8K SPL is a better choice). Other versions of FOCAL are available from DECUS for less than ten dollars.

DEC makes available two forms of paper-tape FORTRAN for the PDP-8. One is for 4K machines and the other is for 8K and larger machines. For each version, the "document" handbook and binary paper tapes run about fifty dollars. See the PDP-8 software catalog for detailed listings of these offerings. (FORTRAN IV is also available, as an add-on to the OS-8 operating system.)

Probably the most popular high-level language today is BASIC. The least expensive form of this language (4K BASIC) is available from PC/M for less than ten dollars.

For the PDP-8, DEC makes available several forms of BASIC, many of which are parts of their EduSystem software packages. Their 8K BASIC, however, is a stand-alone form of paper-tape BASIC. The manual for this BASIC interpreter costs about ten dollars from DEC. The binary paper tape costs about one hundred dollars, however, and at this price level the user would be much wiser to purchase 8K SPL from PC/M.

The user who is interested in exploring the mountain of available PDP-8 software should obtain a copy of the DEC PDP-8 software catalog, and a copy of the DECUS PDP-8 catalog as well. The latter is available at very nominal cost from DECUS at One Iron Way, Marlboro, MA 01752.

4.5 PCM-12 BINARY BOOTSTRAP ROUTINE

The binary-format loader routine, which resides in the control-panel PROM's on the PCM-12, is capable of loading into memory any PDP-8 compatible binary-format program, from either paper tape or audio cassette. The binary program may have been generated by any of the PAL III, PAL D, PAL 8, or

MACRO-8 assemblers, or Intersil's FOPAL FORTRAN cross assembler for the IM6100 microprocessor. The tape may contain diagnostic messages. The BIN BOOT routine is functionally identical to DEC's binary loader for the PDP-8, but does not require any space in main memory. Thus, the top page of memory (locations 7600_8-7777_8), which is normally used to hold the RIM and BIN loaders, is available for the programmer's use.

Tapes to be loaded by the BIN BOOT routine must be in binary-coded format and have a few inches of leader-trailer code (any code with channel 8 punched - preferably code 200) at both ends of the tape. The first two characters after the leader represent the address, or "origin". The initial character of the origin has no punch in channel 8, but channel 7 is punched. The second character designating the origin has no punch in either channel 7 or channel 8. The following data characters also have no punch in either channel 7 or 8. A 12-bit data word is represented by two adjacent 6-bit characters on the tape in channels 6 through 1, channel 6 of the initial character being the most significant bit. The data characters are loaded into sequential memory locations following the origin set-up. If more than 4K of memory is used, the assembler outputs a "field-setting" command of the form 11 XXX 000 (in channels 8-1) to indicate the memory field into which the following data is to be loaded. If, for example, XXX were 101, all data following the field designator would be loaded into memory field 5. Trailer tape is similar to the leader. A concluding 2-character group just before the trailer represents the checksum and has no punches in channel 7 or 8.

When any of the PDP-8 assemblers are used to produce a binary tape, a checksum is automatically punched at the end of the tape. This is the sum of all data on the tape including the origin, but excluding diagnostic messages, leader/trailer code and field-setting data. The checksum is accumulated character-by-character, not word-by-word. Carry out of the Accumulator is ignored.

If the checksum accumulated while using the binary-format loader does not agree with the last two characters on the tape (i.e., the checksum on the tape calculated and placed there by the assembler), an error in loading has occurred.

The PCM-12 will halt after the tape has been loaded. At this point, the content of the Accumulator should be zero if the tape was read without error. If not, the tape must be re-loaded.

If the tape was started before the leader, the PCM-12 will halt at the first leader character, with the Accumulator content equal to 7600_8 or 0000_8, depending on the number of blank characters read before the leader.

The binary-format loader routine may be used to load a binary tape into any valid memory field. However, if the 12540 Memory Extender module is not present in the system, any field-setting data on the binary tape are ignored.

A listing of the binary loader routine, a part of the 12530 control-panel software, will be found at the end of Section 2.0. The routine proceeds as follows: the incoming character is first tested to see if it is a "rub-out" (all eight channels punched). If this is the case, all subsequent data coming from the reader are ignored until another rub-out is found. This is the mechanism by which the assembler's diagnostic messages are detected; they are preceded and followed by a single rub-out character. Within the diagnostic message any character is valid except, of course, a single rub-out character, which would conclude the diagnostic message. Note that two consecutive rub-outs within the diagnostic message would, in effect, be ignored.

Next, the character is tested to see if it is leader/trailer or field-setting code. Leader information is ignored. A "change data-field" routine is executed if the character is in field-setting format.

If the character is not a part of a diagnostic message, leader/trailer or field-setting, then it is part of an origin address or contains part of a data word, and is part of the checksum. The appropriate course is followed. The binary loader always looks ahead one character to see if trailer code follows the character just read. If it does, then the two characters read just before the trailer was found are the checksum.

4.5.1 PROCEDURE FOR LOADING A BINARY-FORMAT TAPE

1. Bring the machine to HALT state. Raise, and release, RESET switch.
2. Place the binary-format tape to be loaded in the reader. The leader code on the tape must be over the read head. Assure that the reader is "on-line".
3. Load the Data Field register with the binary code for the field into which it is desired to load the program, using the EXTD ADDR load switch and the SWITCH REGISTER. (This step is unnecessary if the 12540 Memory Extender is not present in the system.)

4. If the low-speed reader is in use, set SWITCH REGISTER bit 0 to "1" (up), and all other SWITCH REGISTER bits to "0" (down). If the high-speed reader is to be used, all SWITCH REGISTER bits should be set to "0" (down).
5. Activate the BOOT switch. The tape should begin to read in. The PCM-12 will halt at the end of the tape. While the tape is being read in, the MEMORY ADDRESS display shows the address into which each data word is loaded and the Display lamps show the data word itself, regardless of the position of the Rotary Switch.
6. When the machine halts at the end of the tape, the Accumulator holds the checksum result, which should be zero. If the Rotary Switch was in the AC position when the machine halted, the Display lamps will automatically show the checksum result at the end of the load. If not, it is necessary to place the Rotary Switch in the AC position and activate the EXAM switch to read the checksum. The MEMORY ADDRESS display shows the address of the last word deposited to memory. The Data Field register on the 12540 module remains set to the last data field specified by the binary tape.

Now, to start the program:

1. Lift, and release, the RESET switch.
2. Load the Instruction Field register on the 12540 Memory Extender with the field in which the program starts. Load the Data Field register to the proper data-field setting (usually the same as the instruction field).
3. Place the starting address of the program in the SWITCH REGISTER. Activate the ADDR LOAD switch.
4. Bring the RUN/HALT toggle switch to the RUN position.
5. Activate the CONT/SNGL INST switch.

SECTION 5.0

FUNDAMENTAL MACHINE OPERATION

		Page Number
5.1	12510 CENTRAL PROCESSOR UNIT	5-1
	5.1.1 CPU MODULE LOGIC OPERATION	5-1
	5.1.2 SPECIALIZED INSTRUCTIONS FOR THE 12510 CPU	5-3
	5.1.3 HARDWARE SETUP OF 12510 CPU MODULE	5-8
5.2	CPU TIMING	5-9
	5.2.1 DETAILED CPU TIMING	5-13
5.3	PROCESSOR OPERATION PRIORITIES	5-14
5.4	DEVICE INTERRUPT TRANSFERS	5-15
5.5	DIRECT MEMORY ACCESS	5-19
5.6	CONTROL PANEL INTERRUPTS	5-20
5.7	RESET ACTION	5-23
5.8	CPU RUN/HALT FLIP-FLOP	5-25

SECTION 5.0

FUNDAMENTAL MACHINE OPERATION

This section contains a detailed discussion of the fundamental hardware operations of the PCM-12 Omega microcomputer, including operation of the 12510 CPU module. The user who would like to thoroughly understand all machine operations should carefully review this material, as well as the Intersil documentation on the IM6100 microprocessor.

5.1 12510 CENTRAL PROCESSOR UNIT

The 12510 CPU module is the heart of the Omega microcomputer. It is an improved re-design of the PCM-12's original 12010 CPU. The double-sided, plated-through printed-circuit board contains the IM6100 CMOS microprocessor device and the necessary logic to interface it to the PCM-12's TTL-level bus structure. The 12510 module contains the 4.0 MHz system clock, as well as another crystal-controlled oscillator whose output is divided down to provide the 614.4 KHz square-wave BAUDRAT bus signal. Also contained on the module is logic to develop the INTGNT* H bus signal, clock-speed switching circuitry which the user can activate while testing the system's RAM memory space, circuitry to provide the ability for the user to store or retrieve control-panel memory data directly from main-memory (and vice-versa), and logic to allow "hidden" routines stored in control-panel memory to be executed from the user's main-memory program.

5.1.1 CPU MODULE LOGIC OPERATION

See Drawing 12512, the schematic diagram for the 12510 CPU module. All output signals from the IM6100 microprocessor are driven onto the PCM-12 bus through TTL drivers. Signals which are heavily used in the system use high-current drivers. Signals which are used by only a few modules use TTL gates for bus drivers.

The bi-directional, multiplexed data/address lines, DX0-DX11, are interfaced to the microprocessor through devices U27-U30. For "read" data going to the microprocessor, type 74LS365 devices are used at U27 and U30. These are enabled by gate U26B whenever the IM6100 timing signal XTC is high and one of the SELect signals (MEMSEL, CPSEL, DEVSEL or SWSEL) is asserted

low. Write-data and addresses are driven from the microprocessor onto the bus through the 74365 devices at U28 and U29. These drivers are enabled by gate U23D, so that bus data is valid whenever timing signal XTB is high, except during a direct-memory-access (DMA) operation, when DMAGNT is asserted high. During the time that U28 and U29 are third-stated (i.e., not enabled) the CPU module allows the bus DX lines to "float".

The microprocessor output lines XTA, XTB, XTC and DMAGNT are driven onto the bus through sections of 74365 and 74LS365 drivers U21 and U25. XTA, XTB and DMAGNT are never third-stated. XTC and the other five signals driven onto the bus by U25 (MEMSEL, CPSEL, DATAF, LXMAR and DEVSEL) are third-stated during a DMA operation, while pin 1 of U25 is pulled high. All lines left floating by the CPU during a DMA operation are asserted by a DMA port, which is part of a (optional) device-interface for a high-speed peripheral, such as the 12430 hard-disk interface.

The microprocessor output signals IFETCH, SWSEL, and INTGNT are driven onto the bus through sections of 74LS365 device U20, while LINK and RUN are driven by sections of U16. None of these bus signals is ever third-stated.

The Baud-rate oscillator and the system-clock oscillator are independent oscillators contained in U12. The system-clock oscillator runs at 8.0 MHz, and is divided down to 4.0 MHz by flip-flop U5A before delivery to the microprocessor through the A-B jumper, and bus line 7 through gates U10C and U10D. The Baud-rate oscillator runs at 9.216 MHz and is divided by a factor of 15 before delivery at 614.4 KHz to bus line 44. The divide-by-15 circuitry is composed of flip-flops U14 and U17, and gate U10A. The square-wave output of the divider is driven onto the bus through U16E.

The bus lines C0, C1, C2, WAIT, CPREQ, DMAREQ, RESET, SKIP and RUN/HALT are all input control or request lines to the CPU. These are all asserted, at various times, by logic on the control-panel, memory, or device-interface modules. The assertion level of all these lines is a TTL low; they are normally held in the high state by pull-up resistors in R11, R12, or R13.

Pin 8 on the microprocessor is the peripheral-device interrupt-request input. It has an active-low assertion level. Thus, interrupt requests can be prevented from propagating through to the microprocessor by asserting the INTDIS bus line to a logic low, due to the action of gate U13C. This is often done for brief intervals by the Interrupt-Inhibit Flip-Flop (IEFF) on the 12540 Memory Extender module, when this module is present in the system (see

Section 7.0). Interrupt requests from peripheral devices are routed to gate U13C through gate U10B, which acts as an inverter.

The IM6100 can be clocked by the on-board oscillator U5 or, in SNGL CLK (single-clock) mode, from a front-panel toggle switch. The former source is selected when the bus line FREERUN is high; the latter when FREERUN is low. The synchronizing flip-flop U8B prevents clock "slivers" from reaching the microprocessor when switching from one clock source to the other.

The multiplexer device U22 is used to allow the user's main-memory program to send data to/from control-panel memory, and vice versa. When the CPMEM bus line (line 47) is high (its normal unasserted state), all assertions of pin 37 on the microprocessor are routed to the MEMSEL bus line, and all assertions of pin 38 are driven onto the CPSEL bus line. This is the IM6100 microprocessor's normal mode of operation. When CPMEM is driven low (by U1B, for example), however, assertions of pins 37 and 38 while DATAF is high will be driven onto the CPSEL bus line. This allows main-memory and control-panel-memory programs to indirectly access (i.e., send data to and retrieve data from) control-panel memory at the user's pleasure.

The processor IOT instruction CAF (6007_8) is implemented on the 12510 module using a few gates and a section of quad-D flip-flop U6. The decoder device U15, and gates U7A, U9C and U3A decode the CAF when it is fetched from memory, and the output of U3A brings pin 5 of U6 high. U6 is clocked by U23A during the time in each instruction-fetch cycle when U3A's output is valid, so if a CAF is being fetched, pin 6 of U6 will be clocked low. Then in the succeeding IOTA cycle, while DEVSEL is asserted and XTC is high, U3C will deliver a drive pulse to the base of Q1, and the RESET bus line will be briefly pulled low by Q1's collector. This implements the CAF operation.

5.1.2 SPECIALIZED INSTRUCTIONS FOR THE 12510 CPU MODULE

Seven specialized instructions are implemented on the 12510 module to provide some additional functions to the system which increase the computer's general utility and take advantage of some of the microprocessor's unique capabilities. These seven instructions are described in Table 5-1:

(next page)

MNEMONIC	OCTAL	DESCRIPTION
CPEN	61Y1	Enable all succeeding indirect operands to come-from or go-to control-panel memory, rather than main memory. This condition remains in force until the next MEMEN instruction is executed.
MEMEN	61Y2	Restore normal operation of the microprocessor; all indirect operands shall come-from or go-to main memory.
CPRQ	61Y3	Force entry to control-panel memory immediately after execution of this instruction.
SKCPR	61Y4	Skip the next instruction if the CPRQ flip-flop is set, and clear the flip-flop.
HISP	61Y5	Cause the micropocessor clock signal to double in frequency to 4.608 MHz during all succeeding DATAF cycles. This condition remains in force until the next LOSP instruction is executed.
LOSP	61Y6	Restore the microprocessor clock frequency to 2.304 MHz in all succeeding cycles, including DATAF cycles.
VCTR	61Y7	Cause the 6101 PIE device which produced the current interrupt, if any, to generate its interrupt vector.

TABLE 5-1
Instruction Set for 12510 CPU Module

Most of the logic in the upper-right sector of Drawing 12512 is used to implement the specialized functions of the 12510 CPU module described in Table 5-1. These functions are implemented through the seven special IOT instructions, 61Y1-61Y7; Y being the (selectable) least-significant octal device-code digit, which is decoded by device U15. The most-significant six bits of the instruction are decoded by gate U9B and the enable inputs on U15. If the most-significant nine bits of the instruction are true, the output of gate U7B will be high. The quad-D flip-flop U6 is clocked by gate U23A during every instruction-fetch cycle, on the trailing edge of XTA. If one of the special IOT's 61Y1-61Y6 is being fetched from memory, then (since pin 4 of U7B is high) pin 10 of U6 will be clocked high, and the one output of U2 (pins 9-14) which corresponds to the specific instruction will be asserted low at DEVSEL

time during the following IOTA cycle.

Flip-flop U4A and gate U1B implement the control-panel-memory access ability, described above. When the CPEN ($61Y1_8$) instruction is executed from main memory, flip-flop U4A will get set, enabling gate U1B to pull the CPMEM bus line low. With U4A set, every indirect access to memory will go-to/come-from control-panel memory (i.e., the instruction itself and the pointer will come from main memory, but the operand will come-from or go-to control-panel memory while the DATAF line is asserted high). This condition remains in force until a MEMEN ($61Y2_8$) instruction is executed, which causes flip-flop U4A to get reset. Note that the CPEN and MEMEN instruction pair is meant to be used by main-memory programs only; gate U1B is disabled for a period of one instruction after each assertion of the CPSEL bus line during a IFETCH cycle. This action is implemented by gate U18D and pin 15 of U6, and prevents assertion of the CPMEM bus line by U1B during (asynchronous) TIMER-generated control-panel interrupts.

Flip-flop U4B is used to implement the special CPRQ instruction ($61Y3_8$). When the CPRQ is executed, flip-flop U4B gets set, and gate U1A pulls the CPREQ bus line low. This forces an entry to control-panel memory on the very next cycle after execution of the CPRQ is completed. In the control-panel routine, the SKCPR ($61Y4_8$) instruction can then be used to skip an instruction if U4B is set, allowing a determination to be made that it was the CPRQ instruction that caused the entry to control-panel memory. Execution of the SKCPR also resets U4B if it is set. This logic allows software "traps" to control-panel memory, where hidden routines may reside which the user may wish to implement without using space in main memory.

The special instructions HISP ($61Y5_8$) and LOSP ($61Y6_8$) are used to aid in testing of memory. It is a peculiarity of the 6100 microprocessor that it requires faster memory speed while fetching instructions than it does when indirectly accessing memory data. Since indirect accesses are the only practical way to test memory, it is necessary to test memory devices at a higher speed than normally required in order to assure that they have an adequate access-time margin to be reliable for use as program memory. This can be brought about, while testing memory, by forcing the microprocessor to run faster during the indirect-execute cycle (DATAF high). Flip-flops U5B and U11, and gates U23B and U7C are used to implement the HISP/LOSP functionality. The flip-flops in U11 are used to divide the 9.216 MHz output of U12 by either

2 or 4. The divide factor is 4 if DATAF is low, and 2 if U5B is set and DATAF is high. To test memory, the normally-installed A-B jumper is moved to A-C, and the F-H jumper is temporarily installed. This causes the microprocessor to run at 2.304 MHz, except during DATAF cycles while U5B is set, when microprocessor speed is 4.608 MHz. During execution of the memory test program, just before an indirect access is made to the memory space under test, a HISP instruction is executed, which causes flip-flop U5B to be set. All subsequent indirect-executes (DATAF high) will then take place at 4.608 MHz, placing the desired speed stress on the memory devices under test. The LOSP ($61Y6_8$) instruction is used to reset U5B before the next indirect access is required to the memory space containing the test program. Note that control-panel interrupts are disabled by gate U7D during the time U5B is set, to protect against asynchronous TIMER-generated requests.

The VCTR instruction ($61Y7_8$) is used to enhance the vectoring capability of I/O modules which employ the 6101 PIE device. Normally the PIE devices generate their interrupt vector when the first IOT instruction is generated after the INTGNT (interrupt-grant) bus line goes high. However, some additional ease in programming interrupt routines can be brought about if vectoring is delayed until a special instruction for this purpose is executed. That is the function of the VCTR instruction, which is implemented using flip-flop U8A and gates U3B, U18A, U23C and U16B. Normally flip-flop U8A is in the reset state (pin 5 low). However, as soon as a device interrupt is recognized and the INTGNT bus line is pulled high, pin 4 of U16 goes low, putting U8A in the set state. When the VCTR instruction is fetched from memory, it is decoded by U3B and pin 2 of U6 is clocked high. This causes gate U18A to assert the INTGNT* bus line, which allows the appropriate PIE device receiving its pin 2 (INTGNT) input from the INTGNT* line to generate its interrupt vector. On the very next instruction (which will not be a VCTR), U3B's output will be low, and pin 3 of U6 will clock U8A reset, pulling INTGNT* back low. This comletes the actions associated with the VCTR instruction.

For a system including the 12540 programmable real-time clock function, as well as PIE-driven I/O interfaces and DEC-compatible (non-PIE) I/O interfaces, interrupts are prioritized into three groups. The highest priority is the real-time clock, whose INTGNT input (pin 39 on the 6102 device) comes from the INTGNT bus line. This device will vector off the first IOT of any type executed after the interrupt is recognized and the INTGNT bus

line goes high. An IOF instruction (6002_8) can be used for this purpose. The next highest priority devices are the PIE-driven modules, which should receive their INTGNT input from the INTGNT* bus line. These will vector when the special VCTR instruction is executed. The lowest-priority group is the non-vectoring DEC-compatible I/O devices, where the normal DEC-style skip chain execution is required to determine which device generated the interrupt. (Of course, if desired, one or more skip-type devices may be given a higher priority than the PIE-driven devices by attempting a skip off their interrupt flag before executing the VCTR instruction.) A typical entrance to an interrupt, then, may resemble the following:

```
DCA SAVAC       /SAVE THE ACCUMULATOR CONTENTS.
GTL             /GET THE LINK,
DCA SAVLNK      /AND SAVE IT.
IOF             /VECTOR TO CLOCK ROUTINE IF NECESSARY.
GTF             /GET THE FLAGS,
DCA SAVFLG      /AND SAVE THEM.
.
.
.
/DO ANY OTHER REQUIRED OPERATIONS TO SAVE THE MACHINE'S STATE.
.
.
.
VCTR            /SEE IF ANY PIE DEVICE VECTORS.
SKDEV1     /IF NOT, TRY SKIPPING OFF FIRST NON-VECTORED DEVICE.
SKP             /IF NOT THIS ONE TRY NEXT ONE.
JMP I DEV1 /GO TO SERVICE ROUTINE FOR FIRST NON-VECTORED DEVICE.
SKDEV2          /TRY NEXT (LAST) NON-VECTORED DEVICE.
SKP
JMP I DEV2 /GO TO SERVICE ROUTINE FOR LOWEST-PRIORITY DEVICE.
.
.
.
/RESTORE MACHINE STATE.
.
.
.
ION     /TURN INTERRUPTS BACK ON (AFTER NEXT INSTRUCTION).
RETURN  /RETURN = JMP I ZERO, OR POPJ WITH 12540 STACK MODULE.
```

The instructions after the JMP I DEV2 above are used to provide an orderly return to the main-line program in case the previous instructions fail to detect which device caused the interrupt. This can happen if the clock requests an interrupt (pulling the 12540 module's PROUT line low, and thus

disabling all PIE devices from vectoring) between the time the IOF and VCTR instructions are executed when the original interrupt was caused by a PIE. The result would be that the VCTR instruction would not cause the interrupting PIE to vector, and execution would then return to the main program. Since the clock interrupt request would then still be pending, another interrupt would immediately take place, and the clock would get service. This would be followed by another interrupt, which would service the PIE which had requested the original interrupt. The instructions after the JMP I DEV2 also protect the system from a noise-induced interrupt; since the interrupting device could not be determined, execution would simply return to the main-line program.

5.1.3 HARDWARE SETUP OF 12510 MODULE

The 12510 CPU module contains a few jumper-pad groups intended for selecting the operating mode of the module. These jumper groups are discussed in the next few paragraphs.

The A-B-C jumper-pad group is used to select the microprocessor's clock speed. For normal operation, a jumper strap is soldered from pad A to pad B, and the microprocessor runs at 4.0 MHz. For memory testing, however, the A-B jumper is omitted, and a (temporary) strap is installed instead from pad A to pad C. This causes the microprocessor to be clocked at 2.304 MHz. (Also see discussion of F-H jumper, below.)

The D-E jumper is normally omitted, so that the CPEN, MEMEN, CPRQ and SKCPR instructions are available to the programmer. If this jumper is installed, however, these four instructions are permanently disabled. This might be desireable in systems which will run DEC-written software only, where none of these four instructions will ever occur.

The F-H jumper is normally omitted, which disables the HISP and LOSP instructions. When testing memory, however, this jumper shoud be temporarily installed, so that HISP and LOSP can be included in the memory-test software.

The R-T-U-V-W-X-Y jumper pad group is used to select the least-significant device-code digit for the seven specialized 12510 instructions described in Table 5-1. To select device-code 12_8 (the normal device-code used for these instructions), place a jumper from pad Y to pad T. Other possible device codes are 11_8 (jumper Y-R installed), 14_8 (jumper Y-U installed), 15_8 (jumper Y-V installed), 16_8 (jumper Y-W installed), and 17_8 (jumper Y-X installed).

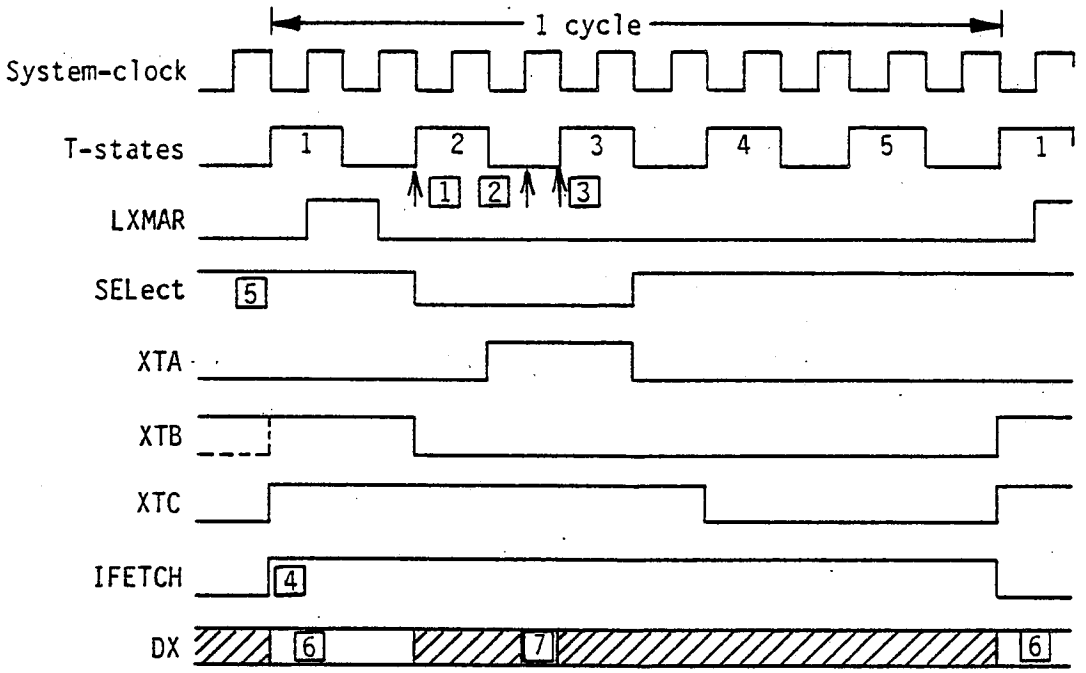
5.2 CPU TIMING

The timing for the most fundamental CPU lines is illustrated in Figures 5-1 and 5-2. The T-state square wave shown is internal to the IM6100; it is not available externally as a timing reference. However, all machine timing is derived from this waveform, so it is an important reference point in CPU timing discussions. Note that the frequency of this waveform is one-half the frequency of the CPU clock.

All machine cycles are composed of either five or six T-states (often referred to simply as "states"). Cycles which do not involve a "write" operation consist of five states (Figure 5-1). When a "write" is called for, the cycle is extended to six states (Figure 5-2).

Each instruction requires 2, 3 or 4 cycles to be fetched from memory and completely executed. The first cycle is always an instruction-fetch cycle consisting of five states. The remaining cycles can consist of either five or six states each. Thus, a complete fetch-and-execute can consist of 10, 11, 15, 16, 17, 21 or 22 states. Table 4-1 details the number of cycles, and states, in each type of instruction.

An instruction fetch-and-execution begins with an instruction-fetch cycle which looks like that shown in Figure 5-1. The IFETCH line is asserted high throughout the duration of this cycle to indicate an instruction is being fetched. The CPU puts the address of the instruction on the DX lines throughout the first T-state. This address is then latched into memory by using the trailing edge of the LXMAR pulse. Next, the MEMSEL line is asserted by the CPU to allow the selected memory devices to drive the DX lines with the instruction data from the addressed location. This data is picked up from the DX lines by the CPU on the rising edge of T3. The rest of the cycle is then used by the CPU for internal operations.



1 - Request lines RESET, CPREQ, DMAREQ, INTREQ and RUN/HALT flip-flop (internal to IM6100) are sampled at this point in last execute-cycle of each instruction. If no overriding actions are requested by these lines, CPU will fetch an (the next) instruction in the next cycle.

2 - CPU samples the WAIT line at this point. If it is found asserted, the CPU will extend the T2 state an integral number of system-clock cycles until, sampling the WAIT line once each system-clock cycle, it finds the WAIT line has returned to the non-asserted state. Processing will then continue at the point in the cycle where the WAIT began.

3 - CPU reads DX lines at this point (allow at least 100 ns set-up time).

4 - If the cycle is an instruction-fetch, IFETCH will be asserted throughout the cycle.

5 - In most cycles either the MEMSEL, CPSEL or SWSEL line will asserted, depending on whether the cycle involves main-memory, control-panel memory, or the front-panel Switch Register, respectively.

6 - CPU puts address or instruction (whichever is appropriate for the given cycle) onto the DX lines during this time. This information is latched into memory modules and peripheral interfaces by the trailing edge of the LXMAR pulse.

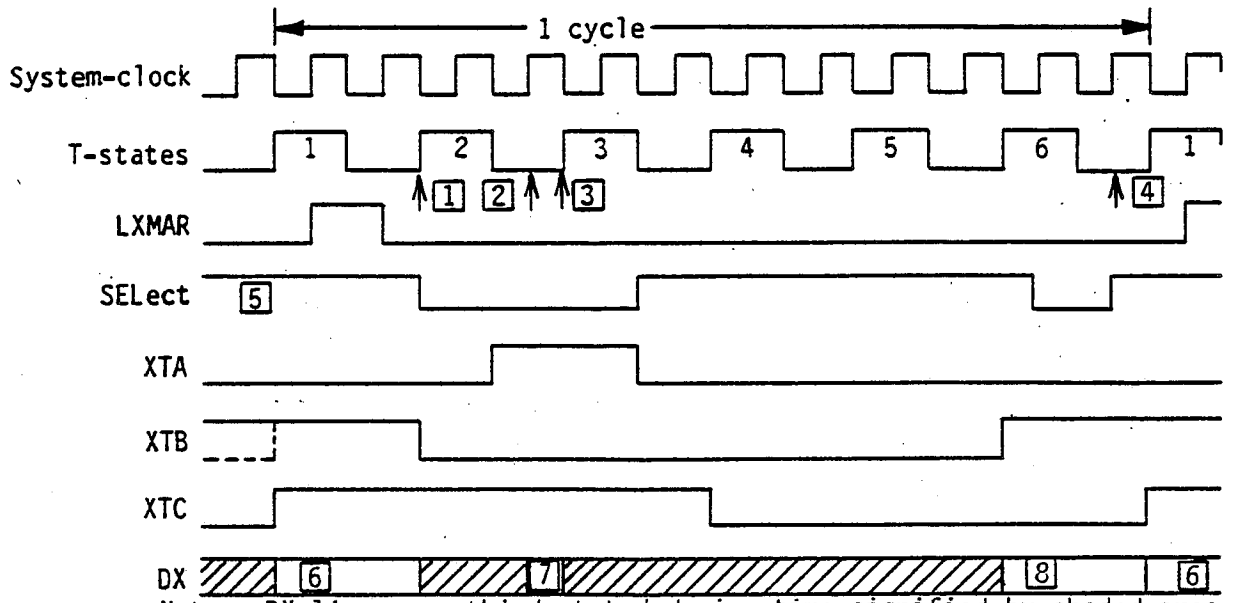
7 - Memory or Switch Register (whichever is selected by SELect line) puts data to be read by CPU onto DX lines during this period of time.

Figure 5-1

Basic 5-state Cycle Timing

The next cycle may be the first (and possibly only) execute cycle, or it may be an "indirect" cycle. The latter type of cycle is entered when the instruction fetched is an indirectly-addressed memory-reference-instruction (MRI). (For a discussion of instruction types, memory organization, and the distinction between direct and indirect addressing, see Section 4.0.) If the instruction is auto-indexed, the indirect-cycle will consist of six states; otherwise an indirect-cycle has just five states. Execute cycles may consist of either five (MRI and Operate instructions) or six (MRI and IOT instructions) states. Some instructions require one, and some two, execute cycles.

The six-state cycle shown in Figure 5-2 is similar to the five-state cycle, except that the cycle has been extended one state so that the CPU can write data into memory or a peripheral device. To accomplish this operation the CPU puts the data on the DX lines throughout state T6. A SELect line (MEMSEL, DEVSEL or CPSEL) is then asserted by the CPU to actuate the "write" operation. The data is normally strobed into the memory or peripheral device on the trailing edge of the SELect pulse. When the SELect line is asserted by the CPU, the logic in the memory or device interface differentiates between a "read" and "write" operation by monitoring the XTC line. When this line is high, an assertion of a SELect line calls for a "read", and when low a "write". Note that every "write" operation is preceeded by a "read", which in most (but not all) cases is ignored by the CPU. During an auto-indexed indirect-cycle, however, the CPU does use the information picked up in the "read" part of the cycle, to determine the "pointer" address. And during the execute cycle of an ISZ instruction a meaningful "read" is performed to pick up the the data in the addressed location just before it is incremented.



1 - Request lines RESET, CPREQ, DMAREQ, INTREQ and RUN/HALT flip-flop (internal to IM6100) are sampled at this point in last execute-cycle of each instruction. If no overriding actions are requested by these lines, CPU will fetch an (the next) instruction in the next cycle.

2 - CPU samples the WAIT line at this point. If it is found asserted, the CPU will extend the T2 state an integral number of system-clock cycles until, sampling the WAIT line once each system-clock cycle, it finds the WAIT line has returned to the non-asserted state. Processing will then continue at the point in the cycle where the WAIT began.

3 - CPU reads DX lines at this point (allow at least 100 ns set-up time). CPU also reads state of C0, C1, C2 and SKP lines at this point in IOTA cycles.

4 - CPU samples the WAIT line again at this point. If WAIT is found asserted, the T6 state will be extended an integral number of system-clock cycles.

5 - In most cycles either the MEMSEL, DEVSEL or CPSEL line will be asserted, depending on whether the cycle involves main-memory, an external device, or the control-panel memory, respectively.

6 - CPU puts address or instruction (whichever is appropriate for the given cycle) onto the DX lines during this time. This information is latched into memory modules and peripheral interfaces by the trailing edge of the LXMAR pulse.

7 - Memory module or device interface (whichever is addressed) puts data to be read by CPU onto DX lines during this period.

8 - CPU puts data to be written into a memory location or device interface onto the DX lines during this period.

Figure 5-2

Basic 6-State Cycle Timing

5.2.1 DETAILED CPU TIMING

While the timing diagrams given in Figures 5-1 and 5-2 adequately detail the relationships between several of the time-critical lines on the CPU device, it should not be inferred that all edges of the illustrated signals are perfectly coincident in time, as shown. When the user designs custom interfaces for the PCM-12 (or the IM6100 as a stand-alone processor), he must be aware of the small timing differentials that appear between the various CPU signals. For this purpose he should consult the Intersil data sheet on the IM6100 device. Figure 5-3 shows some typical timing differentials between the illustrated signals, as measured on the PCM-12 Omega backplane bus. Because of the point of measurement, these figures include the delays caused by the bus-interfacing logic.

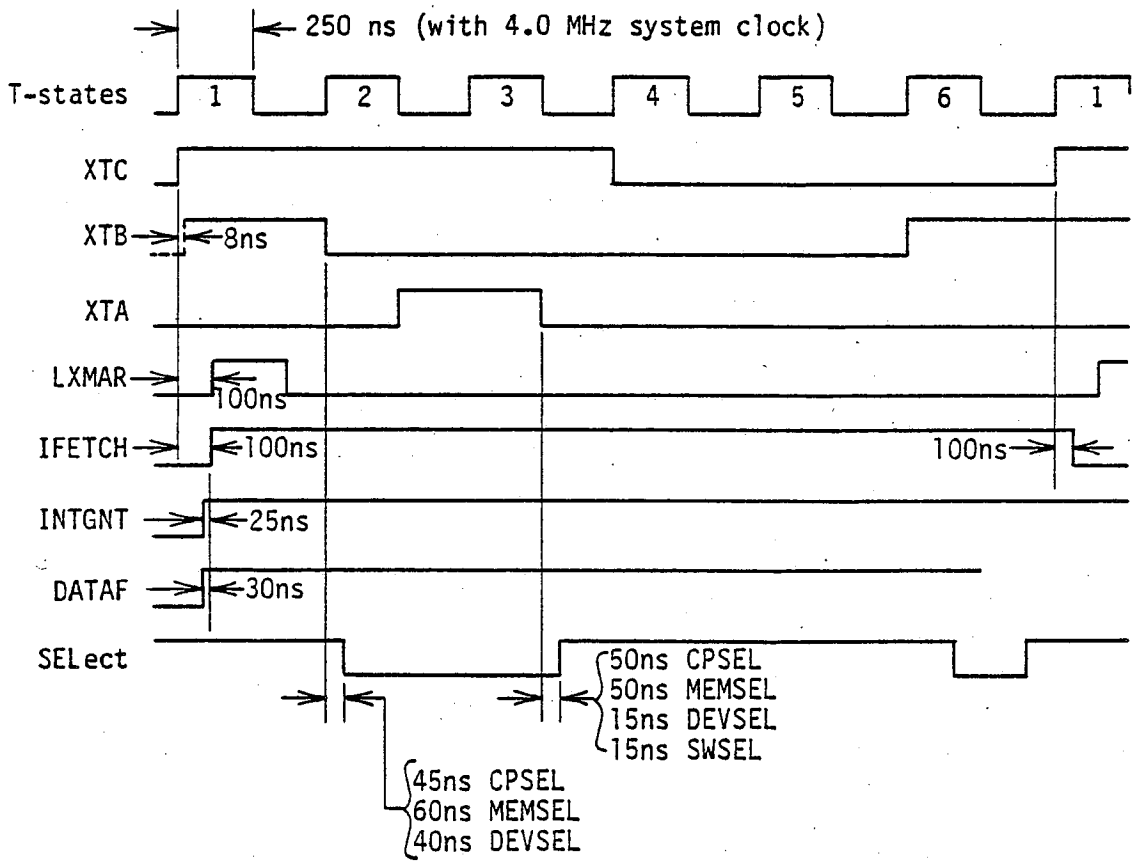


Figure 5-3
CPU Timing Differentials

Note that the first signal to appear in each cycle is the rising edge of XTC. Therefore this signal is used as a reference from which several other delay measurements are made. (The timing lines XTA, XTB, and XTC are actually used by the IM6100 to develop the other signals shown, so the latter are bound to be delayed with respect to the 'X' lines.) Particularly noteworthy is the delay on the IFETCH line. This line does not come to the asserted state until 115 nanoseconds (ns) after it is expected, according to Figure 5-1. And, because it is so slow, it actually overlaps into the next cycle before again going low. A similar phenomenon occurs with DATAF, although to a lesser extent (70 nanoseconds). These delays however, are not detrimental, since these lines are not normally used for critical timing operations.

5.3 PROCESSOR OPERATION PRIORITIES

As indicated in Figures 5-1 and 5-2, the IM6100 samples the RESET line, the request lines CPREQ, DMAREQ, and INTREQ, and the state of its internal RUN/HALT flip-flop on the rising edge of T2 during the last execute cycle of each instruction, to determine what it should do next. If any of the request lines is asserted, or if the RUN/HALT flip-flop has gone to the HALT state, or if the RESET line is asserted, the CPU will perform the requested operation next. In the case of two or more of these conditions existing at the time of the sample, the CPU will perform the requested operation having the highest priority, as listed below. If none of these conditions exist, the CPU will fetch and execute the next sequential instruction, and again sample the request lines, etc.

The priorities, from highest to lowest, are:

RESET — If the RESET line is asserted low at the sample time, the CPU immediatly sets its Program Counter to 7777_8, clears the Accumulator and Link, and puts the CPU in the HALT state. While halted the CPU continues to cycle and generate the timing signals XTA, XTB and XTC.

CPREQ — If the RESET line is not found to be asserted, but the CPREQ line is, the CPU grants the control panel request at the end of the current instruction. See Figure 5-8.

RUN/HALT — If neither the RESET line nor the CPREQ line is asserted, but the CPU finds its internal RUN/HALT flip-flop in the HALT state, it enters the HALT state at the end of the last

execute cycle of the current instruction. While halted the CPU continues to generate timing signals XTA, XTB, and XTC.

DMAREQ — If none of the aforementioned conditions exist, and the CPU finds the DMAREQ line asserted, it grants the DMA request at the end of the current instruction. See Section 5.5.

INTREQ — If neither RESET, control-panel request, HALT, nor DMA action is indicated, and the INTREQ line is found asserted, the CPU will grant the interrupt request at the end of the current instruction. See Section 5.4.

IFETCH — If none of the above actions is indicated, the processor will fetch the next sequential instruction.

The above priority heirarchy is supplemented by internal and external logic and by program software. For example, when the CPU has begun executing a device-interrupt routine (INTGNT high), or is waiting for a DMA action to be completed, control-panel interrupt requests are inhibited by gates on the 12530 control-panel printed-circuit board. During the processing of a control-panel interrupt, device-interrupt requests and DMA requests are ignored by the CPU. When the CPU grants a device-interrupt request, it ignores further interrupt requests until the interrupt system is re-enabled by an ION instruction.

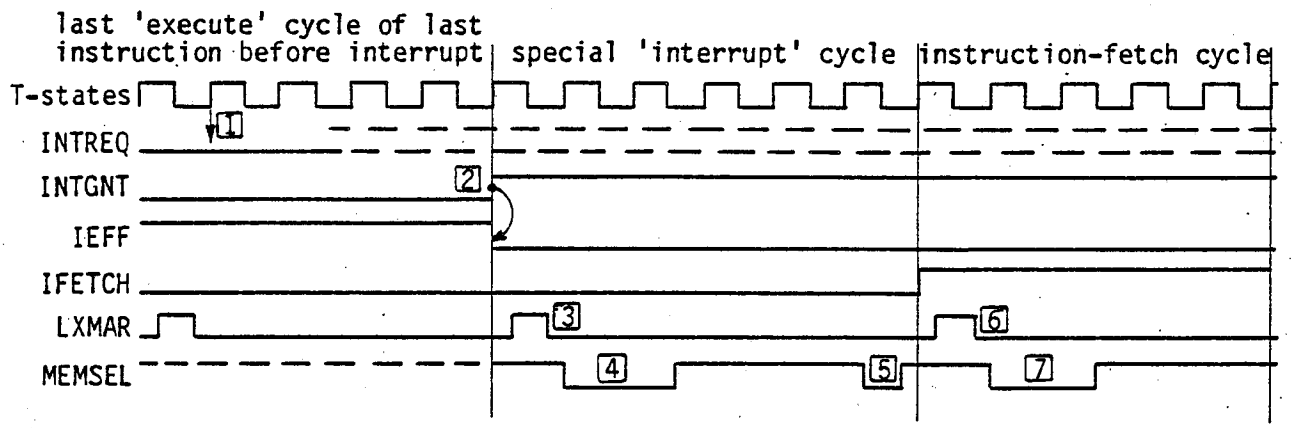
5.4 DEVICE INTERRUPT TRANSFERS

The program-interrupt method is used to transfer data between the CPU and peripheral devices when it is unacceptable to have the CPU wait for the device to indicate that it is ready to output or accept a new data transfer. Using the interrupt system, the CPU is free to go about execution of the "background" program until the external device indicates it is ready for a transfer by requesting an interrupt. This avoids the need to put the processor into a "wait-loop".

An external device requests an interrupt by asserting the INTREQ bus line (line 24) to the low state. If no higher priority operation (viz., RESET, control panel request, HALT, or DMA request) is active when the processor finishes executing the current instruction, the machine will grant the interrupt request at that time if the interrupt system is enabled. The interrupt system is enabled whenever the Interrupt Enable Flip-Flop (IEFF) in the IM6100 is set. [Note, the interrupt system is considered to be disabled when the Interrupt Inhibit Flip-Flop (IIFF) on the 12540 Memory Extender

module is set (causing the INTDIS bus line (line 28) to be asserted low), as this prevents interrupt requests from reaching the IM6100 CPU device.]

The timing diagram for an interrupt request/grant is shown in Figure 5-4. In the first cycle after an interrupt is granted, the CPU stores the current contents of the Program Counter in memory location 0000_8. [This location holds the "return address" the computer needs so that it can return to where it left the main-line program at the end of the interrupt service routine.] Then, in the next cycle, the machine fetches the first instruction in the interrupt routine from location 0001_8.



1 - Interrupt-Request (INTREQ) line is sampled at point shown in last execute-cycle of previous instruction, and found asserted low. Interrupt will be granted at beginning of next cycle.

2 - Interrupt granted, and Interrupt-Enable Flip-Flop (IEFF) reset.

3 - Address 0000_8 (Field 0_8) latched into memory.

4 - "Don't care" read.

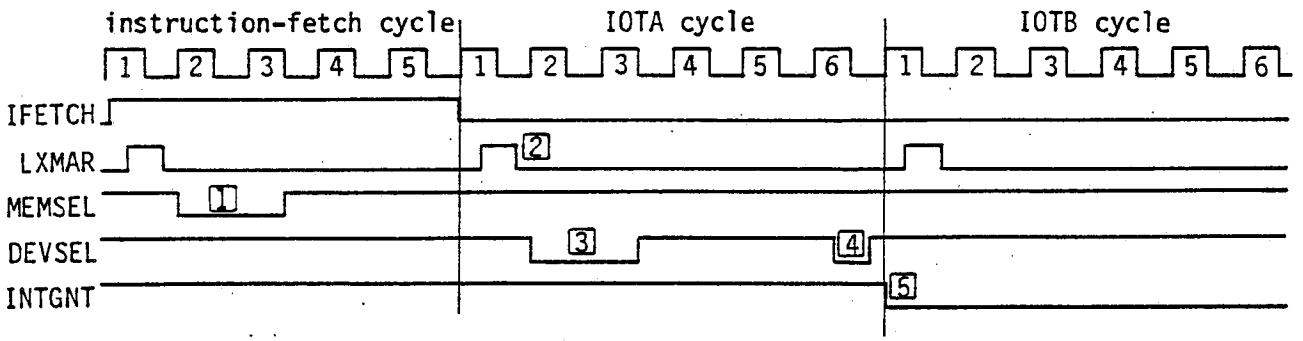
5 - Program Counter is written into location 0000_8 (stores return address).

6 - Address 0001_8 latched into memory (still Field 0_8).

7 - Fetch first instruction in interrupt routine.

Figure 5-4
CPU Interrupt Request/Grant Timing

The positive-going edge (effectively) of the INTGNT line is used by the 12540 Memory Extender module to reset the EMA0 - EMA2 lines (lines 13, 15 and 17) on the bus to the low state. This causes the initial instruction(s) in the interrupt service routine to be fetched from memory field 0, in accordance with PDP-8 conventions. The INTGNT line is reset to the low state midway through execution of the first IOT instruction after the interrupt. See Figure 5-5, the timing diagram for the resetting of the INTGNT line.



1 - First IOT instruction, after interrupt granted, fetched from memory.

2 - IOT instruction latched into device interfaces.

3 - Data transfer to CPU from peripheral devices.

4 - Data transfer to peripheral devices from CPU.

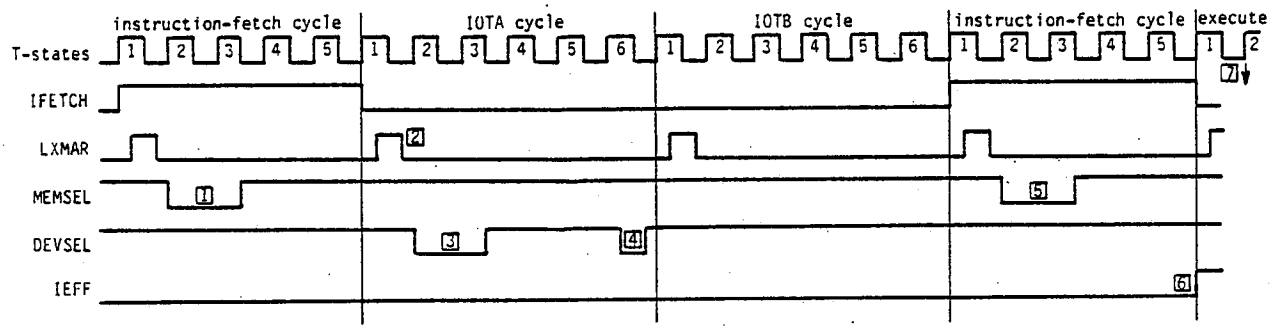
5 - INTGNT line reset at end of IOTA cycle.

Figure 5-5
INTGNT Reset Timing

When an interrupt is granted, the IEFF is reset. It is not set again until an ION or RTF instruction is executed. This gives the processor time to do its house-keeping chores before allowing another interrupt to be recognized. Typically, these chores involve executing a skip-chain (or vectoring) to find which device caused the interrupt, and storing the Accumulator, Link, and Program Counter for restoration later. If nested interrupts are allowed, the Accumulator, Link, and Program Counter need to be saved on a "push-down stack". Interrupt programming, including software-prioritized multiple interrupts, is covered in Chapter 6 of DEC's Introduction to Programming.

Also refer to Section 5.1.2 above for suggestions on programming interrupts onthe PCM-12 system. A timing diagram for setting the IEFF with an ION (or RTF) instruction is shown in Figure 5-6. Note that the IEFF is not actually set until the processor has fetched the next instruction after the ION/RTF; this guarantees one more instruction will be executed after an ION/RTF before the next interrupt can be recognized.

The above discussion largely applies to operation of the PCM-12 system with normal PDP-8 "style" software-prioritized interrupts. However, the PCM-12 also provides for a hardware-prioritized interrupt system employing the IM6101 Parallel Interface Element (PIE) device. Operation of interrupts using this hardware is discussed in section 5.1.2 above, and in the later sections of this manual which cover the specific accessory modules utilizing the PIE device.



1 - Fetch ION instruction from memory.

2 - Latch ION instruction into device interfaces.

3 - "Don't care" device-read.

4 - "Don't-care" device-write.

5 - Fetch next instruction from memory.

6 - IEFF is set (interrupts now enabled).

7 - First active sampling-time on Interrupt-Request (INTREQ) line.

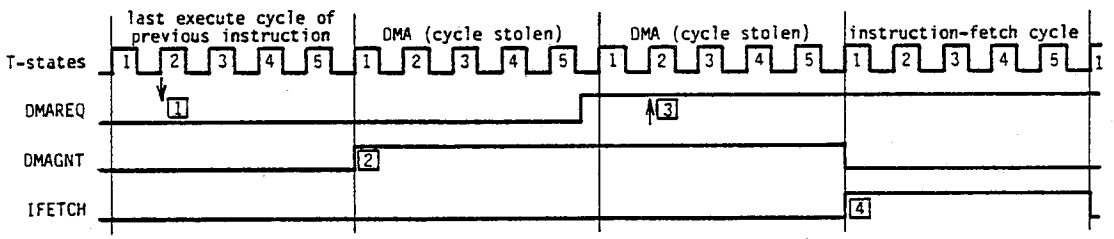
Figure 5-6
Interrupt-Enable Flip-Flop Set Timing

5.5 DIRECT MEMORY ACCESS

Very fast peripheral devices, such as conventional disk memories, usually transfer their data to and from the computer using direct-memory-access (DMA). In DEC manuals this form of transfer is also called "data-break". The PCM-12 system provides for DMA, on a cycle-stealing basis. See Figure 5-7.

DMA port logic in a peripheral device interface requests DMA action by asserting the DMAREQ bus line (line 22) to the low level. This line is sampled by the CPU early in the last execute cycle of each instruction. If the CPU finds the DMAREQ line asserted, and there are no higher priority requests active (viz., RESET, control panel request, or HALT), the CPU suspends program execution at the beginning of the next cycle, and asserts the DMAGNT line on the bus.

When the DMAGNT line is asserted, the CPU tri-states all bus control-lines with which it normally drives the memory modules: all 12 DX lines, MEMSEL, XTC, LXMAR and DATAF. (Timing lines XTA and XTB continue to be driven by the CPU.) Also, the 12540 Memory Extender module third-states the EMA0-EMA2 bus lines. The DMA port logic can then use these lines to read or write data to/from any main memory location. (When the XTC line is third-stated, it is pulled high by resistors on the CPU and memory modules to prevent an inadvertent memory "write".)



1 - DMAREQ line sampled in last execute-cycle of previous instruction, and found asserted low.

2 - DMAGNT line asserted by CPU.

3 - DMAREQ line sampled by CPU, and found to have been released. DMA action will end at conclusion of current cycle.

4 - Program execution resumes with instruction-fetch cycle.

Figure 5-7
DMA Timing

During each machine cycle of a DMA action, the CPU continues to sample the DMAREQ line on the rising edge of T2. When it finds this line to have been released by the DMA port, the CPU will resume execution of the main-line program in the next cycle. Figure 5-7 shows the case where two cycles have been "stolen", but any integral number of cycles can be used by the DMA port in a single DMA action.

DMA transfers actually entail some fairly complex cooperation between program software and DMA-port logic. If a DMA type of peripheral device is present in the system, see the section of this manual covering the interface for that device for a more detailed discussion of how that device transfers data using the DMA method.

In conclusion it should be noted that the DMAREQ line can also be used by external devices as a level sensitive "pause" line. Asserting the DMAREQ line will simply cause the CPU to suspend program execution for an integral number of cycles until the line is released. Since the CPU continues to drive the PCM-12 bus with the timing signals XTA and XTB during a DMA or "pause" action, either of these pulses may be counted to determine the exact number of cycles during the pause.

5.6 CONTROL PANEL INTERRUPTS

Due to the limited number of pins available on a practical semiconducter package, the IM6100, like every microprocessor, does not provide continuous real-time access to many of its internal registers. The state of the Accumulator, Link, Program Counter, MQ register, etc. are multiplexed at various times on the CPU DX lines. To find the state of one of these registers, it is necessary to temporarily suspend mainline program execution, and execute a special control panel routine to bring the required data out of the 6100 device and latch it into external registers or indicators. Thus, it is said that the control panel is "implemented in software".

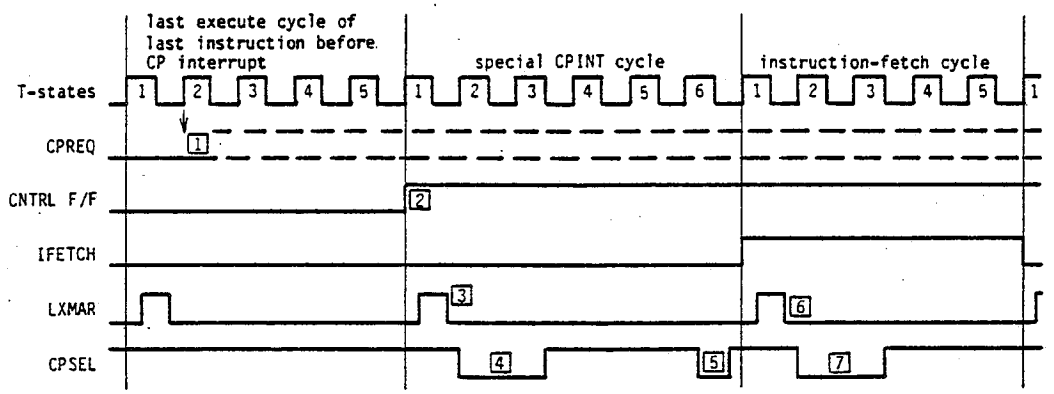
Detailed operation of the 12530 control-panel logic is discussed in Section 2.6 of this manual. That section should be consulted as a supplement to the discussion below.

A control panel interrupt is requested by asserting the CPREQ bus line (line 66) low. The CPU will grant the control panel interrupt request at the end of the current instruction, if a RESET action is not simultaneously being

requested. A control panel interrupt request will be granted even if the machine is in the HALT state; the CPU will be forced into the RUN state for the duration of the control panel interrupt routine, and then return to the HALT state.

During the processing of a control panel interrupt, the CPU ignores DMA and device-interrupt requests. It also ignores further control-panel interrupt requests.

The control panel interrupt system is not affected by the CPU's Interrupt Enable Flip-Flop (IEFF); this logic applies only to device interrupts. Furthermore, the processor IOT instructions ION and IOF do not affect the control panel interrupt system. (In fact, the instructions ION and IOF, if executed during a control panel interrupt, do not even affect the device-interrupt system. IOF is totally disabled during control-panel interrupts. ION is used for a special purpose, as described below.)



1 - CPREQ line is sampled at point shown in last execute-cycle of last instruction, and found asserted low. Control-panel interrupt will be granted at the end of current instruction.

2 - When the CP interrupt is granted, the CNTRL F/F internal to the CPU is set.

3 - Address 0000_8 is latched into the control-panel memory.

4 - "Don't care" read.

5 - Program Counter contents are written into CP memory location 0000_8.

6 - Adress 7777_8 is latched into the control-panel memory.

7 - The first instruction in the CP interrupt routine is fetched from control-panel memory location 7777_8.

Figure 5-8
Control-Panel Interrupt Request/Grant Timing

A control panel interrupt is granted according to the timing shown in Figure 5-8. The CPREQ line is sampled by the CPU at the indicated time in the last execute cycle of every instruction (or, if the machine is halted, the line is sampled every cycle). If the line is found asserted, the CPU grants the control panel request by setting its internal Control-Panel Flip-Flop (CNTRL F/F). While this flip-flop is set, further control-panel requests are ignored, as are DMA requests and device-interrupt requests. Changes to the device-interrupt system by the instructions ION and IOF are also inhibited while the CNTRL F/F is set.

During the CPINT cycle the contents of the CPU's Program Counter are stored in control panel memory location 0000_8. This forms the return address so the CPU can return to the main-memory program at the end of the control-panel service routine. The Program Counter is then jam-set to 7777_8, and the first instruction in the control panel routine is fetched from this control-panel memory location during the next cycle.

During the execution of the control panel interrupt, while the CNTRL F/F is set, the control panel memory, rather than main memory, is selected for all memory reference operations. This is accomplished by the CPU asserting the CPSEL line rather than the MEMSEL line. However, the control panel routine does have access to main memory, through indirectly addressed AND, TAD, ISZ and DCA instructions. When either the TAD or DCA instruction is executed in the indirect address mode, the instruction fetch cycle and the indirect cycle employ the CPSEL line to fetch the instruction and pointer from control panel memory, then use the MEMSEL line during the execute cycle to deposit the data into the selected main memory location with DCA, or fetch data from the main memory location with TAD. Indirectly addressed ISZ and AND instructions operate in an exactly analagous manner. Note that, since these are the same requirements for the DATAF line to be asserted, and for the EMA logic to assert the Data Field rather than the Instruction Field, the control panel routine will access the current "data field". Thus the control panel routine can fetch the data in any main memory location using TAD I, deposit data into any main memory location using DCA I, increment-and-skip on any main memory location using ISZ I, or AND the content of any main memory location into the Accumulator using AND I. (All these operations can access control-panel memory rather than main memory, if desired - see Section 5.1.2.)

The end of the control panel routine is marked by the execution of an ION instruction, followed immediately by an indirect JMP through control-panel memory location 0000_8. The ION instruction has no effect on the device-interrupt system, since the CNTRL F/F is still set when it is executed. However, the ION causes the CNTRL F/F to be reset midway through execution of the next instruction. This next instruction is the aforementioned JMP I, which resets the CPU Program Counter to the address contained in control-panel memory location 0000_8. Normally this will be the stored return address where the CPU left the mainline program to execute the control panel interrupt. However, if the control panel routine modified the content of control-panel memory location 0000_8, the Program Counter will be set to a new starting address upon emergence from the routine. See Figure 5.9.

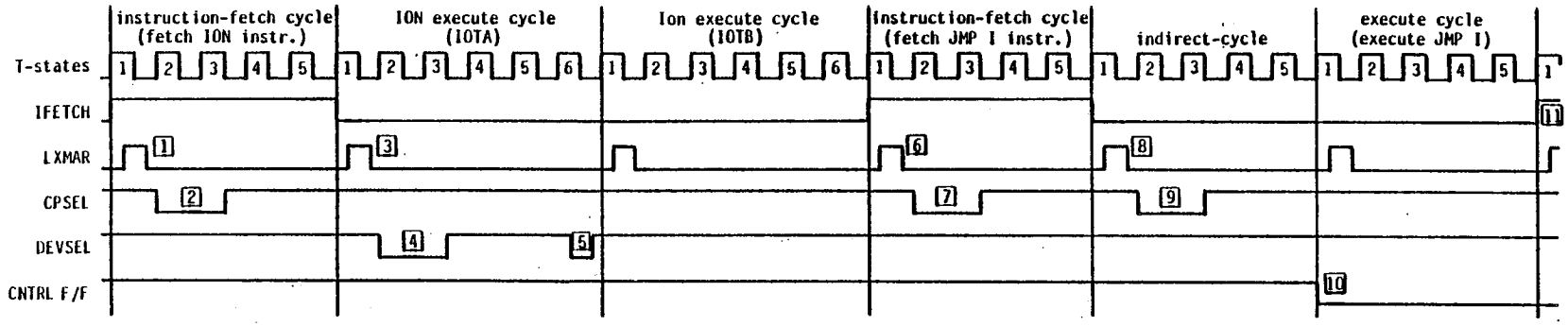
A forced exit from the control panel routine can be achieved by asserting the bus RESET line. RESET can accomplish this forced exit, since it has higher priority than CPREQ, and can override the control-panel routine. Execution of an RTF (6005_8) instruction, like RTF, will reset the CNTRL F/F, and thus can also effectively end the control panel routine.

5.7 RESET ACTION

Raising the RESET switch on the PCM-12 front panel asserts the RESET bus line (line 18) and causes the CPU Accumulator and Link to be cleared, the Program Counter to be set to 7777_8, and the machine to go to the HALT state. When in the HALT state, however, the CPU continues to cycle and produce the timing signals XTA, XTB and XTC. All SELect lines remain unasserted (high). A RESET also clears the Instruction Field and Data Field registers on the 12540 Memory Extender module.

After power-up, the state of the IM6100 is unknown. The IM6100 requires 58 clock cycles to do a complete reset and initialization of all internal states. Once RESET is released, at least 10 clock cycles should occur before the CPU is put in the RUN state by a rising edge on the RUN/HLT input.

Memory for the IM6100 is often organized with non-volatile devices such as PROM's in the high address space of a given memory field. Since the Program Counter is set to 7777_8 and the IF to 0_8 by a RESET, the first instruction to be executed following a RESET (if no other starting address is set up from the front panel) will be fetched from location 7777_8 of Field 0, a



1 - Address of ION instruction is latched into control-panel memory.

2 - ION instruction is read from control-panel memory.

3 - ION instruction is latched into device interfaces (irrelevant).

4 - "Don't care" device-read.

5 - "Don't care" device-write.

6 - Address of JMP I instruction is latched into control-panel memory.

7 - JMP I instruction is read from control-panel memory.

8 - Address 0000_8 is latched into control-panel memory.

9 - Return address is read from location 0000_8 in control-panel memory.

10 - CNTRL F/F is reset midway through JMP I instruction.

11 - Next instruction is fetched from main memory. If machine was halted before CP routine, it returns to HALT state at end of JMP I execution.

Figure 5-9
Control Panel Exit Timing

non-volatile portion of memory. This instruction will typically be a JMP to a power-up routine that can be used to initialize the system.

5.8 CPU RUN/HALT FLIP-FLOP

The IM6100 microprocessor contains an internal RUN/HALT flip-flop. When this flop-flop is in the RUN state, the RUN bus line (line 8) is asserted low and the RUN lamp on the 12530 front panel is lighted. The state of the RUN/HALT F/F is changed by pulsing the RUN/HALT bus line (line 21). This is accomplished by the CONT/SNGL INST switch on the 12530 module. See the detailed discussion of the actions of this switch in Section 2.6. The RUN/HALT F/F is toggled on the rising edge of a pulse on the RUN/HALT line.

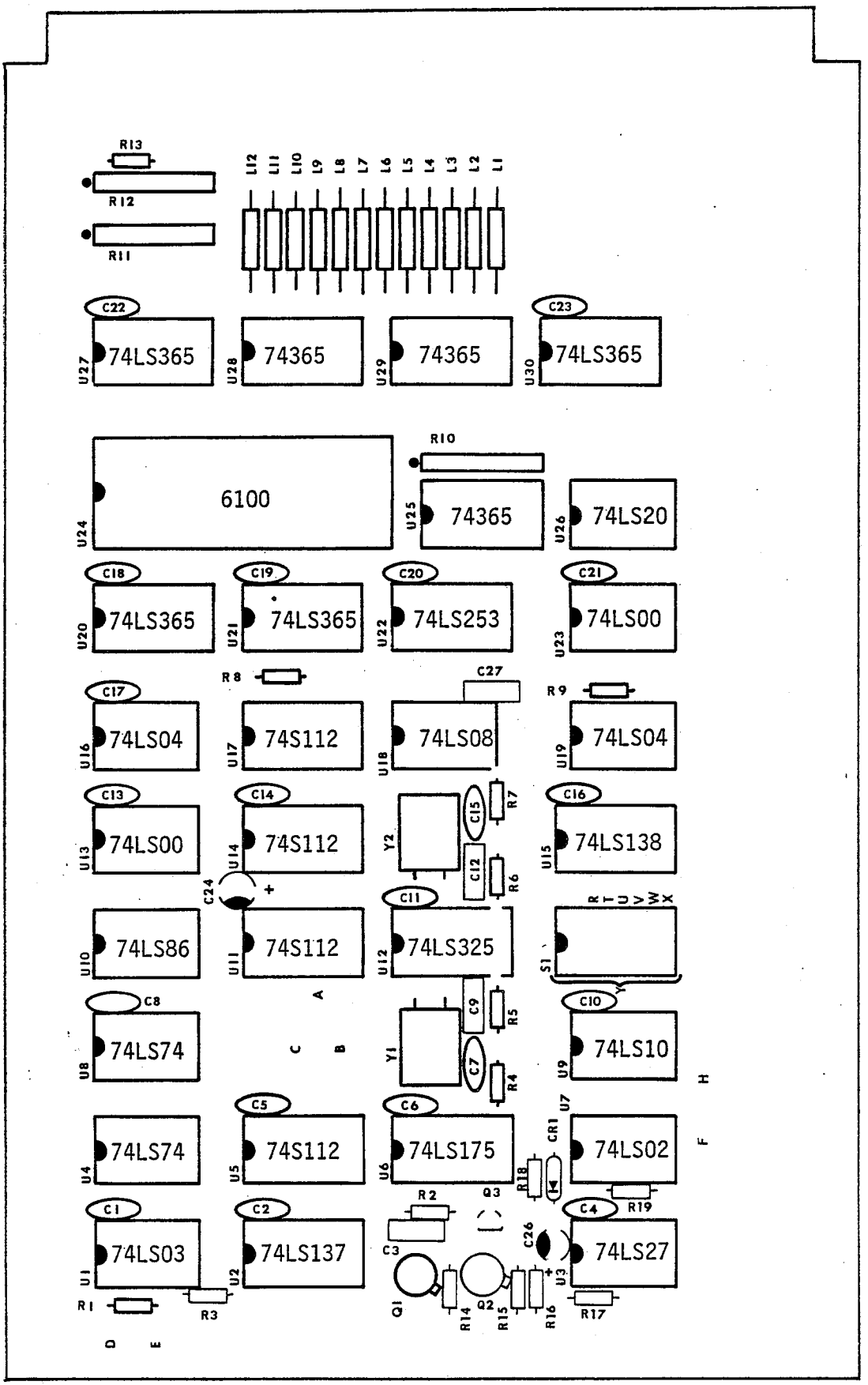


Figure 5-10
Printed-Circuit Layout for 12510 CPU Module

SECTION 6.0

12020-B STATIC MEMORY MODULE

		Page Number
6.1	MEMORY MODULE OPERATION	6-1
6.2	PROGRAMMING THE MEMORY MODULE	6-3

SECTION 6.0

12020-B STATIC MEMORY MODULE

The 12020 static memory is the basic memory module for the PCM-12 system. It provides 4096 12-bit words of semiconductor memory - equivalent to one full "field" of PDP-8 style memory. This module is used for storage of both instructions and data in the PCM-12 computer. The module contains 48 1024-bit static, N-channel metal-oxide-semiconductor (MOS) random-access-memory (RAM) devices, plus the necessary logic to interface these devices to the PCM-12 bus. Maximum access time for each memory word is 400 nanoseconds.

The 12020-B memory module represents an improved re-design of the original 12020 and 12020-A memory modules. Due to the higher-speed design techniques employed, and the use of a faster-grade memory device, no memory "wait" is ever required. Therefore, the 12020-B module provides no logic for assertion of the WAIT bus line.

Each 12020-B memory module is jumper-programmable to the desired "field" of operation (an 8-pole DIP switch can be substituted for the jumpers in systems where rapid re-programming of the memory module is necessary). The first module of memory in the system is always Field Ø. Other modules may be programmed to any field from 1 to 7, in keeping with PDP-8 type memory organization.

6.1 MEMORY MODULE OPERATION

Refer to Drawing 12022-B, the schematic diagram for the 12020-B memory module, and also Figures 5-1 and 5-2 in this manual. Early in each memory-access cycle, the 12-bit address of the location to be accessed is latched into the 74LS375 latch devices U4 - U6 on each 12020-B module. The CPU puts the address onto the bus DX lines at $\overline{T1}$ time, then delivers the LXMAR pulse. While LXMAR is high, the address passes through the 'LS375 latches and is then latched at their outputs on the trailing edge of LXMAR. Compared to using D-flip/flops for the address latches, this technique improves memory access time by allowing the address to propagate straight through to the memory devices just as soon as

it is available on the bus (while LXMAR is high).

Meanwhile, the EMAØ, EMA1 and EMA2 bus lines are decoded by U1 to determine the "field" being addressed. If the declared field is the one to which the given 12020-B module is programmed by the FIELD SELECT switch, both U10a and the C-input (pin 13) on U11 will be enabled.

After passing through the latch at U4, the two most significant address bits are decoded by U11. Then, if pin 13 on U11 is enabled (high), one of the four rows of memory devices will have their chip-select inputs (pin 13 on each 2102-type memory device) enabled during the time when LXMAR is low. For a "read" cycle, each enabled memory device makes its addressed location's data available at its output (pin 12). Then when MEMSEL is asserted low by the CPU (with XTC high for a memory read) U7 and U8 are enabled through U10a to drive the read data onto the bus DX lines. Note that each 12020-B memory module is allowed to drive the DX bus (during "read" time) only when its programmed output from U1 is asserted; thus two memory modules never try to drive the bus at the same time.

If the cycle requires a "write" to memory, the actions described above are first completed (usually this is an irrelevant, or don't-care "read"), then during T6 the MEMSEL line is again asserted. At this time the CPU puts the data to be written into memory onto the DX bus lines. Since XTC is low, the paralleled gates U10b, U10c and U10d are enabled. When the MEMSEL pulse is delivered, the write-enable pin (pin 3) on all RAM devices is asserted through these gates, and the 12 RAM's whose chip-enable inputs are also asserted (if any) are written into. This completes the "write" operation.

The physical organization of the memory devices on the 12020-B module is straightforward. Each 1024-word block of memory is composed of one row of 12 memory devices, and provides 2000_8 memory locations, as identified by the epoxy legend on the printed-circuit board (and as noted in Figure 13-1 of the PCM-12 Assembly and Checkout Manual). The most-significant bit in each block is marked by the nomenclature MSB, and the least-significant bit by LSB, again as shown in the epoxy legend.

6.2 PROGRAMMING THE MEMORY MODULE

The only programming necessary on the 12020-B memory module is to place a jumper across selected pins on the "FIELD SELECT" switch silhouette to select the memory field as which the given module will operate. If only one memory module is present in the system, it is always programmed to Field Ø. When additional modules are added, they may be programmed to any field from 1 to 7.

To program the memory field, locate the FIELD SELECT switch silhouette on the memory module. As is the case with all integrated circuits on this module, pin 1 on this switch is nearest the top of the board, as it is viewed with the gold contact fingers on the right. FIELD SELECT programming is accomplished according to the following table:

Desired Memory Field	FIELD SELECT jumper
Ø	1-16
1	2-15
2	3-14
3	4-13
4	5-12
5	6-11
6	7-10
7	8-9

SECTION 9.0

POWER SUPPLY CONSIDERATIONS

	Page Number
9.1 AC LINE VOLTAGE SELECTION	9-1
9.2 DC LEVEL ADJUSTMENTS	9-2

SECTION 9.0

POWER SUPPLY CONSIDERATIONS

The PCM-12 computer is basically a 5-volt machine. That is, plus 5 volts is the power supply level required by all plug-in modules. A few modules also require plus and/or minus 12 volts for operation of MOS devices, developing EIA RS-232 interface levels, etc. The bus also provides a fourth, unused, power supply line which the user may use for his own purposes. The bus lines distributing power are:

Lines 3,4,5,6:	+5 volts
Lines 19,20:	+12 volts
Lines 57,58:	-12 volts
Lines 69,70:	unused, available to user

All standard PCM-12 OMEGA mainframes are shipped with the 12900 power supply. This is a well-designed, high-quality OEM module supplied by a major power-supply manufacturer. The supply can operate from 100, 120, 220 or 240 volts AC, at 50-60 Hz frequency (output derated 10% for operation at 50 Hz). The supply has automatic fold-back current limiting to protect it from overloads, and over-voltage protection on the 5-volt output to protect the logic to which it supplies power. The supply is rated to provide full output in ambient temperatures up to 50 degrees C. Regulation and ripple combined are better than 0.1%. The supply provides 5 volts at 12 amps, and plus/minus 12 volts at 1.7 amps. This is normally sufficient power to supply even the very largest PCM-12 system configurations.

9.1 AC LINE VOLTAGE SELECTION

The POWER Switch, power receptacle and fuse for the PCM-12 Omega microcomputer are located on the rear panel of the machine. The power receptacle contains both the fuse and a programming card for selection of the proper line voltage.

The programming card allows selection of four different nominal line voltages: 100 volts, 120 volts, 220 volts or 240 volts, at either 50 or 60 Hz. The input line voltage should be within plus/minus 10% of the nominal value. Set-up of the power receptacle for the desired line voltage is accom-

plished by correctly positioning the programming card behind the clear plastic window in the receptacle. The voltage for which the card is set can be read through the window. As shipped by PC/M, this will normally be "120" for domestic units and "240" for units shipped internationally. To change to another line voltage, simply slide the window to the side, remove the fuse by pulling the fuse-removal lever, and pull the programming card out with a pliers. Then turn the card and re-insert it into the receptacle. Be sure the proper line voltage is now readable through the window, and replace the fuse.

DO NOT ATTEMPT OPERATION WITHOUT FIRST SETTING THE POWER RECEPTACLE FOR THE PROPER LINE VOLTAGE. PERMANENT DAMAGE MAY OTHERWISE RESULT.

A 3.0 ampere medium-blow fuse should be used for 100- or 120-volt operation. For 220- or 240-volt operation, use a 1.5 ampere fuse.

9.2 DC LEVEL ADJUSTMENTS

The nominal 5-volt and plus/minus 12-volt power supply output levels are adjusted by PC/M before shipment to the following values:

5-volt output:	adjust to 5.15 volts
+12-volt output:	adjust to +12.3 volts
-12-volt output:	adjust to -12.3 volts

The small increments over the nominal supply voltage levels provide for the slight drops which occur in printed-circuit traces, and assure that semiconductor devices which depend on their supply voltages being within a critical voltage band will receive the required levels.

The user should ocassionally check the power supply output levels with an accurate digital voltmeter to assure that component aging does not change the adjusted values. The voltage levels should be measured at the power supply connection posts on the backplane bus board. The adjustment potentiometers are on the small printed-circuit boards on the power-supply chassis.

SECTION 10.0

12080 AUDIO CASSETTE RECORDER INTERFACE

		Page Number
10.1	LOGIC OPERATION	10-2
10.2	MODULATION/DEMODULATION AND CLOCK-RECOVERY	10-5
10.3	INSTRUCTION SET AND DECODING	10-8
10.4	PROGRAMMING THE INTERFACE MODULE	10-10
10.5	OPERATING THE 12080 INTERFACE MODULE	10-10
10.5.1	VCO CENTER FREQUENCY ADJUSTMENT	10-11
10.5.2	INITIAL OPERATION	10-11
10.5.3	LOADING A BIN-FORMAT PROGRAM	10-12
10.5.4	OPERATION USING DEVICE CODES 03 & 04	10-13
10.5.5	USING MORE THAN ONE 12080 INTERFACE	10-14
10.5.6	CHOICE OF RECORDER AND TAPE	10-14
10.5.7	FINAL COMMENTS	10-15

SECTION 10.0

12080 AUDIO CASSETTE RECORDER INTERFACE

The 12080 interface module provides the ability to substitute a low-cost audio cassette recorder (or reel-to-reel recorder) for the more typical paper-tape storage medium used with the PCM-12 computer. This magnetic-tape data storage technique offers the advantage of faster operation than the low-speed paper-tape reader/punch found on a TTY terminal (typically 30 characters per second versus 10 characters per second for the paper tape equipment) with but one practical disadvantage - the lack of a stop-on-character capability for either the "reader" or "punch" function.

The 12080 module contains all the necessary modulation, signal conditioning and logical interfacing circuitry required to form the complete interface between the input/output of a typical cassette recorder and the PCM-12's bus structure. The module's modulation scheme employs the "BYTE-standard" self-clocking FM recording technique. Data is recorded on the tape at two tone frequencies - a logical "one" is recorded as an eight-cycle burst at 2400 Hz, and a "zero" is recorded as four cycles at 1200 Hz.

When recording data on tape the 12080 module employs a crystal-controlled, digitally-synthesized sine-wave which is coherently shifted between the two tone frequencies; there are no phase discontinuities. An amplitude-compensating circuit equalizes the levels of the two tone frequencies. The transmitter section of an Universal Asynchronous Receiver Transmitter (UART) device acts as the interface between the parallel-format data on the PCM-12 bus, and the serial-format data which must be input to the recorder.

When receiving data from the magnetic tape, the 12080 module filters the signal to remove energy at unwanted frequencies, converts the received "sine-wave" to a square-wave, and translates the received level to that suitable for input to standard CMOS logic. Hysteresis at the input comparator's switching points provides further discrimination against noise at the input. Digital processing eliminates errors due to high-frequency noise (hiss, et al) on the tape, and further conditions the signal for input to the phase-lock-loop (PLL) clock-

recovery circuit.

The data-recording scheme used on the 12080 module is self-clocking; that is, the clock signal required to recover the data as it is received from the tape is an exact multiple of the two tone frequencies into which the data is encoded on the tape. For 300-Baud operation the PLL circuitry synthesizes a 4800 Hz square-wave from the received tones at 1200 and 2400 Hz. The 4800 Hz signal provides the required 16-times-data-rate clock for the receive side of the UART device. The input circuitry to the PLL operates "straight-through" for a 2400 Hz input signal, and provides the extra transitions required to convert a 1200 Hz input signal to an apparent frequency of 2400 Hz. The PLL filters its received input signal, and delivers a smoothed clock signal at 4800 Hz. The voltage-controlled-oscillator (VCO) in the PLL runs at 19,200 Hz (i.e., 8 times 2400 Hz). The phase-detector in the PLL operates at a frequency of 2400 Hz. A divide-by-8 counter converts the VCO frequency to that required for phase-detector operation.

The 12080 module normally utilizes device codes 01 & 02 and/or 03 & 04 for its operation. When employing device codes 01 and 02 for its operation, it emulates a standard high-speed paper-tape reader/punch interface, i.e., it executes the standard DEC instruction set for devices 01 and 02. The module can also be made to emulate the interface for the low-speed reader and punch normally found on a TTY terminal, using device codes 03 and 04. When emulating devices 03 and 04, the interface executes the same instruction set as that for a standard TTY terminal. Response to instructions for devices 03 and 04 can be disabled by a jumper on the module, or by the front-panel AUX switch.

Typically for operation as the second 12080 interface in the system, the module can be jumper-programmed for operation with device codes 11 and 12, and/or 13 and 14, instead of 01 - 04. In this case all comments above applying to devices 01 and 02 then apply to devices 11 and 12, and those for devices 03 and 04 apply to devices 13 and 14.

10.1 LOGIC OPERATION

Refer to Drawing 12082, the schematic diagram for the 12080 interface module. Also refer to Section 4.2.2 of this manual for details concerning IOT instruction operation. Early in the IOTA execute cycle for each IOT instruction, the least significant six bits of the instruction are latched into device U2 by the trailing edge of the LXMAR pulse. The trailing edge of LXMAR also latches

the state of DX bits 3 - 5 into flip-flop U4B after they are decoded to one bit by gates U26A and U26B, and the inverter U24F. Only if the result is a low at pin 12 of U4B, so that the flip-flop is brought to the "clear" state by LXMAR's trailing edge, will the interface module be enabled to respond to the IOT instruction.

Next, the latched six bits from U2, the Q output of flip-flop U4B and the 03/04 ENABLE line are brought to the eight address inputs of the read-only-memory (ROM) device U3 for decoding. The 256-word-by-4-bit ROM translates the specific instruction into a four-bit code at its outputs for input to devices U5 and U6. The program for the ROM is given in Table 10-1.

The large number of instructions to which the 12080 module responds can be broken down into 15 groups of one or more instructions each. A logical output for each of these groups is provided by the decoder devices U5 and U6. For any instruction intended for the 12080 module, one (and only one) output of either U5 or U6 will be asserted low at DEVSEL time. The four outputs of device U3, plus the state of the XTC bus line (to decide whether the instruction is a "read" or "write" operation) determine which of the fifteen outputs will be asserted. A sixteenth "default" state (all output pins on U3 low) is entered when the instruction is not intended for the 12080 module. This state causes an assertion of the unused output (pin 15) of device U6. Thus the full decoding of each instruction is accomplished by the program in U3 and the 3-line-to-8-line decoder devices U5 and U6.

Because of the similarity of the instruction sets for devices 01 & 02 and 03 & 04, many of the decoded instructions perform similar or identical operations to other instructions. Thus gates U8B, U25D, U9A, U25A, U8A, U25B, U8C and U9C "or" two or more of the decoded instructions together before performing the required operations.

Since each of the outputs of devices U5 and U6 is labeled with its instruction code, the specific operations of the interface for each instruction can be understood by comparing the action indicated in Tables 10-2 and 10-3 with the logical operation indicated by the gates on the schematic diagram.

For example, when the Data-Ready flag (pin 19 on the UART device) is high, either instruction 6011 or (if enabled) 6031 will cause the next instruction to be skipped. This is accomplished by pin 11 of U5 being asserted low at T3 time, which in turn asserts the SKP bus line low through inverter U24A and gate U10B if pin 19 of U27 is high.

All locations programmed to produce an output of 0000, except the following:

Location Number (decimal) *	D3	D2	D1	D0
8	1	0	0	0
9	0	1	0	0
10	0	1	0	1
12	1	0	1	1
14	0	0	0	1
16	1	0	0	1
17	0	1	1	0
18	1	1	0	0
19	1	1	1	1
20	1	1	0	1
22	1	1	1	0
24	1	0	1	1
25	0	1	0	0
26	0	0	1	0
28	0	1	0	1
29	1	0	1	0
30	0	0	1	1
32	1	1	1	1
33	0	1	1	0
34	1	1	0	0
36	1	1	0	1
37	0	1	1	1
38	1	1	1	0
72	1	0	0	0
73	0	1	0	0
74	0	1	0	1
76	1	0	1	1
78	0	0	0	1
80	1	0	0	1
81	0	1	1	0
82	1	1	0	0
83	1	1	1	1
84	1	1	0	1
86	1	1	1	0

* - Location 0 corresponds to all address inputs low; address 255 to all address inputs high.

TABLE 10-1

ROM Device U3 Programming

Similarly, instructions 6022, 6026, 6042 and 6046 (the latter two when enabled) all cause the Write-Ready flag flip-flop (U4A) to be cleared by

asserting low an input to gate U8A. Instruction 6007 or a front-panel RESET will also clear this flag by asserting low the third input to this gate.

Flip-flop U7A is the Cassette Interrupt Enable Flip-Flop. When this "flop" is set (either by a front-panel RESET or instructions 6007, 6010 or 6035) the 12080 interface is enabled to request a device-interrupt.

Flip-flop U4A is the Write-Ready Flag flip-flop. This flip-flop is normally set by an assertion of the TRE (Transmitter Register Empty) terminal (pin 24) of the UART. However, it may also be forced to the set state by instructions 6023 or (if enabled) 6040 for initializing the "punch" function of the interface.

Flip-flop U7B is provided for control of the drive motor in the cassette recorder, if desired. Either instruction 6014 or (if enabled) 6030 will set this flip-flop; it is reset only by instruction 6007 or a front-panel RESET. Motor control can be accomplished by a reed-relay driven by gate U25C.

10.2 MODULATION/DEMODULATION AND CLOCK RECOVERY CIRCUITRY

Data is recorded on the magnetic tape in standard asynchronous serial data-communication format: one (low) start-bit , eight data bits and one or more (high) stop-bits. This serial data stream is produced at the TRO (Transmitter Register Out) terminal (pin 25) of the UART. The sense, high or low, of this serial bit stream is used to control the count modulus (divide-by-two or divide-by-four) of the two-stage divider formed by flip-flops U20A and U20B. The clock input to these flip-flops is at a frequency of 38,400 Hz for 300-Baud operation, and is derived from the crystal-controlled BAUDRAT bus line through the frequency-divider device U23. The output from flip-flop U20B is then at a frequency of either 19,200 Hz (logical "one") or 9600 Hz (logical "zero"), depending on the sense of the particular bit in the serial stream. This output is then delivered to device U21, which is connected as a four-stage Johnson counter. The outputs of this device are combined to synthesize a 2400 (or 1200) Hz sine-wave at the input of the summing amplifier/filter device U22A. The field-effect-transistor (FET) Q1 is used to boost the gain of U22A slightly for a transmitted "one" (2400 Hz) to produce balance in amplitude between the 1200 and 2400 Hz tone bursts. Device U22B serves as a buffer amplifier and low-pass filter to drive the output to the tape recorder's MICROPHONE input. The direct output, at pin 8 of the CABLE connector, provides a peak-to-peak signal amplitude of about 0.5 volts, at relatively high impedance. The output at pins 6 and 7 of the same connector provide a signal am-

plitude of about 2 millivolts, peak-to-peak, at an impedance level of about four ohms.

When taking data from tape, the 1200/2400 Hz tones are received at pin 2 of the CABLE connector, at a peak-to-peak signal amplitude of approximately two volts. The low-noise amplifier U14 is used as an amplitude-comaprator with a small amount of hysteresis at the switching points, to produce CMOS-compatible square-wave drive for the logic circuitry to follow. It is the purpose of this logic circuitry to recover a 16-times-Baud-rate clock signal from the tone bursts for delivery to the Receive Register Clock (RRC) terminal (pin 17) of the UART, and to decode the tone bursts themselves back into digital one's and zero's.

The counter device U13, the gates U11A, U11B and U11C, the flip-flop U15A, and the combinational and sequential logic contained in device U12 are used to decode the tone bursts into one's and zero's. A basic timing diagram for this operation is shown on the next page. The basic scheme is to use the counter U13 to time the interval between transitions of the square-wave at pin 7 of U14. If this interval exceeds a standard, the tone frequency is determined to be 1200 Hz. Otherwise, it is determined to be 2400 Hz. The result is that the recovered data is found at U27, pin 20, as shown in the timing diagram, Figure 10-1. Inverting transistor Q4 drives this data into the serial Receiver Input (RI) terminal (pin 20) of the UART.

As a by-product of the data-decoding process, a distorted 4800 Hz (for 300-Baud operation) square-wave is produced at pin 5 of U12, as shown in Figure 10-1. Device U15 is then used to smooth the phase irregularities in this waveform, and deliver a 2400 Hz driving signal to the phase-detector in the PLL device U16. The PLL, in conjunction with the counter U17, then generates a 4800 Hz square-wave to drive the RRC input of the UART. For 300-Baud operation the VCO in U16 runs at 19,200 Hz. This is divided by four to obtain 4800 Hz before delivery to the UART clock input. The center frequency of 19,200 Hz for the VCO must be established when initially aligning the circuitry, by placing a jumper between pad U and pad V and adjusting potentiometer R23 until a square-wave of this frequency is found at pin 4 of U16. The U - V jumper is then removed.

Although the BYTE-standard calls for operation of this circuitry at a data rate of 300-Baud, users with better-quality recording equipment may select operation at 600 or 1200 Baud. See Section 10.4 for proper placement of jumpers for operation at these rates. Also, for operation at 600 Baud, capacitors C2, C4, C6

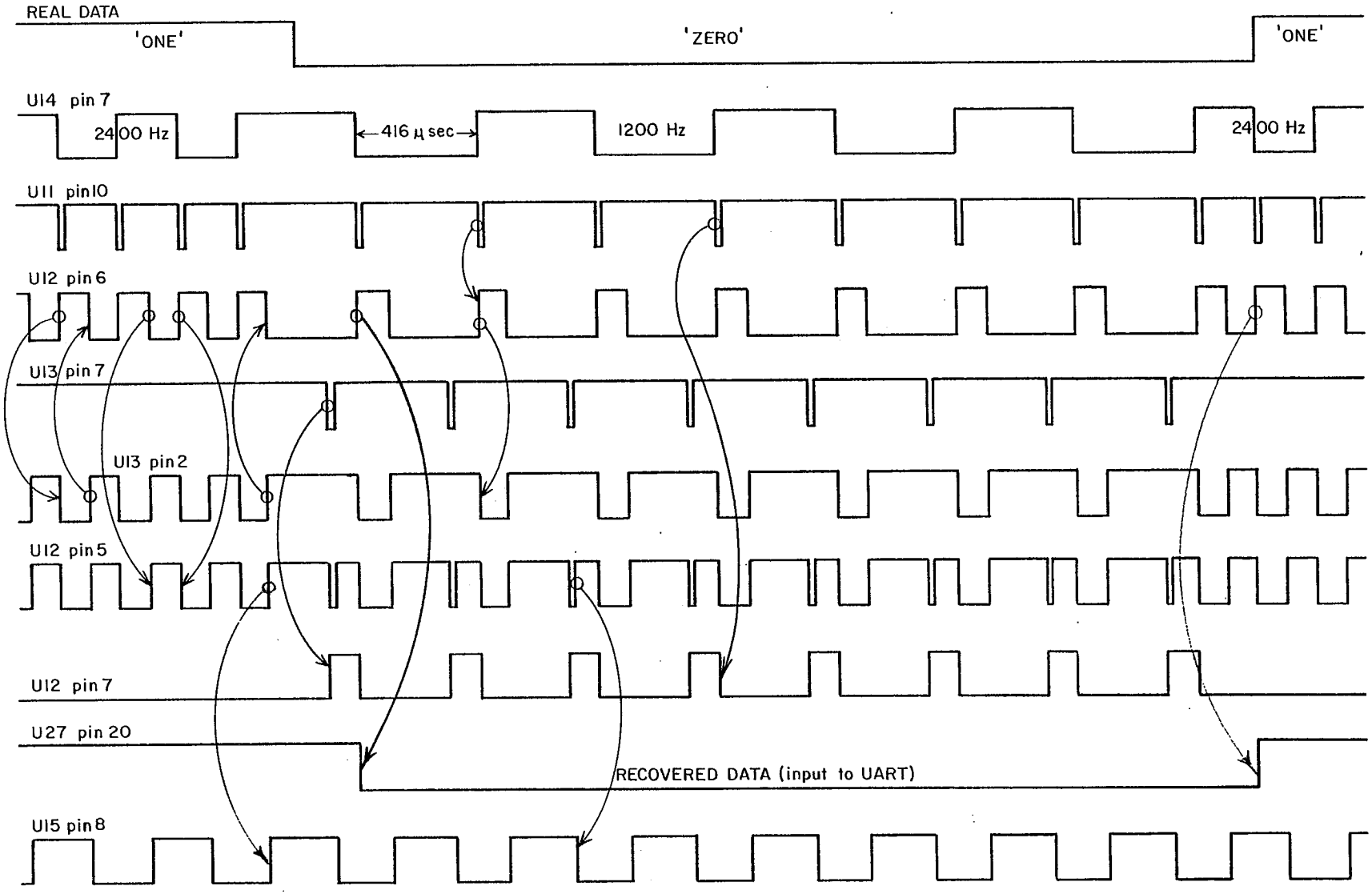


FIGURE 10-1
Timing Diagram for Data Recovery Circuitry

C10, C12 and C13 must all be halved in value. For operation at 1200 Baud, their values must all be halved again.

10.3 INSTRUCTION SET AND DECODING

The 12080 interface module executes the instruction set shown in Table 10-2 for devices 01 and 02 (high-speed reader/punch) and that shown in Table 10-3 for devices 03 and 04 (low-speed reader/punch) when a jumper exists between pads R & S.

MNEMONIC	OCTAL	OPERATION
RPE	6010	Set the Cassette Interrupt Enable Flip-Flop (CIEFF).
RSF	6011	Skip the next instruction if the Data-Ready flag is set.
RRB	6012	'OR' the UART contents into the AC, and reset Data-Ready flag.
RFC	6014	Reset Data-Ready flag (and set Cassette-Run Flip-Flop, U7B).
RRB,RFC	6016	'OR' the UART contents into the AC, and reset Data-Ready flag.
PCE	6020	Clear the Cassette Interrupt Enable Flip-Flop (CIEFF).
PSF	6021	Skip the next instruction if the Write-Ready flag is set.
PCF	6022	Reset the Write-Ready flag.
-	6023	Set the Write-Ready flag (not part of DEC instruction set).
PPC	6024	AC contents non-destructively transferred to UART.
PLS	6026	Combination of PCF & PPC. Write-Ready flag is reset and AC contents non-destructively transferred to UART.

TABLE 10-2

Instruction Set for Devices 01 & 02

When the jumper is between points R & T, Table 10-2 applies for devices 11 and 12, and Table 10-3 for devices 13 and 14. The instruction sets given in these tables are the standard DEC instruction sets for the high- and low-speed paper-tape equipment. Thus, the 12080 interface will operate compatibly with standard DEC- or user-written software for devices 01 & 02, and 03 & 04.

Note that in Table 10-2, one instruction (6023) has been added to the

standard DEC set for devices 01 and 02. Instruction 6023 can be utilized in new software written expressly for the 12080 interface. It will jam-set the Write-Ready Flag (U4A) for initializing the "punch" side of the interface without the need for outputting a character to tape. This addition is merely for programmer convenience, and has no effect on DEC-software compatibility.

Also note that since the normal reader-enabling function has no utility in a reader without stop-on-character capability, a motor-start function has been

MNEMONIC	OCTAL	OPERATION
KCF	6030	Reset the Data-Ready flag (and set Cassette-Run Flip-Flop, U7B).
KSF	6031	Skip the next instruction if Data-Ready flag is set.
KCC	6032	Reset the Data-Ready flag and clear the Accumulator (AC).
KRS	6034	'OR' the UART contents into the AC.
KIE	6035	Load AC bit 11 into the CIEFF. If AC(11) is a "one", the CIEFF is set, otherwise it is cleared.
KRB	6036	Reset the Data-Ready flag and jam-load the UART contents into the AC. This is a combination of KCC and KRS.
TFL	6040	Set the Write-Ready flag.
TSF	6041	Skip the next instruction if the Write-Ready flag is set.
TCF	6042	Reset the Write-Ready flag.
TPC	6044	AC contents non-destructively transferred to UART.
TSK	6045	Skip the next instruction if the CIEFF is set and either the Data-Ready flag or the Write-Ready flag is set.
TLS	6046	Reset the Write-Ready flag, and non-destructively transfer the AC contents to the UART. This is a combination of TCF & TPC.

TABLE 10-3

Instruction Set for Devices 03 & 04

added to instructions 6014 and (when enabled) 6030. Execution of either of these instructions can be used to set flip-flop U7B under software control. (See Section 10.1 for discussion of this flip-flop.)

The 12080 module also responds to instruction 6007. This instruction, which is decoded on the 12060 TTY interface module and asserts the bus RESET line

low, clears the UART device, the receive-side Data-Ready flag (UART pin 19), the Write-Ready flag (U4A, pin 5) and flip-flop U7B. It also sets the Cassette Interrupt-Enable Flip-Flop (CIEFF), U7A.

10.4 PROGRAMMING THE INTERFACE MODULE

The 12080 module contains several jumper pads for programming the specific operation desired of the module. Each of the jumper pad sets is discussed below.

Pads A, B & C program the UART device for either one- or two-stop-bit operation. Two-stop-bit operation provides slightly better reliability, but at a 10% decrease in through-put rate. To program the module for two-stop-bit operation, solder a jumper between pads A & B. For one-stop-bit operation, the jumper goes between pads A & C (leaving pad B unconnected).

For normal operation at 300 Baud, the counter U23 must be programmed by placing a jumper from pad D to pad G, and another from pad H to pad L. For operation at 600 Baud, jumper pad D to pad F, and pad H to pad K. For operation at 1200 Baud, jumper pad D to pad E, and pad H to pad J.

Placing a jumper from pad P to pad M allows the front-panel AUX switch to control the module's response to device codes 03 & 04 (or 13 & 14). Placing the jumper from pad P to pad N permanently enables these device codes; omitting the jumper entirely permanently disables them.

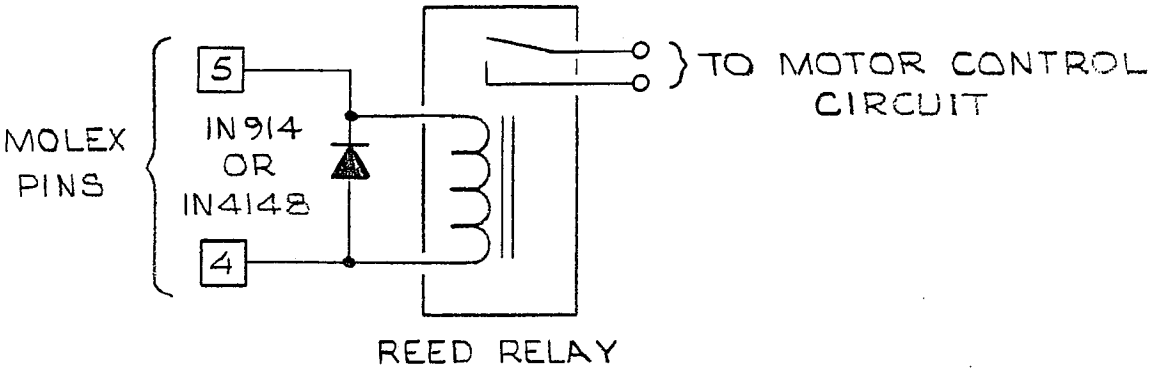
Depending on the use of the R-S-T jumper pad set, the 12080 module will respond to instructions for device 01, 02, 03 and 04, or 11, 12, 13 and 14. Device codes 01 - 04 are enabled when a jumper is placed from pad R to pad S. Device codes 11 - 14 are enabled with a jumper from pad R to pad T. The former selection is normally made if only one 12080 module will exist in the system; the latter selection is usually reserved for the second 12080 module in the system.

Refer to Section 10.5.1 for operation of the U and V jumper pads.

10.5 OPERATING THE 12080 INTERFACE MODULE

To begin using the 12080 interface module, first connect the tape recorder input/output to the Molex connector on the rear panel of the computer. (See page 12-8 of the Assembly Manual for a listing of the Molex pin connections.) The output of the recorder should be connected to pin 2 of the Molex connector, while the high-impedance recorder input (if available) goes to pin 8. Ground on the recorder should be connected to pin 3 of the Molex connector. If you will use the recorder's MICROPHONE input, a 2-millivolt driving signal is available across pins 6 & 7 on the Molex connector.

To make use of the interface's motor-control capability, a reed-relay can be wired in series with the recorder's "power" circuit, as shown below:



Next, be sure you have the interface jumpers properly programmed (see Section 10.4, above). The discussion below will assume that a jumper exists between pads P & M, and also between pads R & S. Thus, devices codes 01 and 02 are always enabled, and codes 03 and 04 are enabled (on the 12080 module) when the front-panel AUX switch is in the "up" position.

10.5.1 VCO CENTER FREQUENCY ADJUSTMENT

Before using the 12080 module to load data from tape, it is necessary to adjust the center frequency of the PLL voltage-controlled oscillator (VCO). To make this adjustment, "tack-solder" a jumper from pad U to pad V, and slide the module into the computer. Then, using a frequency-counter or oscilloscope to observe the frequency of the square-wave at pin 4 of U16, adjust potentiometer R23 for a frequency of 19,200 Hz (300 Baud operation). Try to set it as accurately as possible, then remove the U - V jumper. The PLL is now ready to operate.

10.5.2 INITIAL OPERATION

The following short program can now be toggled in from the front panel and run to familiarize yourself with operation of the hardware, and also to learn what leader/trailer code sounds like when recorded on tape. Learning the distinctive "note" of the leader/trailer is important because when loading a binary-format program with the control-panel ROM routine, the BIN BOOT switch must be raised when the recorder is sending leader code to the interface. The program is as follows (next page):

ADDRESS	CONTENTS	COMMENT
0200	7300	Clear Accumulator and Link
0201	6023	Set Write-Ready flag
0202	1210	Fetch leader-code to Accumulator
0203	6021	Skip if Write-Ready flag is set
0204	5203	Jump back if not
0205	6026	Punch the leader-code on tape
0206	7200	Clear the Accumulator
0207	5202	Loop back and do it again
0210	0200	Leader-code

Toggle the program into Field Ø, then turn the recorder on and adjust its record-level control for the best recording level. Next, start the program at address 0200 and let it run for about two minutes. Then stop the computer, rewind the tape, and play it back through the recorder's loudspeaker. Note that the leader code is easy to identify by its distinctive "note". The "random" data that usually follows it in a BIN-format tape will sound much different.

10.5.3 LOADING A BIN-FORMAT PROGRAM FROM TAPE

To load a binary-format program from a tape cassette (one of the programs on the cassette furnished with the interface module, for example), follow these instructions (also refer to Section 4.5 of this manual):

1. Put the Rotary Switch in the AC position.
2. Set the Data Field to the field into which the program should load (usually Field Ø).
3. Set SWITCH REGISTER bit Ø "down" (device codes 01 and 02 will be used).
4. Start the recorder running in "playback" mode. Find the leader-code segment for the desired program. Adjust the level at the input to the interface module (Molex connector pin 2) for a signal of 2 volts peak-to-peak amplitude.
5. Now simply lift and release the BIN BOOT switch and the program will begin loading. Note that the LINK light flashes on and off at a rapid rate when the interface is reading leader code. And remember - the recorder must be sending leader code to the interface at the time the BIN BOOT switch is activated.
6. While the program is loading, the MEMORY ADDRESS lamps will show the address each character is loaded into, and the Display lamps will show the character itself.
7. When the loader routine finds the trailer code at the end of the

program, it will halt the computer, with the checksum result in the Display lamps. If the checksum is zero, the program has loaded properly. If not, the entire procedure must be repeated.

10.5.4 OPERATION USING DEVICE CODES 03 AND 04

Both the 12060 TTY interface module (with jumper L - M omitted) and the 12080 cassette interface module (when enabled by jumpers R - S, and P - M or P - N) have the ability to respond to device codes 03 and 04. However, it must never be the case that both modules are enabled to respond to these codes simultaneously, because both would then try to drive the bus and a conflict would result.

To eliminate this potential difficulty, jumper L - M is installed on the 12060 module, and jumper P - M on the 12080 module. Then when the front-panel AUX switch is in the "down" position, the 12060 module will be enabled to respond to device codes 03 and 04 and the 12080 module will be disabled from responding to these codes (although it will still respond to codes 01 and 02). With the AUX switch in the "up" position, the 12080 module is enabled, and the 12060 module is disabled.

(NOTE: The 12080 module's ability to respond to device codes 03 and 04 is provided so that some DEC-written software which assumes that only a low-speed reader and punch are available can be used. If the user only needs the 12080 interface for emulation of a high-speed reader/punch, the entire question of potential device 03 - 04 conflict can be made moot by omitting the jumper at pad set M - P. This will completely eliminate the 12080 module's ability to respond to device codes 03 and 04.)

When using the 12080 interface with device codes 03 and 04, the user can instantly switch back and forth between the 12060 and 12080 interfaces with the AUX switch. For example, suppose that a program is running which uses the TTY terminal for program communication. The AUX switch should then be in the "down" position, to activate the 12060 TTY interface. If the user should then come to a point in the program where it was required to load data from tape using device codes 03 and 04, he would start the recorder running, then type the characters required to start the loading procedure and flip the AUX switch to the "up" position to activate the 12080 interface and deactivate the TTY. After the data load was complete, he would put the AUX switch back in the "down" position and proceed with the TTY terminal enabled.

10.5.5 USING MORE THAN ONE 12080 INTERFACE MODULE

Since the "read" and "punch" functions of the 12080 interface are completely independent, it is entirely possible to operate two recorders simultaneously from one interface - one recording data, while the second is being read from. To maximize this flexibility, however, it is most convenient to have two 12080 interface modules in the system.

When employing two 12080 interfaces, the second should have the R - T jumper (rather than R - S) in place. The second module will then operate with device codes 11 - 14, rather than 01 - 04. All comments above concerning device codes 01 and 02 will then apply to devices 11 and 12, and those for devices 03 and 04 will apply for devices 13 and 14. In particular, device codes 13 and 14 can be controlled from the front-panel with the AUX switch if jumper P - M is in place. Normally, however, this would not be necessary and device codes 13 and 14 can be permanently enabled by putting the jumper from pad P to pad N. Or, codes 13 and 14 can be permanently disabled by entirely omitting the jumper at pad set N - P.

10.5.6 CHOICE OF RECORDER AND TAPE

Either a reel-to-reel or standard low-cost audio cassette recorder can be used with the 12080 interface module. Normally, the cassette recorder will be found to be most convenient in use. In choosing a recorder, look for a capstan-drive unit with low flutter and wow (less than 5% peak-to-peak, or better if possible). A tape-footage counter will be found very handy for roughly locating programs on a long tape. A V-U meter is convenient for setting record-level, but is not too necessary, especially for the typical audio cassette machine that has built-in automatic-level-control. A microphone-input jack is important, for receiving input from the interface module. Also, look for a machine with an on/off switch on the companion microphone. This indicates the recorder will be controllable from the Cassette-Run Flip-Flop in the interface module. Finally, a variable-speed machine can be useful when playing back data recorded on another recorder; this gives the ability to compensate for motor-speed differences between different recorders. The J. C. Penney Model 6536 is a good choice in a low-cost cassette unit. The Sony C-104 has many more desireable features, but is considerably more expensive. A stereo-quality cassette recorder is not necessary for operation with the 12080 interface module, and in fact is often out-performed by

units in the $30 - $50 price range.

With any recorder the tape itself will be the weak link in the chain. Do not attempt to use "bargain-basement" magnetic tape. AUDUA TDK C-60 cassettes are highly recommended.

10.5.7 FINAL COMMENTS

While the audio cassette recorder provides a very convenient storage medium for any binary-format program, and also for bulk data storage, it does have one fundamental weakness - lack of a stop-on-character capability for either the reader or punch function. Many standard PDP-8 programs, while running, accept data from a tape reader with stop-on-character ability, or punch data on tape in bursts of a few characters each in stop-start manner. Using an audio cassette recorder as a substitute for paper-tape equipment under these circumstances may be less than satisfactory, since the running program cannot control the tape equipment. Once the cassette recorder's motor starts running, it cannot be instantly stopped and re-started. The only possible solution to this problem is to re-write the program software so that a stop-on-character ability is not required.

Even considering this single drawback, the audio cassette medium still offers one of the very best price/performance trade-offs available for data storage today.

SECTION 11.0

12560 SERIAL INTERFACE MODULE

		Page Number
11.1	INTERFACE OPERATION	11-2
11.1.1	DATA-RATE, INTERFACE LEVEL & STOP-BITS SELECTION	11-2
11.1.2	DEVICE-CODE SELECTION	11-3
11.1.3	TERMINAL INTERFACE - ELECTRICAL	11-3
11.1.4	TERMINAL INTERFACE - MECHANICAL	11-5
11.1.5	INSTRUCTION SET AND DECODING	11-5
11.1.6	OTHER LOGIC OPERATION	11-9
11.1.7	SUMMARY OF JUMPER OPTIONS	11-9
11.2	TELETYPE TERMINAL SELECTION AND MODIFICATION	11-10
11.2.1	SELECTION OF A TELETYPE MACHINE	11-10
11.2.2	TELETYPE TERMINAL MODIFICATIONS	11-11
11.2.2.1	FULL-DUPLEX OPERATION	11-12
11.2.2.2	20-MA OPERATION	11-13
11.2.2.3	READER-ENABLE RELAY	11-14
11.2.2.4	KEYBOARD PARITY	11-16
11.2.2.5	RECEIVE-SIDE CONNECTIONS TO INTERFACE	11-17

Note: This version of Section 11.0 replaces the previous version, which applied to the now-obsolete 12060 TTY/CRT interface module.

June, 1978

Section 11.0
12560 SERIAL INTERFACE MODULE

The 12560 serial interface module represents an improved re-design of the original 12060 TTY/CRT interface for the PCM-12 microcomputer. The new design incorporates such additional features as selectable device codes, a CMOS UART for either 5- or 10-volt operation, a greater range of Baud-rate selection, independently-selectable transmit and receive serial data rates and an optional break-on-framing-error capability. The module is completely compatible with the DEC software instruction set for the usual serial terminal connected to a PDP-8 computer system.

The 12560 module will be most frequently used as the interface to a Teletype (TTY) or cathode-ray-tube (CRT) terminal for direct operator communication with the microcomputer. It will also find useage, however, in communicating with a telephone-line modem or any other serially-oriented peripheral on the computer system.

The TTY or CRT terminal is the most fundamental peripheral for the PCM-12 computer. This device allows the user to communicate directly with the computer using a keyboard for input, and printed copy output. (Only the TTY terminal provides "hard copy" output, of course; CRT terminal output is limited to what the display screen can hold.) Additionally, the TTY terminal (ASR models) can provide a low-speed paper-tape input and output capability. CRT terminals may operate at any standard 75×2^n rate from 75 Baud to 19,200 Baud (using one stop bit), or at 110 Baud (using two stop bits). The TTY terminal operates at 110 Baud only, using two stop bits. A CRT terminal may operate using a 20-milliamp (20-ma) current-loop interface, or at RS-232 interface levels. The TTY terminal interfaces to a 20-ma current-loop only.

By jumper selection, the 12560 interface module will operate at any of the above-mentioned data rates, with one or two stop bits, and with RS-232 or 20-ma interface to the terminal. There is also provision on the 12560 module for use of an audio cassette recorder as a substitute device for the paper tape reader/punch on the ASR TTY terminal (when the 12080 audio cassette recorder interface is present in the system).

To achieve compatibility with the PDP-8 "style" architecture of the PCM-12, a Teletype machine requires some small modifications to its standard wiring. The required changes are discussed in Section 11.2.

11.1 INTERFACE OPERATION

Operation of the 12560 serial interface module is quite straightforward. For the following discussion, refer to Drawing 12562, the schematic logic diagram for the interface module.

11.1.1 DATA-RATE, INTERFACE LEVEL & STOP-BITS SELECTION

The desired operating serial data rates are selected by switches S5 and S6. Switch S5 determines the "transmit" speed (that is, the speed at which the computer will send data to the terminal) by selecting the appropriate output from the frequency-dividers U5 and U6. Similarly, switch S6 selects the "receive" data rate, the rate at which the computer expects to receive serial data from the terminal. Each switch is simply programmed by soldering a jumper at the desired rate stenciled on the interface board along-side the silhouettes for S5 and S6 (for example, soldering a jumper from pin 2 to pin 15 of S5 would select a transmit data rate of 9600 Baud). Rapid re-selection of the operating data rates can be accomodated by using 10-position rocker-arm DIP switches rather than solder jumpers at S5 and S6. If independent transmit and receive data rates are desired, then switches must be soldered at both the S5 and S6 positions. If the transmit and receive rates will always be identical, however, a single switch (installed at either S5 or S6) will suffice if a jumper is installed between pads R and T.

The U5/U6 frequency divider receives its input, at pin 1 of U5, at approximately 614 KHz from the BAUDRAT line on the backplane bus. The outputs from S5 and S6 are actually at 32 times the desired serial data rate. The UART requires an input clock at 16 times the data rate, and the flip-flops in U13 are used to divide-by-two and square up the S5 and S6 outputs. U5 is a simple eight-stage binary divider. U6 is programmed to divide-by-eleven to produce the proper frequency (3520 Hz at pin 9 of S5 and S6) for 110 Baud operation.

Operation with one stop bit is selected by placing a jumper from pad H to pad F; for two stop bits, the jumper goes from pad H to pad G.

When interfacing with the 20-ma current-loop, the RS-232 input/output can be ignored. However, when using the latter levels for interfacing, the

20-ma receive current loop must be completed by placing a jumper from pad J to pad K.

11.1.2 DEVICE-CODE SELECTION

The 12560 interface module has the ability to be jumper-programmed to any required device code for its serial-interface functions. For any selected serial-interfacing device code, the module will also decode, and respond to, processor-IOT instruction 6007, which asserts the bus RESET line (see Section 11.1.5, below).

The receive-side device code is selected by placing an appropriate jumper at the position designated S1 for the most-significant octal digit (MSD) of the desired code, and another jumper at S2 for the least-significant digit (LSD). Most often, this programming will require a jumper from pad RM to the nearby pad numbered 0, and a jumper from pad RL to the closeby pad marked 3. This will result in a receive device-code of 03, which is the code DEC normally assigns to the system-console keyboard.

Similarly, the transmit-side device code is selected by means of jumpers at the positions designated S3 and S4. These are most often programmed to device code 04 (DEC's code for the system-console printer) by placing a jumper from pad XM to nearby pad 0, and another from pad XL to closeby pad 4.

11.1.3 TERMINAL INTERFACE - ELECTRICAL

All serial data input and output between the 12560 interface and the terminal is made via U20, a Universal Asynchronous Receiver/Transmitter (UART) device. Input from the terminal arrives at pin 20 of the UART. Output to the terminal comes from UART pin 25.

On the other side of the UART, the data is in parallel format. Inputs from the PCM-12 bus arrive at pins 26-33 via U23 and a portion of U24. Outputs to the bus are at pins 5-12. Since the UART does not have sufficient output to drive the bus directly, devices U21 and U22 buffer the UART outputs onto the bus. These devices are enabled only during the "read" portion of an IOTA cycle, when the 12560 module has been addressed for input to the CPU. Note that the eight data bits used in asychronous communication come-from/go-to the eight least significant bits in the CPU's Accumulator.

Data communication between the terminal and the UART is as follows. On the receive side of the UART, pin 20 is always high when the keyboard and

reader are inactive. This is the "marking" state; the terminal is sending a constant stream of stop-bits to the computer. When a key is struck on the keyboard, the immediately-resulting "spacing" start-bit is detected at pin 10 of U19 for RS-232 interfacing levels or at pin 4 for a 20 ma current-loop interface. In either case, pin 20 of U20 is driven low through gate U24c, which causes the UART to begin clocking in the eight data bits from the terminal, and also causes the reader relay to be disabled by clocking flip-flop U14b to the reset state through U12e. As soon as the entire character has been clocked into the UART, the UART's Data-Ready flag is set, pulling pin 19 of U20 high. This causes the INTREQ bus line to be asserted low, if the Teletype-Interrupt-Enable Flip-Flop (U14a) is set, through gates U18a and U9a. The program software then uses instructions KSF and KRS or KRB to read the data into the CPU's Accumulator. Note that during the read operation, since four inputs on U21 are tied to the pull-up bus (PUB), the most significant four bits are read into the Accumulator as 0's.

On the transmit side, the CPU sends the data to the UART using instructions TSF and TPC or TLS, when it finds the Printer/Punch-Ready flag (U15a) is set. When the UART picks up the data from the DX lines on the bus, it begins to transmit the character to the terminal in serial format, by first sending a "spacing" start-bit, then the eight data bits (least-significant bit first), then one or more stop-bits. At the end of the character, pin 25 is in its normal rest (marking) state, where it stays until another character is ready to transmitted to the terminal. Also, at the end of the character, the UART signals completion of transmission by pulling its pin 24 high. This action, through gate U17a, clocks the Printer/Punch-Ready flag (U15a) set, so the CPU can ascertain that the UART is ready to transmit another character.

Gate U17a is used to condition the UART's setting of U15a, depending on the state of pin 4 on the module's 16-pin I/O connector, J1. When pin 4 is simply left un-connected, as it most often will be, gate U17a is enabled to allow the UART to directly set U15a. When pin 4 is used to implement the RS-232 Data Terminal Ready (DTR) function, however, one section of the triple RS-232 line receiver, U19, can be used to control gate U17a. When pin 4 is at a negative level U17a is enabled, and when pin 4 is positive, U17a is disabled. In addition to implementing the RS-232 DTR function, this circuitry can be used to accomodate slow-responding terminals, such as the Texas Instruments Silent 700 which requires a 250-millisecond delay after "printing" a carriage-return to avoid losing the first two or three characters

to be printed on the next line.

11.1.4 TERMINAL INTERFACE - MECHANICAL

Inputs and outputs to/from the terminal are made via a 16-pin DIP socket, J1, and a matching ribbon-cable and connector. Pin numbers on the DIP socket are noted in square boxes on the schematic diagram of the interface module. The RS-232 input is at pin 11 and the corresponding output at pin 15. The 20 ma current-loop input pins are numbered 13 (source) and 14 (sink), and the corresponding pin-pair for output is 16 (source) and 12 (sink). Current for the reader-control relay is at pins 9 (source) and 10 (sink). The RS-232 Data Terminal Ready input (optional) is at pin 4. An extra RS-232-level output is available at pin 2. By placing a jumper from pad N to pad P, pin 2 can be used as an independent duplicate output to that already available on pin 15. Alternatively, by placing a jumper from pad N to pad Q, pin 2 can be used to provide an RS-232-level duplicate of the reader-control relay signal. All other pins on J1 are grounded. Note that with respect to the corresponding connector on the earlier 12060 serial interface module, all pins on J1 have the same functionality except pins 2 and 4. The latter provide new signals which were not available on the 12060 interface; on the earlier module pins 2 and 4 were grounded.

At the rear panel of the PCM-12, an 8-pin Molex connector is provided for connection of the computer to the terminal. Assignment of the pin numbers on this connector is according to Table 11-1. Note that the recommended pin numbers are in reverse order to that shown in the corresponding table for the earlier 12060 interface module. This numbering reversal was done to accomodate the pin numbers which Molex has begun to mold into the rear-panel connector. Only the Molex connector pin numbers have changed; the physical wiring itself has not been modified.

11.1.5 INSTRUCTION SET AND DECODING

The 12560 interface module, when jumper-programmed for device codes 03 and 04, is fully compatible with all DEC software instructions intended for the Teletype or system-console terminal. The device code is embedded in the IOT instruction at bits 3-8. The instruction set for the Teletype or system-console interface is shown in Table 11-2.

In addition to all instructions for the serial peripheral device, the 12560 interface also decodes instruction 6007 (Clear All Flags, CAF). When

executed, this instruction clears all interrupt-request flags (U15a and the Data-Ready flag in the UART) on the 12560 module, and sets the Interrupt-Enable Flip-Flop, U14a. In addition, CAF causes the bus RESET line to be asserted low by Q1, which clears all the similar flags on other

MOLEX PIN #	20-MA OPERATION	RS-232 OPERATION
1	Reader Relay, J1 pin 9	Ground, J1 pin 8
2	Reader Relay, J1 pin 10	Ground, J1 pin 7
3	(-)20-ma output, J1 pin 12	Not used
4	Ground, J1 pin 5	RS-232 DTR, J1 pin 4
5	(+)20-ma input, J1 pin 13	RS-232 input, J1 pin 11
6	(-)20-ma input, J1 pin 14	Aux RS-232 output, J1 pin 2
7	Ground, J1 pin 2	RS-232 output, J1 pin 15
8*	(+)20-ma output, J1 pin 16	Ground, J1 pin 1

* - The Molex connector is mounted so that pin 8 is closest to the center of the rear panel. Note pin numbers molded into body of connector.

TABLE 11-1

Recommended Rear Panel Connections for TTY or CRT terminal

peripheral interfaces. (Note that CAF, although asserting RESET, is only asserted during the time DEVSEL is active, and this does not include the time when the CPU samples the RESET line. CAF does not cause a CPU halt.)

Each IOT instruction is latched into devices U1 and U2 by the trailing edge of the LXMAR pulse. (It is not necessary to latch, or decode, the most significant three DX bits, since these always form 6 (octal) for an IOT, and this Opcode and the DEVSEL pulse are redundant. DEVSEL only appears during IOTA cycles.) Decoding of the instruction takes place in the programmable read-only memory (PROM) U10. Due to the coding built into this PROM, a pulse for each decoded instruction (for the device whose device code has been selected by S1-S4) appears during DEVSEL time at the outputs of U10. Each instruction pulse, with negative sense, is noted on the schematic at the

MNEMONIC	OCTAL	OPERATION
KCF	6030	Reset the Keyboard/Reader Data-Ready flag.
KSF	6031	Skip the next instruction if the Keyboard/Reader Data-Ready flag is set.
KCC	6032	The reader is enabled to fetch the next character, the Data-Ready flag is reset, and the AC cleared.*
KRS	6034	The keyboard/reader data is OR'ed into the AC.
KIE	6035	AC bit 11 is loaded into the Teletype Interrupt-Enable Flip-Flop (TIEFF). This sets the TIEFF if AC(11)=1, and clears it if AC(11)=0.
KRB	6036	The keyboard/reader data is jam-loaded into the AC, the Data-Ready flag is reset, and the reader is enabled to fetch the next character.
TFL	6040	The Printer/Punch Ready flag is set.
TSF	6041	Skip the next instruction if the Printer/Punch Ready flag is set.
TCF	6042	Reset the Printer/Punch Ready flag.
TPC	6044	The AC is non-destructively written into the UART transmit buffer, from where the character is automatically delivered to the I/O device.
TSK	6045	Skip the next instruction if the TIEFF is set and either the Keyboard/Reader Data-Ready flag or the Printer/Punch Ready flag, or both, is set.
TLS	6046	The AC is written to the device, as in TPC. The Printer/Punch Ready flag is reset. The flag will be automatically set again when the UART is ready to send another character to the I/O device.
CAF	6007	Reset the Keyboard/Reader Data-Ready flag and the Printer/Punch Ready flag. Set the TIEFF. Assert the RESET bus line low.

* - The TTY reader is automatically disabled after fetching each character.

TABLE 11-2

Instruction Set for 12560 Serial Interface Module
(using typical device codes 03 and 04)

instruction pulse, with negative sense, is noted on the schematic at the corresponding pins on U10 (device codes 03 and 04 have been assumed). Comparing the information given in Table 11-2 with the subsequent logical action caused by each of these pulses will make operation of the interface clear. The contents of PROM U10 are described in Table 11-3.

For example, instruction 6030 resets the Keyboard/Reader Data-Ready flag by pulling pin 18 high on the UART, through gates U16d and U24b. Note that, as determined by gates U17b and U12c, this action occurs during the "read" portion of the IOTA cycle, when XTC is high. Instruction 6036 performs

All locations programmed to produce a logic 1 (high) output, except:

INPUTS					OUTPUTS							
ADDR-E	ADDR-D	ADDR-C	ADDR-B	ADDR-A	DO8	DO7	DO6	DO5	DO4	DO3	DO2	DO1
1	0	1	1	0	0	0	0	1	1	1	1	1
1	0	1	0	1	1	1	1	1	1	1	1	0
1	0	1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	0	1	0	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	1	0	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1
0	1	1	1	0	1	1	0	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	1	0	1
0	1	1	0	0	1	1	1	0	1	1	1	1
0	1	0	1	0	1	1	0	1	1	1	1	1
0	1	0	0	1	1	1	1	1	1	0	1	1
0	1	0	0	0	1	1	1	1	0	1	1	1

ADDR-A=pin 10 ADDR-B=pin 11 ADDR-C=pin 12 ADDR-D=pin 13 ADDR-E=pin 14
DO1 =pin 1 DO2 =pin 2 DO3 =pin 3 DO4 =pin 4 DO5 =pin 5
DO6 =pin 6 DO7 =pin 7 DO8 =pin 9

TABLE 11-3
PROM Device U10 Programming

this same operation, but also enables the reader to fetch the next character (by pulling low pin 10 on U14b), enables the UART to drive the DX bus lines with the "read" data during state T2 and T3 when XTC is high (by pulling pin 15 low on U21 and U22), and asserts the C0 and C1 bus lines low to cause the CPU to read the DX data as a jam-transfer into the Accumulator.

Similarly, instruction 6041 causes the next instruction to be skipped if the Printer/Punch Ready flag (U15a) is set, by aserting the SKP line on the bus, through gates U11b, U18b and U9b.

Instruction 6007 is decoded by gates U7b, U8b and U7c. The latter gate provides the drive to Q1, which in turn asserts the bus RESET line.

11.1.6 OTHER LOGIC OPERATION

Jumper-pad set A-B-C allows the UART to be operated from either a 5- or 10-volt power-supply level. Normal 5-volt operation is selected by soldering a jumper from pad A to pad B. When higher-speed operation is desired of the UART (not required for 4 MHz CPU operation), it can be operated from the +10-volt bus line by placing the jumper from pad A to pad C. When using 10 volts on the UART, devices U13, U17, U23 and U24 must be made by either Fairchild or Motorola.

The framing-error output from the UART is available at pin 14 of U20. This pin will be asserted high whenever the UART detects the absence of a stop-bit at its proper position. This condition can be forced, for example, by pushing the BREAK key on the terminal keyboard. By installing jumper X-Y, a framing error can be used to assert bus line 62 to the low state. This signal can then be used with logic external to the 12560 module to perform special functions, such as generation of a "panic" interrupt. An open-collector driver, U9c, is used to drive bus line 62 so that several such drivers can be wire-OR'ed together on the bus.

Jumper option L-M is provided so that the normal serial-interface functions of the 12560 module (but not the ability to decode and act on instruction 6007) can be disabled by assertion (lifting) of the front-panel AUX switch when the L-M jumper is installed. This jumper is normally omitted until the 12080 Audio Cassette Interface, or other module requiring the ability to disable the serial interface, is installed in the system.

11.1.7 SUMMARY OF JUMPER OPTIONS

Listed below is a summary of the functions of each of the jumper-pad

groups on the 12560 interface. Note that (to be consistent) for each of the jumper options on the 12560 module which also existed on the earlier 12060 interface, the lettering of each jumper pad is the same on both modules (except that the lettering of pads J and K is reversed, a trivial difference). Note that the jumper pad group A-B-C has a new function on the 12560 module, and that there is no D-E jumper pair on the 12560 interface. Further, jumper groups N-P-Q, R-T and X-Y did not exist on the 12060 interface module.

The jumper-pad group A-B-C is for selection of 5- or 10-volt UART operation. See Section 11.1.6.

Jumper-pad group F-G-H is for determination of the number of stop bits to be used in asynchronous communication format. See Section 11.1.1.

Jumper-pad pair J-K is for completion of the 20-ma input current loop. This jumper must be present when an RS-232 terminal is to be employed and no TTY terminal is connected between pins 13 and 14 of J1. See Section 11.1.1.

Jumper pair L-M is for disabling the 12560 interface from the front panel AUX switch. See Section 11.1.6.

Jumper-pad group N-P-Q is for selection of the function of the auxiliary RS-232 output. See Section 11.1.4.

Jumper R-T is installed only when the receive and transmit serial data rates are to be identical. See Section 11.1.1.

Jumper X-Y is for activation of the break-on-framing-error function. See Section 11.1.6.

11.2 TELETYPE TERMINAL SELECTION AND MODIFICATION

To achieve compatibility with the PDP-8 "style" architecture of the PCM-12, some small modifications may be required to the user's Teletype terminal. Selection of a Teletype terminal, and the required modifications, are described in this section.

11.2.1 SELECTION OF A TELETYPE MACHINE

There are many types and configurations of Teletype terminals available today. The specific machines recommended for operation with the PCM-12 are the Teletype Corporation Models 33-ASR and 35-ASR. (These include a paper-tape reader and punch. If the latter equipment is not required, a Model 33- or 35-KSR is recommended. These models simply omit the paper-tape equipment, providing only a printer and keyboard.) The Model 35 incorporates essentially the same features as the Model 33, but is a heavy-duty unit made

for round-the-clock operation. Since the Model 33 is considerably less expensive, and entirely adequate for most purposes, it will be chosen by most users.

If a new 33-ASR terminal is selected, the user should order Teletype Corp. catalog number 3321-3JC. This is one of the least expensive 33-ASR models, and is easily adapted for operation with the PCM-12. The 3321-3JC incorporates a paper-tape punch, manually-controlled reader, 60-Hz synchronous motor, friction-type paper-feed, a pedestal mount, and the ME typewheel with DSL keytops. (The latter keyboard/printer style corresponds closely to that shown in DEC documentation for the PDP-8.) The corresponding catalog number for the KSR model is 3311-3EC. The price of the 3321-3JC is approximately $1200.00 when purchased new.

The modification instructions provided below apply to the 3321-3JC. Since there are so many different types of Teletype terminals, it is impossible to give modification instructions for each. The user planning to employ some other model terminal should seek the assistance of a competent Teletype serviceman. Provided with the instructions given herein, he should be able to make the corresponding modifications to any Model 33- or 35-ASR or KSR. (Actually, the instructions apply rather broadly to many 3320/3321 and 3310/3311 model machines. Armed with the wiring diagram for his machine, the user can usually achieve the intent of the modifications for most terminals by tracing the wiring through the machine. The serviceman might be treated as a last resort.)

11.2.2 TELETYPE TERMINAL MODIFICATIONS

To achieve PDP-8/PCM-12 campatibility, modifications to the Teletype's wiring may be required in four areas. Each of these is discussed below.

However, before executing the following modification instructions for his Teletype terminal, the user should read each instruction carefully and decide whether or not it is actually needed for his particular terminal. If the terminal was originally purchased for use with a computer, the intent of one or more of the modifications may already have been achieved, since all computer-oriented equipment requires Teletype terminal modifications in order to adapt the TTY terminal to computer I/O operation. (The Teletype terminal was originally designed for relatively high-voltage, high-current, current-loop operation, which is obviously not compatible with the low-voltage, low-current operation of computer circuitry.)

In particular, the user may find his TTY terminal already equipped with a reader-control relay (see Section 11.2.2.3). If so, and if the rated coil voltage of the relay is 5 to 12 volts, it is likely that the instructions of Section 11.2.2.3 can be skipped - simply connect the relay coil leads to pins 1 and 2 of the rear-panel Molex connector, as indicated in Figure 11-2.

Also, the TTY terminal may already have a reed-relay or opto-isolator installed for distributor isolation. If so, this would make step 3 of Section 11.2.2.2 unnecessary. Just hook the relay contacts or opto-isolator output to pins 5 and 6 of the rear-panel Molex connector. (If the isolator is a solid-state device, be sure to observe proper current polarity. Note that pin 5 of the Molex connector is a current source, while pin 6 is a sink - see Figure 11-1.)

Actually, although not recommended when long-term reliability is an important consideration, the user can choose to omit the distributor-isolating circuitry entirely and still find that his TTY terminal will operate properly. The major purpose of the opto-isolator circuitry shown in Figure 11-1 is to allow the TTY distributor to operate at Teletype Corporation recommended levels (70 volts and 40 milliamps), while signalling to the computer at low-voltage, solid-state compatible levels. High-voltage operation on the TTY side assures that the distributor contacts will stay clean and noise-free. If relatively low-voltage operation of the TTY distributor can be tolerated (17 volts is the open-circuit potential at the current-loop input terminals on the 12560 interface), simply omit all the circuitry of Figure 11-1, and connect pins 5 and 6 of the rear-panel Molex connector directly to terminals 3 and 4 of Terminal Strip 151411 on the TTY terminal (polarity is not important for this connection). Do not skip steps 1 and 2 in Section 11.2.2.2, however.

To remove the outer cover from the Teletype, remove the knob on the front-panel Mode Switch, and the knob on the paper roller. Loosen the thumbscrews on the back of the terminal. Remove the paper and tape from the terminal. Remove the small screw on the reader cover. Remove the metal front-panel faceplate by sliding it down and away from the machine. Under the faceplate are four screws - loosen them. The cover can now be lifted off the machine.

11.2.2.1 FULL-DUPLEX OPERATION

Wire the Teletype for full-duplex operation by moving the

BROWN/YELLOW wire from terminal 3 to terminal 5, and the WHITE/BLUE wire from terminal 4 to terminal 5, on Terminal Strip 151411. This terminal strip is easily located at the rear of the TTY, just below the bank of nylon Molex connectors. Viewed from the rear of the terminal, pin 9 on this terminal strip is at the right-hand end. See Figure 11-4.

11.2.2.2 20-MA OPERATION

The terminal should be modified for 20-ma (rather than 60-ma) current-loop operation. This is accomplished by:

1. Remove the PURPLE wire connected to terminal 8 on Terminal Strip 151411, attach it to terminal 9. This sets the receive-side of the TTY to expect and accept 20 ma inputs.
2. Locate R1, the large flat-style power resistor which is bolted to the mounting plate of the call-control unit on the right-hand side of the terminal. Note that this resistor has four terminals. Ascertain that the BLUE wire is attached to terminal 1, the inboard terminal (i.e., the terminal closest to the center of the machine). If necessary, move the BLUE wire to this terminal. This sets the loop current at 20 ma when the TTY terminal is operating in LOCAL mode. (See Figure 11-3 for location of resistor R1.)
3. Establish an isolated circuit for data traveling to the 12560 interface from the TTY terminal, according to Figure 11-1:

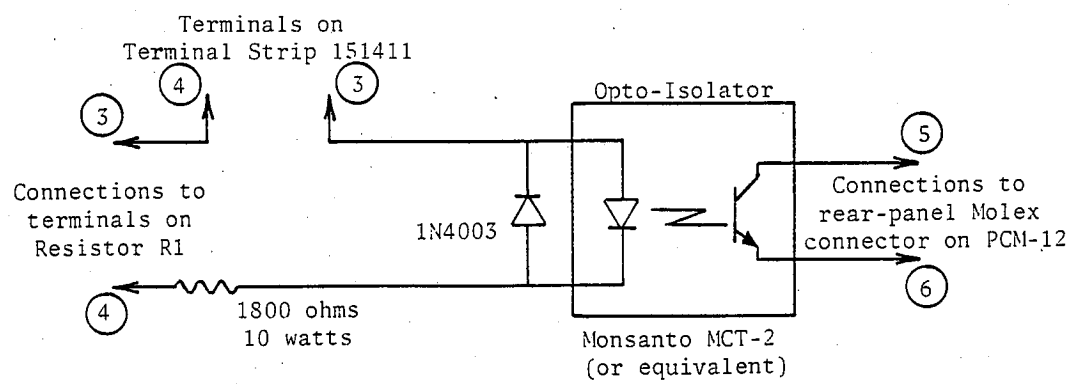


FIGURE 11-1

Optical-Isolation of TTY Distributor

This circuitry is active in the LINE mode of TTY operation, when the TTY is sending data to the PCM-12. The optical isolator protects against noisy operation of the distributor mechanism in the TTY. Mount the components securely inside the TTY enclosure. Provide adequate ventilation or heat-sinking for the 1800-ohm resistor, since it will run warm. Terminal 3 on resistor R1 can be located by counting over from terminal 1, which was located in step 2, above. Terminal 4 is the outboard terminal. Provide for strain-relief of the two wires which run to the computer.

11.2.2.3 READER ENABLE RELAY

To prevent the tape-reader from over-running the computer's ability to accept data, it is necessary that the computer have the ability to disable the reader. This is accomplished by adding the reader-control relay. By means of this relay, the KCC and KRB instructions enable the reader to fetch a character. Then as soon as the character is sent to the 12560 interface from the reader, the reader is automatically disabled, by flip-flop U14b on the interface, until another KCC or KRB instruction is received. Wire the reader-control relay according to Figure 11-2:

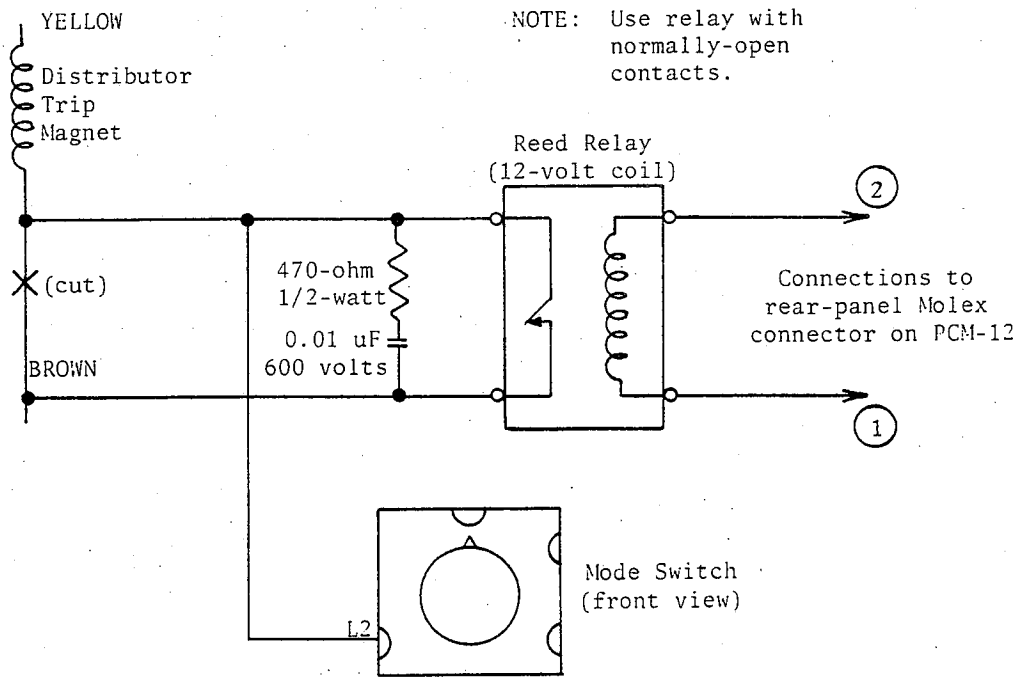


FIGURE 11-2

Connections for Reader-Control Relay

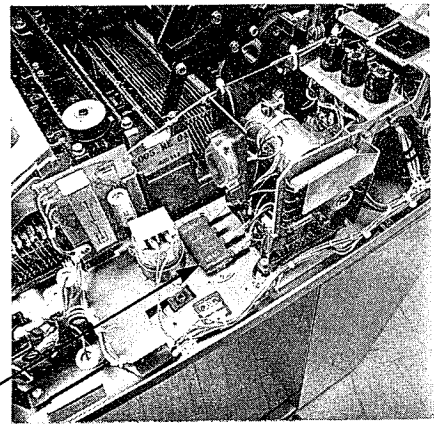


FIGURE 11-3

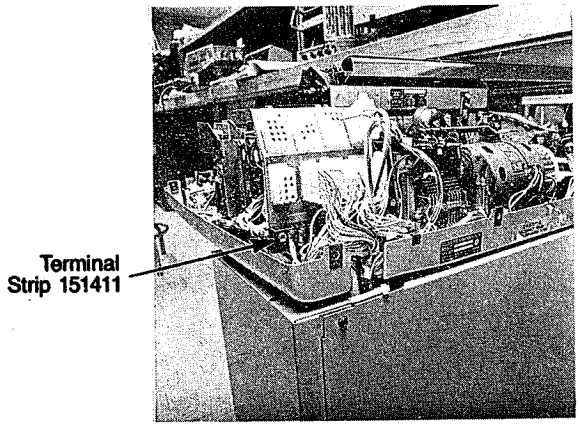


FIGURE 11-4

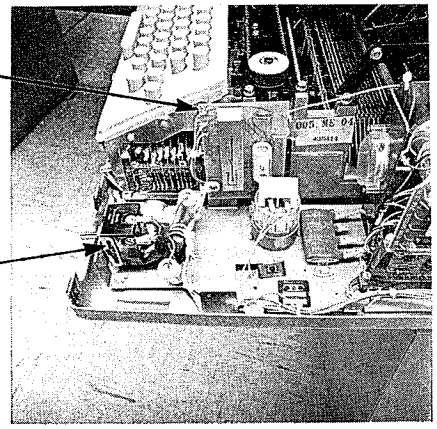


FIGURE 11-5

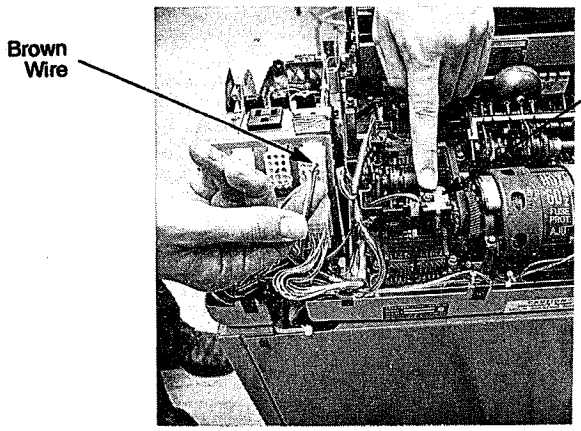


FIGURE 11-6

Locate the Distributor Trip Magnet at the rear of the TTY terminal. See Figure 11-6. Note that a BROWN and a YELLOW wire emerge from it. Break the BROWN wire and connect the normally-open contacts of the reader-control relay in series with it. Then connect a wire from terminal L2 (physically located as shown in Figure 11-2) of the Mode Switch to the portion of the BROWN wire that is still attached to the Distributor Trip Magnet. The Mode Switch is shown in Figure 11-5. Provide strain-relief for the two wires going to the computer.

11.2.2.4 KEYBOARD PARITY

For PDP-8 compatibility, the keyboard (but not the reader) must transmit bit 8, the most significant bit in each ASCII character, as a "mark" or "one". In other words, the parity function is defeated, and the parity bit is always asserted, in every character. To make the TTY function in this manner, the proper connection must be made to the keyboard contact blocks at either end of the keyboard. The connections are simple, and require only the moving of jumpers.

Viewing the terminal from the front, examine the contact block at the left end of the keyboard, and note that a WHITE/BLACK wire is attached to one of the contacts on the block. Also note that a GREEN wire is spliced to the WHITE/BLACK wire. This GREEN wire should be connected to the other terminal on the block. If a RED/GREEN wire is already attached to the other terminal, remove it, cover the end with electrical tape, and tie it to the keyboard frame with string. It will be left floating. Now connect the GREEN wire to the exposed terminal.

At the right-hand end of the keyboard a second terminal block will be found. Ascertain that a GREEN wire is attached to the upper left-hand terminal. If not, locate the GREEN wire and attach it to that terminal. (The GREEN wire may be found, unused, lashed to the side of the keyboard frame.)

If your keyboard does not correspond to the description above, you do not have the usual keyboard supplied with the 3321-3JC terminal, and must consult a Teletype serviceman to determine how to strap the machine for "level-8 always mark". (When accomplishing this conversion, do not let the serviceman modify the machine so that the reader also always marks bit 8; the reader should read a tape exactly as it is punched, in all 8 channels.)

11.2.2.5 RECEIVE-SIDE CONNECTIONS TO 12560 INTERFACE

It is now only necessary to make the receive-side connections to the TTY from the computer. To do this, connect two wires between the TTY and the PCM-12 in the following way:

1. From terminal 7 on Terminal Strip 151411 to pin 8 of the PCM-12's rear-panel Molex connector.
2. From terminal 6 on Terminal Strip 151411 to pin 3 on the rear-panel Molex connector.

Congratulations - you now have a fully-compatible TTY terminal. While you are at it, make sure the automatic carriage-return/line-feed function is disabled on your terminal, and that the margin bell is enabled. The instruction manual for your terminal explains how to select these options.

You now have a six-wire cable running to your PCM-12 computer from the TTY terminal, two wires for send, two for receive, and two for the reader-control relay. Carefully go back over the modifications you have made to the TTY terminal, and the connections to the computer, and make sure everything is correct. Then replace the cover on the terminal.

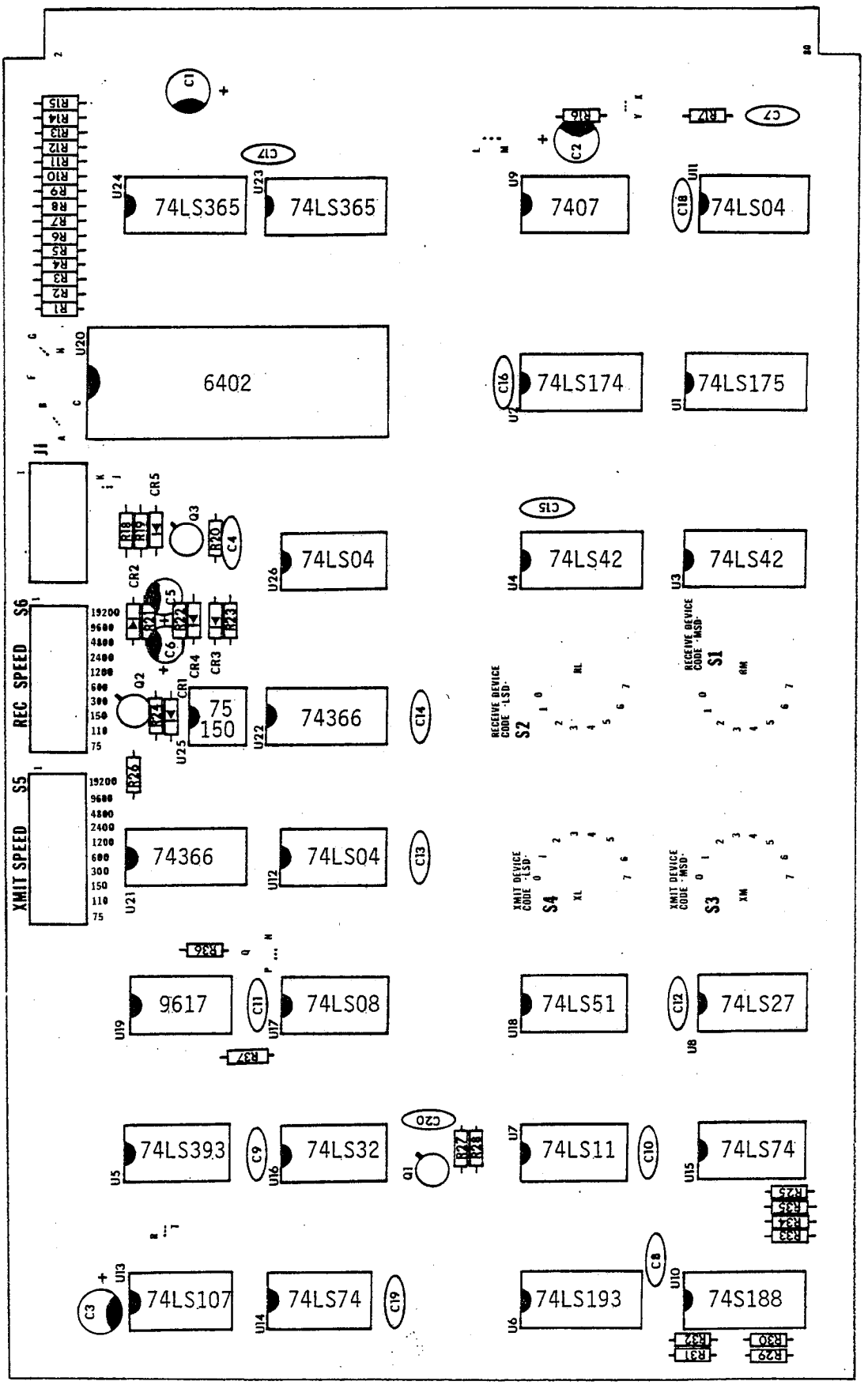


FIGURE 11-7
Printed-Circuit Board Layout
12560 Module

SECTION 12.0

12080 AUDIO CASSETTE RECORDER INTERFACE MODULE - ASSEMBLY

The 12080 interface module is an add-on accessory for the PCM-12 computer. The following instructions assume that assembly and test of the basic computer have already been completed. These instructions form a supplement to the PCM-12 assembly manual, and should be added to the three-ring binder which houses those instructions. After assembly is complete, don't discard these instructions - you will want to refer to them from time-to-time in the future.

( ) Locate the 12080 module printed-circuit board. Very carefully inspect the board for:

a. Open or broken traces.

b. Any trace which might be shorted to an adjacent trace or pad.

For any questionable-appearing trace, compare the printed-circuit trace routing with that indicated on the schematic diagram, Drawing 12082 in the PCM-12 System Operating Manual (the supplementary material for the operating manual was provided with this kit).

( ) Note that the printed-circuit board carries a white-epoxy component-placement legend on one side. This is the component side of the board.

( ) Locate the bag containing the parts kit for the 12080 module. Check off each part against the following list:

INTEGRATED CIRCUITS

( )	1 ea	74LS00	or 9LS00	(14-pin DIP)
( )	2 ea	74LS03	or 9LS03	(14-pin DIP)
( )	1 ea	74LS04	or 9LS04	(14-pin DIP)
( )	1 ea	74LS08	or 9LS08	(14-pin DIP)
( )	1 ea	74LS11	or 9LS11	(14-pin DIP)
( )	1 ea	74LS32	or 9LS32	(14-pin DIP)
( )	2 ea	74LS74	or 9LS74	(14-pin DIP)
( )	1 ea	74C74		(14-pin DIP) *
( )	1 ea	74C76		(16-pin DIP) *
( )	1 ea	74C86		(14-pin DIP) *

( )	2 ea	74LS138	or 9LS138	(16-pin DIP)
( )	1 ea	74LS174	or 9LS174	(16-pin DIP)
( )	1 ea	74S287		(16-pin DIP marked with BLACK dot)
( )	2 ea	74365	or 8095	(16-pin DIP)
( )	1 ea	4018	or 14018	(16-pin DIP) *
( )	1 ea	4029	or 14029	(16-pin DIP) *
( )	2 ea	4040	or 14040	(16-pin DIP) *
( )	1 ea	4046	or 14046	(16-pin DIP) *
( )	1 ea	LM311		(8-pin DIP)
( )	1 ea	LM747		(14-pin DIP)
( )	1 ea	P-6431		(16-pin DIP marked with BLUE dot) *
( )	1 ea	TR-1602 UART		(40-pin DIP)

* - MOS integrated circuit. Do not remove from conductive foam until ready for insertion in proper socket.

RESISTORS AND CAPACITORS

( )	1 ea	1K ohm resistor (color code: BROWN-BLACK-RED)
( )	9 ea	2.2K ohm resistor (RED-RED-RED)
( )	7 ea	10K ohm resistor (BROWN-BLACK-ORANGE)
( )	1 ea	15K ohm resistor (BROWN-GREEN-ORANGE)
( )	4 ea	20K ohm metal-film resistor
( )	2 ea	27K ohm resistor (RED-VIOLET-ORANGE)
( )	1 ea	47K ohm resistor (GRAY-RED-ORANGE)
( )	3 ea	100K ohm metal-film resistor
( )	2 ea	133K ohm metal-film resistor
( )	1 ea	1M ohm resistor (BROWN-BLACK-GREEN)
( )	1 ea	4.3M ohm resistor (YELLOW-ORANGE-GREEN)
( )	1 ea	50K ohm cermet trimmer potentiometer
( )	1 ea	100 pF dipped-mica capacitor
( )	1 ea	330 pF dipped-mica capacitor
( )	1 ea	1000 pF dipped-mylar capacitor
( )	3 ea	2200 pF disc-ceramic capacitor
( )	1 ea	0.05 uF (or 0.047 uF) dipped-mylar capacitor
( )	21 ea	0.01 uF disc-ceramic capacitor
( )	2 ea	0.22 uF dipped-epoxy tantalum capacitor
( )	4 ea	10 uF dipped-epoxy tantalum capacitor

OTHER COMPONENTS

( )	1 ea	E-271 field-effect transistor
( )	3 ea	2N2222 NPN transistor
( )	1 ea	1N4148 diode
( )	1 ea	8-pin integrated-circuit socket
( )	12 ea	14-pin integrated-circuit socket
( )	13 ea	16-pin integrated-circuit socket
( )	1 ea	40-pin integrated-circuit socket
( )	1 ea	audio transformer (1.2K ohm: 4 ohm)

MISCELLANEOUS

( )	1 ea	ribbon-cable with connector and matching socket attached
( )	1 ea	panel-mounting Molex connector housing (with mounting "ears")
( )	1 ea	cable-mounting Molex connector housing
( )	8 ea	male Molex "pins"
( )	8 ea	female Molex "pins"
( )	1 ea	80-pin edge connector
( )	1 ea	T-shaped edge connector key
( )	2 ea	4-40 x 1/2" binder-head machine screw

( ) Carefully mount and solder 16-pin integrated-circuit sockets at U2, U3, U5, U6, U12, U13, U16 - U21, and U23. Note that the orientation of pin 1 on all integrated circuits except U27 is toward the top of the printed-circuit board (gold "fingers" on the right and legend-side up). Refer to Figure 12-1.

( ) Similarly, mount and solder 14-pin integrated-circuit sockets at U1, U4, U7 - U11, U15, U22, and U24 - U26.

( ) Mount and solder the 8-pin integrated-circuit socket at U14.

( ) Mount and solder the 40-pin integrated-circuit socket at U27. Remember that pin 1 on this integrated circuit goes toward the bottom of the board - opposite to all other integrated circuits.

( ) Mount and solder the 1N4148 diode at CR1. Observe that diode polarity is marked on the printed-circuit board. Mount the diode with the polarity indicated:

( ) Mount and solder the 1K ohm resistor at R5.

( ) Mount and solder 2.2K ohm resistors at R1, R2, R14, R16, R18, R25 and R31 - R33. Make sure you have mounted each resistor in the right place.

( ) Mount and solder 10K ohm resistors at R4, R13, R15, R17 and R26 - R28.

( ) Mount and solder the 15K resistor at R22.

( ) Mount and solder 20K metal-film resistors at R3, R6, R7 and R21.

( ) Mount and solder 27K ohm resistors at R12 and R24.

( ) Mount and solder the 47K ohm resistor at R8.

( ) Mount and solder 100K ohm metal-film resistors at R9, R11 and R20.

( ) Mount and solder 133K ohm metal-film resistors at R10 and R19.

( ) Mount and solder the 1M ohm resistor at R30.

( ) Mount and solder the 4.3M ohm resistor at R29.

( ) Mount and solder the 100 pF dipped-mica capacitor at C5.

( ) Mount and solder the 330 pF dipped-mica capacitor at C11.

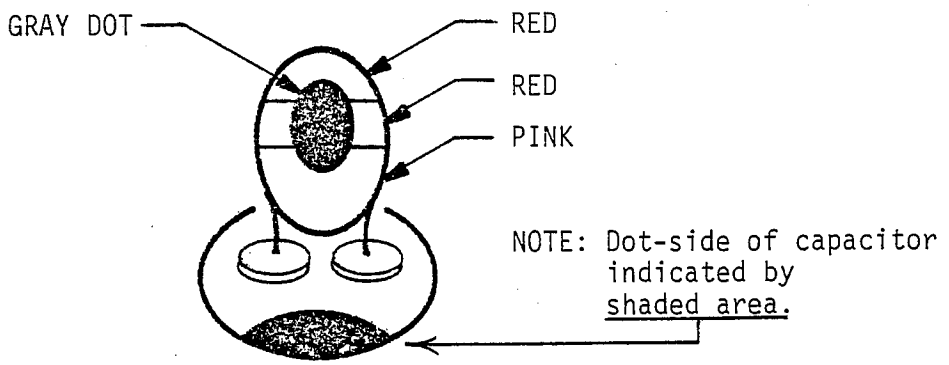
( ) Mount and solder the 1000 pF mylar capacitor at C6.

( ) Mount and solder 2200 pF disc ceramic capacitors at C2, C4 and C13.

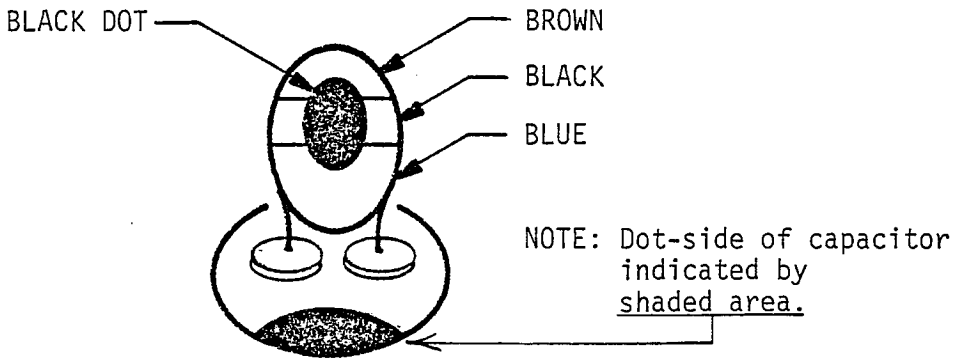
( ) Mount and solder the 0.05 (or 0.047) uF dipped-mylar capacitor at C10.

( ) Mount and solder 0.01 uF disc-ceramic capacitors at C7, C8, C9, C17, and C18 - C34.

( ) Mount and solder 0.22 uF dipped-epoxy tantalum capacitors at C12 and C35. Note that each of these capacitors carries a polarity-indicating dot on its side. Proper orientation of each capacitor is with the dot-side mounted above the shaded area in the component silhouette:



( ) Similarly, mount and solder 10 uF tantalum capacitors at C3, and C14 - C16. Observe capacitor polarity as indicated below:



( ) Mount and solder the 50K ohm trimmer potentiometer at R23. The adjustment screw should face away from the gold fingers at the opposite end of the board.

( ) Mount and solder the E-271 field-effect transistor at Q1. The flatted side of the transistor body should mount above the flat side on the component silhouette.

( ) Mount and solder 2N2222 transistors at Q2 - Q4. To orient each transistor properly, make sure the tab on the transistor is mounted above the tab position shown in the component silhouette.

( ) Mount and solder the audio transformer at T1. Note that one side of the transformer is marked with a "P" (primary). This is the in-board side of the transformer, as shown on the component silhouette. The secondary-side of the transformer has one lead clipped off; only one end and the center-tap of this winding are used.

( ) Remove the special 16-pin socket from the ribbon-cable connector, and solder it into place at the CABLE socket position shown in the component-placement legend. Pin 1 of this socket goes toward the gold-finger end of the board.

( ) If two stop-bit operation is selected, solder a jumper "strap" from pad A to pad B. For one stop-bit operation, solder the jumper between pads A and C (leaving pad B unconnected).

( ) For 300-Baud operation, solder a jumper from pad D to pad G, and another jumper from pad H to L. (For operation at other Baud rates, see Sections 10.2 and 10.4 of the System Operating Manual.)

( ) Decide if the 03/04 ENABLE H line should be held low on the interface module. If so, solder a jumper from pad P to pad N. If it is desired to control the 03/04 ENABLE H line from the front panel, the jumper should be soldered from pad P to pad M, leaving pad N unconnected. If it will never be desired to pull the 03/04 ENABLE H line low, then place no jumper at all at pads N, M and P. (See the operating manual for discussion of the M - P jumper pad set.)

( ) If the module is to respond to device codes 01 - 04, solder a jumper from pad R to pad S. If, instead, the module is to respond to device codes 11 - 14, solder the jumper from pad R to pad T, leaving pad S unconnected.

( ) Carefully plug the 74LS00 (or 9LS00) integrated circuit into its socket at U9. Pin 1 on each integrated-circuit package is marked with a dot or a notch in the I.C. body at the pin-1 end. Pin 1 of each integrated circuit except U27, the UART, goes toward the top of the printed-circuit board.

( ) Similarly, mount the following integrated circuits:

( )	74LS03	(or 9LS03)	at U1 and U10.
( )	74LS04	(or 9LS04)	at U24.
( )	74LS08	(or 9LS08)	at U25.
( )	74LS11	(or 9LS11)	at U8.
( )	74LS32	(or 9LS32)	at U26.
( )	74LS74	(or 9LS74)	at U4 and U7.
( )	74C74		at U15.
( )	74C76		at U20.
( )	74C86		at U11.
( )	74LS138	(or 9LS138)	at U5 and U6.
( )	74LS174	(or 9LS174)	at U2.
( )	74S287	(BLACK dot)	at U3.
( )	74365		at U18 and U19.
( )	4018	(or 14018)	at U21.
( )	4029	(or 14029)	at U13.
( )	4040	(or 14040)	at U17 and U23.
( )	4046	(or 14046)	at U16.
( )	LM311		at U14.
( )	LM747		at U22.
( )	P-6431	(BLUE dot)	at U12.

( ) Mount the TR-1602 UART device in its socket at U27. Make certain that you have pin 1 on this integrated circuit oriented toward the bottom of the board.

( ) Carefully inspect each integrated circuit to make sure its pins are properly inserted in its socket, and that pin 1 is properly oriented.

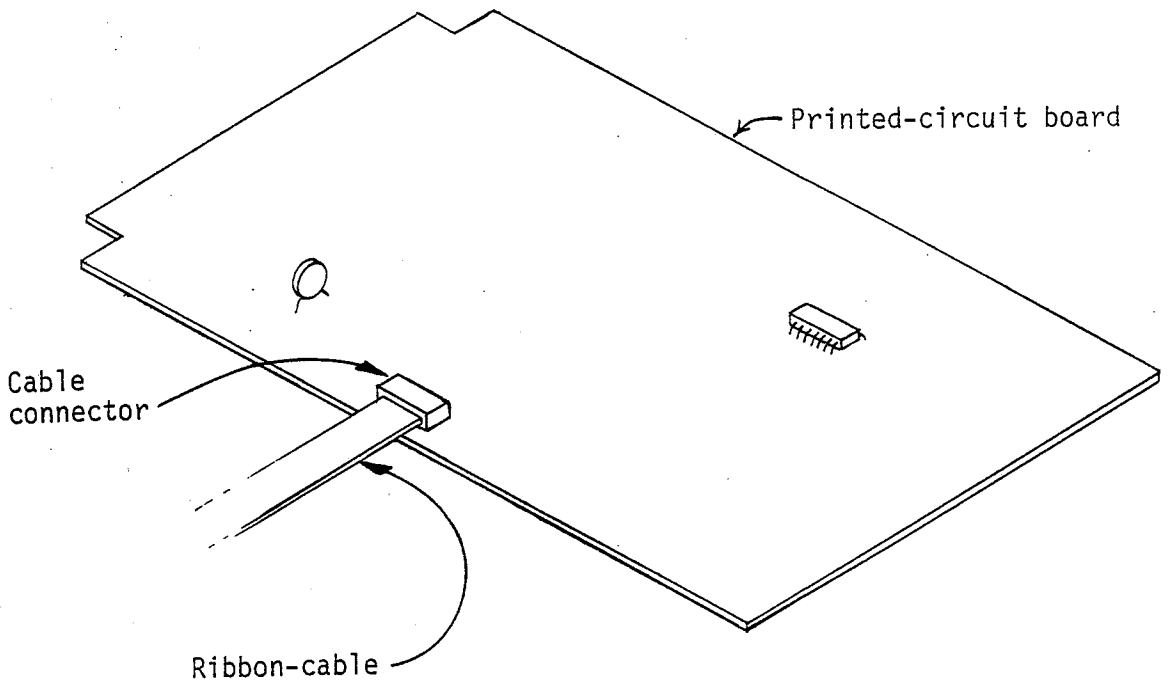
( ) Check each diode and tantalum capacitor to make sure it is properly oriented. Polarity of each of these components is very important.

( ) Check each transistor to make sure its orientation-indicating tab is mounted above the tab indicated on the component silhouette. The transistors will not operate properly if they are not mounted correctly.

( ) Very carefully inspect each solder connection on the board to make sure it is a good (not "cold-solder") joint, and that it is not shorted to an adjacent trace or pad.

( ) This completes assembly of the 12080 module itself. The only remaining assembly requirement is to wire the interface module to the magnetic-tape recorder through a rear-panel mounting Molex connector. Refer to Section of the operating manual to familiarize yourself with proper connection of the interface module to the recorder.

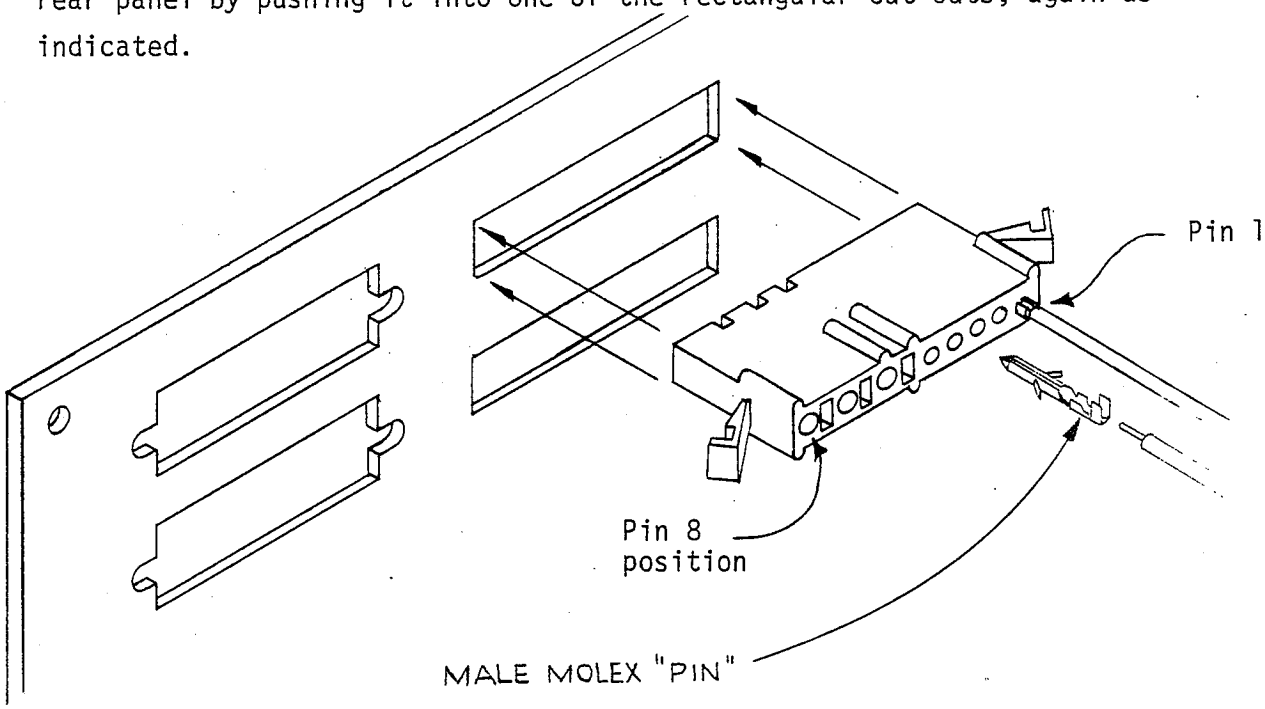
( ) Locate the ribbon cable with its attached male DIP connector. Refer to the detail below to determine the manner in which this cable plugs into the CABLE connector on the interface module. Now before proceeding to the next step, make sure you understand which wire in the ribbon-cable is connected to each pin on the CABLE connector.



( ) Only eight of the wires from the ribbon-cable can be accomodated by the rear-panel mounting Molex connector. However, this is a sufficient number for all necessary connections to the audio tape recorder. The following table gives a suggested scheme for wiring the panel-mount Molex connector.

Connect this CABLE connector pin number	To this Molex connector pin number
16	1
2	2
15	3
4	4
5	5
6	6
7	7
8	8

( ) Refer to the detail below and connect the ribbon-cable wires to the rear-panel mounting Molex connector, according to the table in the previous step. Each selected wire in the ribbon-cable is soldered to a male Molex "pin" as shown. The pin is then mounted in the connector housing by pushing it in from the rear, as indicated. The connector housing itself mounts in the rear panel by pushing it into one of the rectangular cut-outs, again as indicated.



( ) You still have the other Molex connector housing and eight female Molex pins. left after the above assembly steps have been completed. These may be used to connect your audio cassette recorder to the rear-panel connector you just finished installing.

( ) Use the interface module to properly locate the T-shaped key in the 80-pin edge connector, and snap it into place. Then remove the backplane assembly from the computer, and install the new connector. Replace the backplane.

Assembly of the 12080 interface module is now complete. It can be slid into the appropriate slot in the computer card-cage, and the ribbon-cable plugged into the CABLE connector. When it is desired to remove the 12080 module from the computer, always be sure to first unplug the ribbon cable from the module to prevent damage to the connector.

Before attempting operation of the 12080 module for the first time, be sure to read Section 10 in the PCM-12 System Operating Manual to familiarize yourself with the operation of your new interface module.

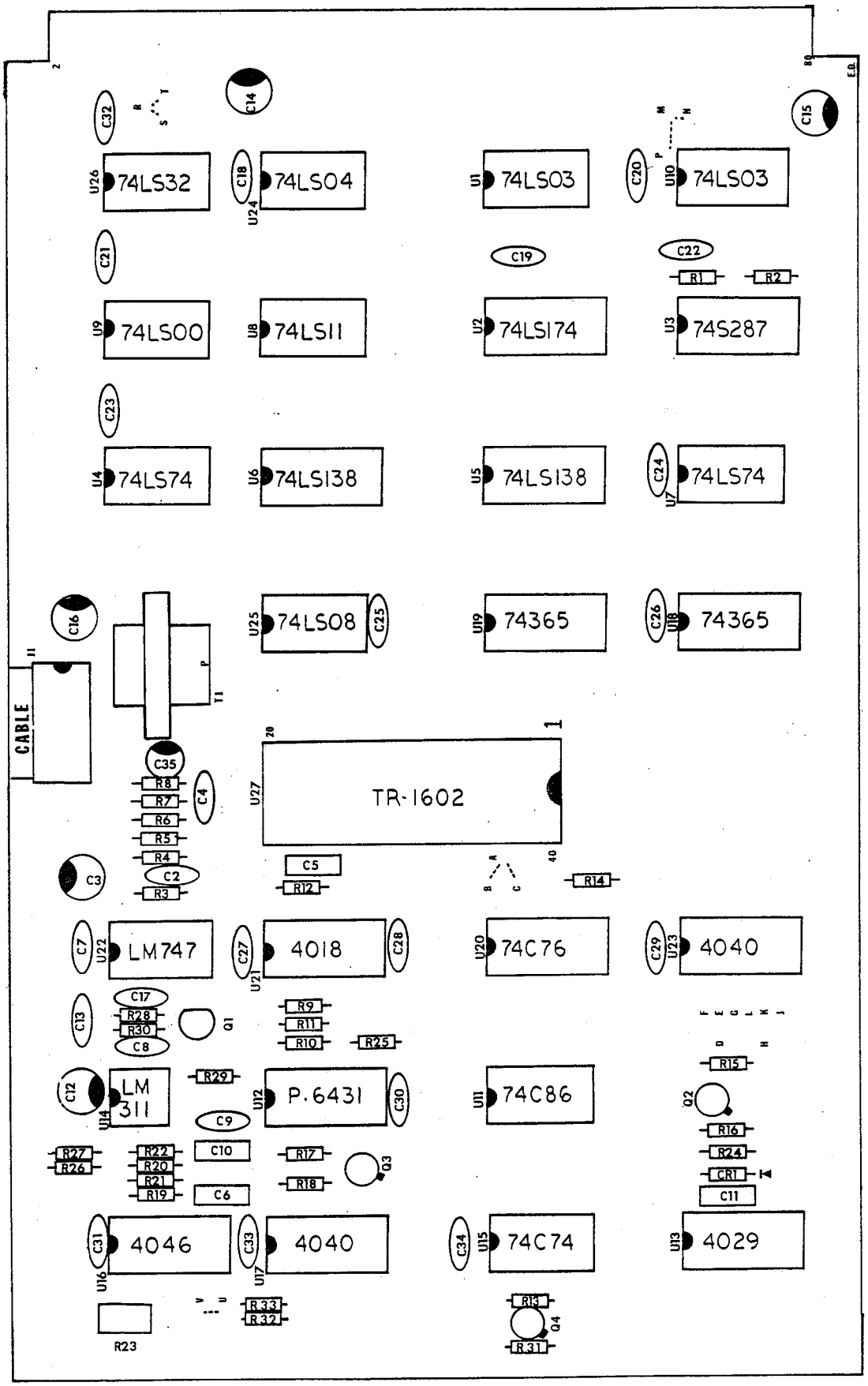

74LS32
74LS04
74LS03
74LS03
74LS00
74LS11
74LS174
74S287
74LS74
74LS138
74LS138
74LS74
74LS08
74365
74365
CABLE
TR-1602
LM747
4018
74C76
4040
LM 311
P.6431
74C86
4046
4040
74C74
4029

FIGURE 12-1

SECTION 13.0

12020-B STATIC MEMORY MODULE - ASSEMBLY

These instructions are for assembly of the 12020-B static memory module for the PCM-12 computer. This document forms a supplement to the PCM-12 assembly manual, and should be added to the three-ring binder which houses that manual. After assembly is complete, don't discard these instructions - you may want to refer to them at some point in the future.

( ) Locate the 12020-B memory module printed-circuit board. Very carefully inspect the board for:

a. Open or broken traces.

b. Any trace which might be shorted to a neighboring trace or pad.

For any questionable-appearing trace, compare the printed-circuit trace routing with that indicated on the schematic diagram, Drawing 12022-B.

( ) Note that the printed-circuit board has an epoxy-paint legend on one side. This is the component side of the board.

( ) Locate the bag containing the parts kit for the 12020-B module. Check off each part against the following list:

INTEGRATED CIRCUITS

( ) 48 ea	2102-family memory device	(16-pin DIP)
( ) 3 ea	74LS04 or 9LS04	(14-pin DIP)
( ) 1 ea	74LS00 or 9LS00	(14-pin DIP)
( ) 2 ea	74LS42 or 9LS42	(16-pin DIP)
( ) 2 ea	8096 or 74366	(16-pin DIP)
( ) 3 ea	74LS375 or 9LS375	(16-pin DIP)

CAPACITORS

( ) 11 ea 0.01 uF disc ceramic capacitor
( ) 1 ea 10 uF tantalum capacitor (dipped-epoxy type)

SOCKETS AND MISCELLANEOUS

( ) 4 ea 14-pin integrated-circuit socket
( ) 55 ea 16-pin integrated-circuit socket
( ) 1 ea 80-pin edge connector
( ) 1 ea T-shaped edge-connector key
( ) 2 ea 4-40 x 1/2" binder-head machine screw

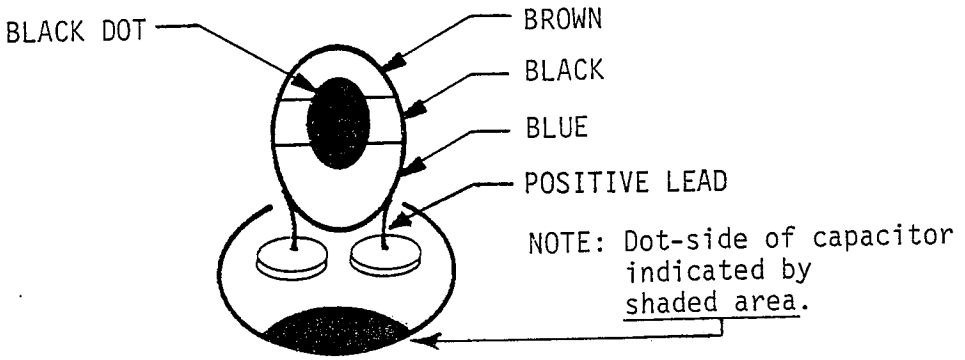
( ) Refer to Figure 13-1. Carefully mount 16-pin I.C. sockets at U1, U4 - U8, U11, and U12 - U59. Note that the orientation for all integrated circuits is with pin 1 toward the top of the board (gold fingers on the right). Carefully solder all socket pins to their respective pads.

( ) Similarly, mount and solder 14-pin I.C. sockets at U2, U3, U9 and U10.

( ) Carefully check your soldering work for possible shorts at each socket pin.

( ) Mount and solder 0.01 uF disc ceramic capacitors at C1 - C11. For best appearance, mount these capacitors as close as possible to the board.

( ) Mount and solder the 10 uF tantalum capacitor at C12. If this capacitor has one lead identified with a "+" mark, be sure to align it with the corresponding "+" mark on the printed-circuit board, to properly observe capacitor polarity. If the capacitor is the type with color bands on it, observe that it carries a polarity-indicating dot on one side. The dot-side is indicated by the shaded portion of the component silhouette on the printed-circuit board. See mounting detail, next page.



( ) Program the memory module to the desired field of operation by soldering a jumper at the proper location in the FIELD SELECT switch silhouette, as described in Section 6.2 of the PCM-12 System Operating Manual. (Also see Section 4 of the System Operating Manual for discussion of PCM-12 memory organization.) See Table 13-1 on next page.

( ) Carefully plug the 74LS00 (or 9LS00) device into its socket at U10. Pin 1 on the integrated-circuit package is marked with either a dot or a notch in the I.C. body at the pin-1 end. Pin 1 of each I.C. goes toward the top of the pronted-circuit board.

( ) Similarly, plug the following integrated circuits into the indicated sockets:

( ) 74LS04 (or 9LS04) at U2, U3 and U9.
( ) 74LS42 (or 9LS42) at U1 and U11.
( ) 74366 (or 8096) at U7 and U8.
( ) 74LS375 (or 9LS375) at U4, U5 and U6
( ) 2102-family memory devices at all remaining socket positions

( ) Carefully inspect each integrated circuit to make sure its pins are properly inserted in its socket, and that pin 1 is properly oriented.

( ) Carefully inspect each soldered connection on the board to assure it is a good joint, and that it is not shorted to an adjacent pad or trace.

( ) Remove the backplane assembly from the computer, and solder the new edge connector into one of the unused positions. Use the 4-40 screws to mount the edge connector, and remount the backplane assembly into the computer.

Assembly of the memory module is now complete, and it can now be inserted into the computer's bus and operated. The test programs furnished in paper-tape form with this kit may be used to test the memory module for proper operation. Refer to the documentation furnished with the tapes for instructions on the operation of the test programs.

TABLE 13-1

FIELD SELECT Programming

FIELD	JUMPER STRAP	
	From Pin #	To Pin #
Ø	1	16
1	2	15
2	3	14
3	4	13
4	5	12
5	6	11
6	7	10
7	8	9

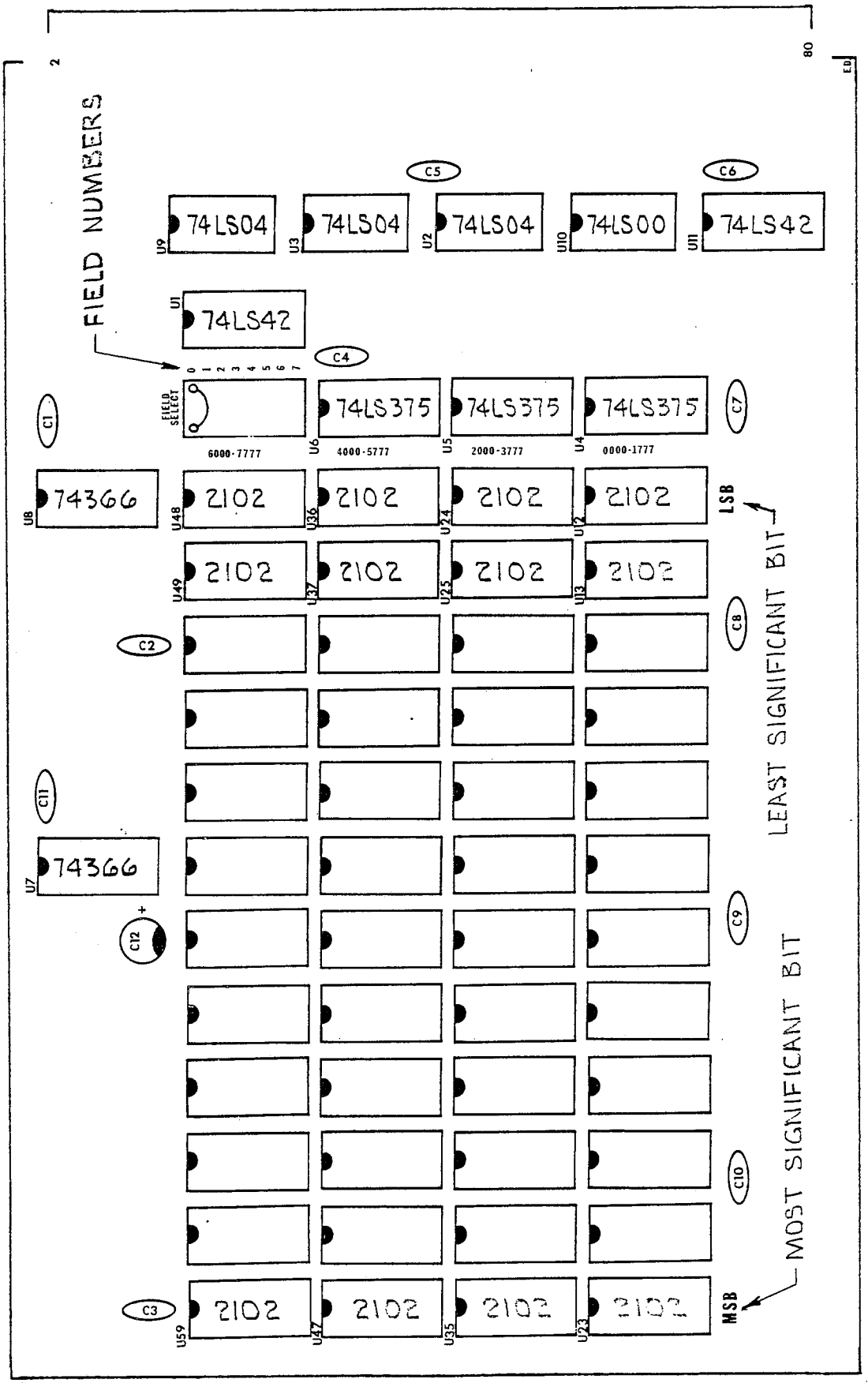

2
80
ED
FIELD NUMBERS
C5
C6
74LS04
74LS04
74LS04
74LS00
74LS42
U9
U3
U2
U10
U11
U1
74LS42
C4
0 1 2 3 4 5 6 7
FIELD SELECT
74LS375
74LS375
74LS375
C1
C7
6000-7777
4000-5777
2000-3777
0000-1777
U6
U5
U4
U8
74366
2102
2102
2102
2102
LSB
U48
U36
U24
U12
2102
2102
2102
2102
U49
U37
U25
U13
C8
C2
C11
U7
74366
C12
C9
C10
C3
2102
2102
2102
2102
MSB
U59
U47
U35
U23
LEAST SIGNIFICANT BIT
MOST SIGNIFICANT BIT

FIGURE 13-1