EK-KD11A-MM-001

KD11-A processor maintenance manual

digital equipment corporation · maynard, massachusetts

1st Edition, September 1973
2nd Printing, October 1973
3rd Printing, February 1974
4th Printing, August 1974
5th Printing, December 1974
6th Printing, March 1975

Printed in U.S.A.

CONTENTS

CONTENTS (Cont)

ILLUSTRATIONS

TABLES

CHAPTER 1
INTRODUCTION

1.1 SCOPE

This manual describes the KD11-A Processor which is the basic component of the PDP-11/35 and PDP-11/40 computer systems. The processor is connected to the Unibus as a subsystem and controls time allocation of the Unibus for peripherals, and performs arithmetic and logic operations through instruction decoding and execution. The information contained in this manual pertains primarily to the processor itself, however, certain processor options are also described (KY11-D, KJ11-A, KM11-A, and KW11-L).

Table 1-1 lists the other manuals that are necessary for a complete understanding of the basic PDP-11/35 or PDP-11/40 system.

Table 1-1
Related Documents

Title	Document Number	Remarks
PDP-11/35 (BA11-DA, DB 10-1/2" Mounting Box) System Manual	EK-11035-TM-001	Describes overall PDP-11/35 system and includes sections on installation, operation, and programming.
PDP-11/40, PDP-11/35 System Manual (21" Chassis)	EK-11040-TM-002	Describes overall PDP-11/40 system and includes sections on installation, operation, and programming.
KE11-E & KE11-F Instruction Set Options Manual	EK-KE11E-TM-002	Covers the KE11-E Extended Instruction Set and KE11-F Floating Instruction Set processor options.
KT11-D Memory Management Option Manual	EK-KT11D-TM-002	Covers the KT11-D Memory Management Option.

1.2 ORGANIZATION

The description of the KD11-A Processor is divided into four major sections: microprogramming (Chapter 2), block diagram (Chapter 3), flow diagrams (Chapter 4), and logic diagrams (Chapter 5).

Chapter 2 first discusses the processor and briefly covers the conventional method of implementing the instruction set. The remainder of the chapter is devoted to a discussion of microprogrammed implementation, the basic microprogram memory, and the structure of the microprogram word.

Chapter 3 describes the processor at a block diagram level and introduces the processor architecture by describing the basic block diagram which illustrates all of the major logic elements and interconnections within the processor. The narrative in this chapter is summarized by a table that lists each functional block on the diagram, describes the block, and lists inputs and outputs to and from that block.

Most of the information required to follow a sequence of machine states on a flow diagram is contained in the flow diagram itself. Chapter 4 is, therefore, divided into two major parts: The first part explains the format of the flow diagram, and the second part provides examples of tracing instruction operations through the flow diagrams.

Chapter 5 is the last section covering the KD11-A Processor. It provides a description of the processor logic and includes an explanation of print set conventions.

Chapter 6 of this manual provides a complete description of the KY11-D Programmer's Console, with the exception of operating procedures which are covered in the *PDP-11/40, PDP-11/35 System Manual (21" Chassis),* EK-11040-TM-002.

Chapter 7 describes three of the internal processor options that may be used with the KD11-A. These options are: the KW11-L Line Frequency Clock, the KJ11-A Stack Limit Register, and KM11-A Maintenance Console. The other available processor options (KE11-E, KE11-F, and KT11-D) are included in the manuals listed in Table 1-1.

A complete drawing set is supplied with this manual in a companion volume entitled *PDP-11/40 System Engineering Drawings.* The drawing set includes the basic block diagram, microword format, function tables, flow diagrams, and logic diagrams. Familiarity with the ISP notation (Paragraph 4.2 of the *PDP-11/40 System Manual*) as well as the print format (Paragraph 5.2 of this manual) will aid in understanding the prints.

CHAPTER 2
MICROPROGRAMMING

2.1 SCOPE

This chapter provides a general introduction to the microprogramming techniques used in the KD11-A Processor. Because microprogramming is the key to KD11-A Processor operation, it is essential to understand the basic techniques before attempting to use the block diagram, flow diagrams, and logic diagrams. This chapter first describes the basic processor and briefly covers the conventional method of implementing the instruction set. An introduction into microprogrammed implementation is then covered. The remainder of the chapter is devoted to a discussion of the basic microprogrammed memory and the structure of the microprogrammed word.

2.2 BASIC PROCESSOR

A computer system must be capable of manipulating, storing, and routing data. The component of a computer that operates on the data is the processor. Although the processor is designed to effect complicated changes to the data that it receives, it actually consists of elements making only simple changes. The complex data manipulations are achieved by combining a large number of these simple changes in a variety of ways.

The processor consists of logical elements, each element designed to perform a specific function. For example, some elements store data, some read data from another part of the computer, and others perform simple modifying functions such as complementing the data or combining two operands by either addition or by logical ANDing. These simple basic operations can be combined into functional groups known as *instructions.* An instruction can include a number of operations so that data can be combined, changed, moved, or deleted. The instructions can be further combined into *programs* which use a number of instructions to construct even more complex operations.

The basic logical elements of a processor can perform only a small number of operations at one time. Therefore, to combine a number of these operations into an instruction, the instruction must be divided into either a series of sequential steps or into groups of functions that can be performed simultaneously. One method of describing the procedure the processor uses to execute an instruction is to call each operation (or group of operations) a *machine state.* An instruction then becomes a sequence of machine states which the processor always enters in a specific, predetermined order, depending on the individual instruction.

The processor can be divided into three general functional parts: the INTERFACE section, which exchanges data with devices external to the processor; the DATA PATHS section, which performs data handling functions; and the MICROCONTROL section, which includes the logic that determines which operations are to be performed during a particular state and what the next machine state should be. (These sections of the processor and their component elements are shown on the KD11-A Processor Block Diagram, drawing B-BD-KD11-A-BD.) The INTERFACE section consists basically of logic necessary for transferring data between the processor, the Unibus, and the programmer's console. The DATA PATH and MICROCONTROL sections interact to perform the three main processor functions of data storage, modification, and routing.

In order for the processor to combine data operands, it must be capable of storing data internally while simultaneously reading additional data. The processor often stores information about the instruction being executed, about the program from which the instruction was taken, and about the location of the data being handled, in addition to storing a number of data operands. When the processor must select some of this internally stored data, or store new data, the MICROCONTROL section provides the required control signals to initiate appropriate actions within the data storage section.

Data manipulation is performed both on data that remains within the processor and on data being transferred between the processor and the rest of the system. In some instances, the data remaining within the processor is used to control the processor by providing inputs to the sensing logic in the MICROCONTROL section. The various logic elements that actually modify data are controlled by signals from the MICROCONTROL section which selects the particular operation to be performed.

Interconnections between the logic elements that store data and the logic elements that manipulate data are not fixed; they are established as required by the specific machine state. The MICROCONTROL section generates signals that cause data routing logic elements to form appropriate interconnections within the processor and between the INTERFACE and DATA PATHS sections of the logic.

2.3 CONVENTIONAL IMPLEMENTATION

Before attempting to understand the microprogramming implementation of the MICROCONTROL section, which is the key to the KD11-A Processor, it is advantageous to reveiw the conventional method of implementation which uses combinational logic networks to produce the necessary control outputs.

In a conventional processor, each control signal is the output of a combinational network that detects all of the machine states, as well as other conditions, for which the signal should be asserted. The machine state is represented by the contents of a number of storage elements (such as flip-flops) which are loaded from signals that are, in turn, outputs of combinational networks. The inputs to these networks include: the current machine state, sensed conditions within the processor, and sensed external conditions.

The number of logical elements in a conventional processor is often reduced by using logic networks to generate intermediate signals that can be used to produce a variety of control signals and/or machine states. Unfortunately, while this sharing of logic reduces processor size, it increases the complexity and makes it more difficult to understand the processor logic because it is no longer obvious what conditions cause each signal. In addition, the distinction between sequence control and function control is often lost, making it more difficult to determine whether improper operation is caused by a faulty machine state sequence or by erroneous control signals within an otherwise correct machine state.

A simplified block diagram of a conventional control section of a processor is shown in Figure 2-1. The Instruction Register (IR) and associated decoding logic determine the logic function (instruction) that is to be performed. The major and minor state identification logic serves as a sequence control to determine the order of functions to be performed. The major state logic selects the major operation to be performed, such as Fetch (obtain an instruction), Source (obtain the source operand), Destination (obtain the destination operand), Execute (perform the action specified by the instruction), or Service (handle required interrupts, traps, etc.).

Within each major state, the processor MICROCONTROL section must perform several minor operations. For example, the Fetch major state obtains an instruction from core memory. Minor states during Fetch include: retrieve the instruction from memory, update the Program Counter, load the Instruction Register, and decode the instruction.

Finally, a set of subcommands must be generated to perform the elemental operations required by a minor state. The subcommand set that is selected is dependent on which major and minor states have been selected by the state control logic.

The sequence control of the processor (major state, minor state, and subcommand set logic) is practical only if a well-defined set of elementary operations is generated. This is the function of the state control logic shown in Figure 2-1. The state control consists of a complex array of combinational logic that monitors the output of the IR decoder which defines the instruction, the current machine states (major and minor), and external sources (state of Processor Status Register, console switches, Unibus signals, etc.) to set the required major and minor machine states at the occurrence of each system clock pulse. It should be noted that the state control logic selects the next elementary operation as a function of the current operation and external conditions.

Although the KD11-A Processor does not employ the type of MICROCONTROL section discussed in this paragraph, the concepts presented serve as a review of conventional control and are a desirable starting point from which to discuss the principles of microprogramming. In both cases, the prime function of the MICROCONTROL section is the same; only the hardware implementation differs.

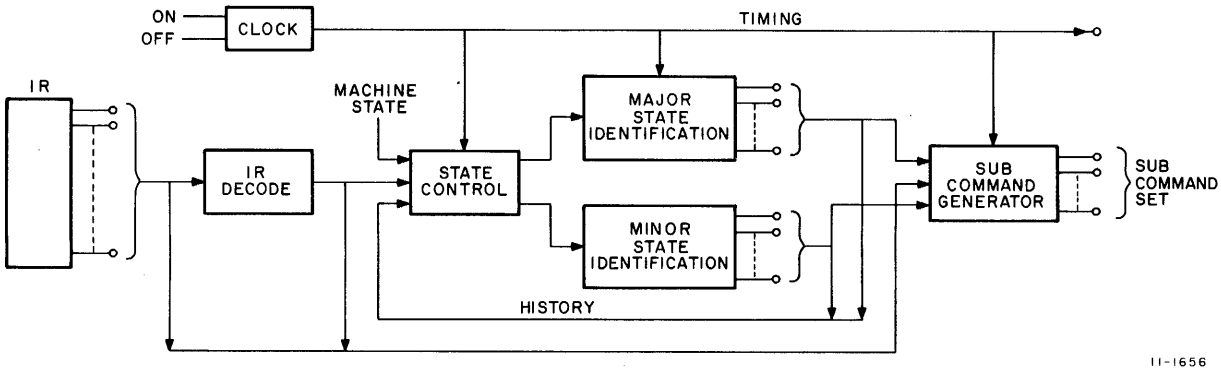


Figure 2-1 Conventional Control Section, Simplified Block Diagram

2.4 MICROPROGRAMMED IMPLEMENTATION

When the control system is implemented by microprogramming techniques, each control signal is completely defined for every machine state. The section of the processor that selects the control signals can thus be implemented as a storage device (read-only memory). This memory is divided into words; there is a separate word for each machine state. Each word, in turn, contains a bit for every control signal associated with the related machine state. During each machine state, the contents of the corresponding word in the read-only memory is transmitted on the control lines. For most control signals, the output of the memory is the control signal and no additional logic is required.

The heart of the microprogrammed processor is the read-only memory (ROM) which stores a copy of the required control signals for each machine state and a list of the machine states to follow the current state. Each word in the ROM defines an elementary operation and the bit pattern within the word corresponds to subcommands. All that is required to generate a unique set of subcommands is to read out the contents of a location in the ROM. To generate a sequence of elementary operations, the address input to the ROM is changed with each system clock pulse. Some of the bits in the ROM are used to define the next location to be read, often depending on conditions sensed by the processor.

Each microprogram word that defines an elementary operation or machine state is referred to as a *microword* (sometimes referred to as a microinstruction). Sequences of microwords are referred to as *microroutines.* The register that defines which microword is to be read is referred to as the *microprogram pointer.*

An instruction fetched from core memory is loaded into the Instruction Register, decoded, and used in generating a microprogram address that points to the starting location of a group of microroutines stored in the ROM. When the microroutines are executed, the required subcommand sets are produced to activate other elements within the processor, such as DATA PATHS or Unibus control in the INTERFACE section.

The microprogram may be viewed as a group of hardware subroutines carefully designed to implement the PDP-11/40 instruction set and permanently stored in the ROM.

In order to maintain proper sequencing of a microroutine, each microword contains an address field for the *next* microword. However, provisions are made to modify this address when it is required to branch to other microwords or microroutines because of conditions sensed within the processor (e.g., instruction, address mode, interrupt flag, etc.).

A simplified block diagram of the microprogrammed control logic is shown in Figure 2-2. As can be seen on the diagram, the instruction loaded into the Instruction Register (IR) from core memory is decoded to provide a ROM address. This address causes a specified control word (microword) to be retrieved from the ROM and loaded into a Buffer Register. This microword contains the control fields used by the processor to perform the selected function. The microword also contains a next address field and a branch test field which are fed back to the ROM Address Generator to select the next microword in the sequence. The microword present in the Buffer Register operates on the machine while the next microword in the sequence is being fetched.

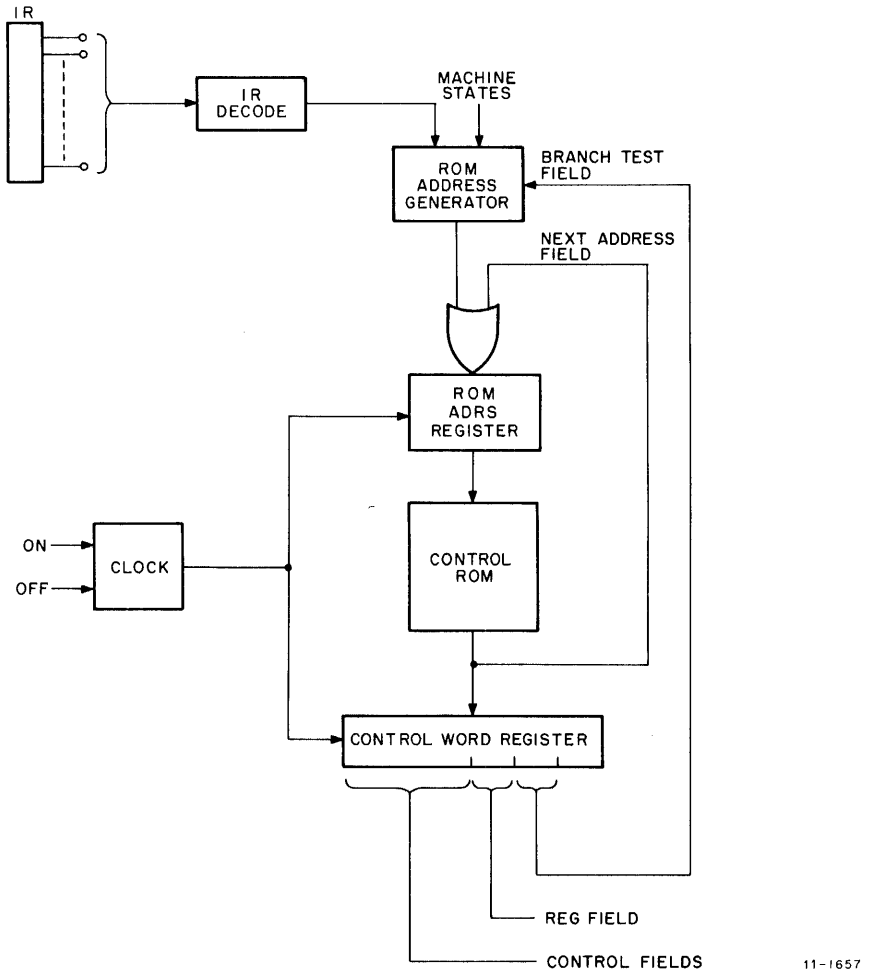


Figure 2-2 Microprogrammed Control Section, Simplified Block Diagram

The combination of the next address field and the branch test field controls the sequence of microroutines. The next address field provides a base address which selects the next microword to be used in the normal sequence. However, this base address can be modified prior to being loaded into the microprogram Address Register.

Before discussing the address modification, it is important to understand that modification occurs prior to storage in the ROM ADRS Register and, therefore, is performed on the subsequent next address (the *next,* next address). For example, microword 1 in the sequence contains an address pointing to microword 2, and microword 2 contains an address pointing to microword 3. When microword 1 is being operated on, the next address field (microword 2) is already present in the ROM ADRS Register and, therefore, cannot be modified. However, when word 1 is being used and the address for word 2 is in the ROM ADRS Register, the address for word 3 can be modified between the ROM output and the ROM ADRS Register.

The branch test field of the microword specifies conditions to be tested and controls when the test is to occur; the conditions determine to what location the microprogram is to branch. Logic within the processor permits testing of the Instruction Register, flags, and other internal and external conditions to determine if branching is required. If a branch is necessary, processor logic modifies the address of the next ROM microword. After the modified address has been loaded into the ROM ADRS Register, a branch occurs to the new location and the specified microword is retrieved from the ROM.

2.5 BASIC READ-ONLY MEMORY (ROM)

The microprogram read-only memory (ROM) contains 256 56-bit words. During each processor cycle, one word is fetched from this ROM and stored in a Buffer Register. The outputs of the Buffer Register are transmitted to other sections of the processor to act as control signals or to be used as the address of the next microword. The first eight bits of every microword (07:00) are used to hold the address of the next microword to be used. The remaining bits (56:09) are control bits. (Bit 08 is reserved for use with extended ROM addresses for Extended Instruction Set options.)

Figure 2-3 illustrates the basic structure of the microwords in the ROM. The detailed format of the microword is shown in print D-BD-KD11-A-BD and in Figure 2-4. Note that this format is identical for all 256 microwords in the ROM. The function of each bit position in the microword is described in Table 2-1.

NOTE

In the KD11-A Processor documentation, the prefix *MICRO* (from the Greek Mu) is abbreviated as U (similar to μ). The U abbreviation appears in the names for the microword buffer (U WORD), in the ROM ADRS (Microprogram Pointer, UPP), and in other logic block names and signal names.

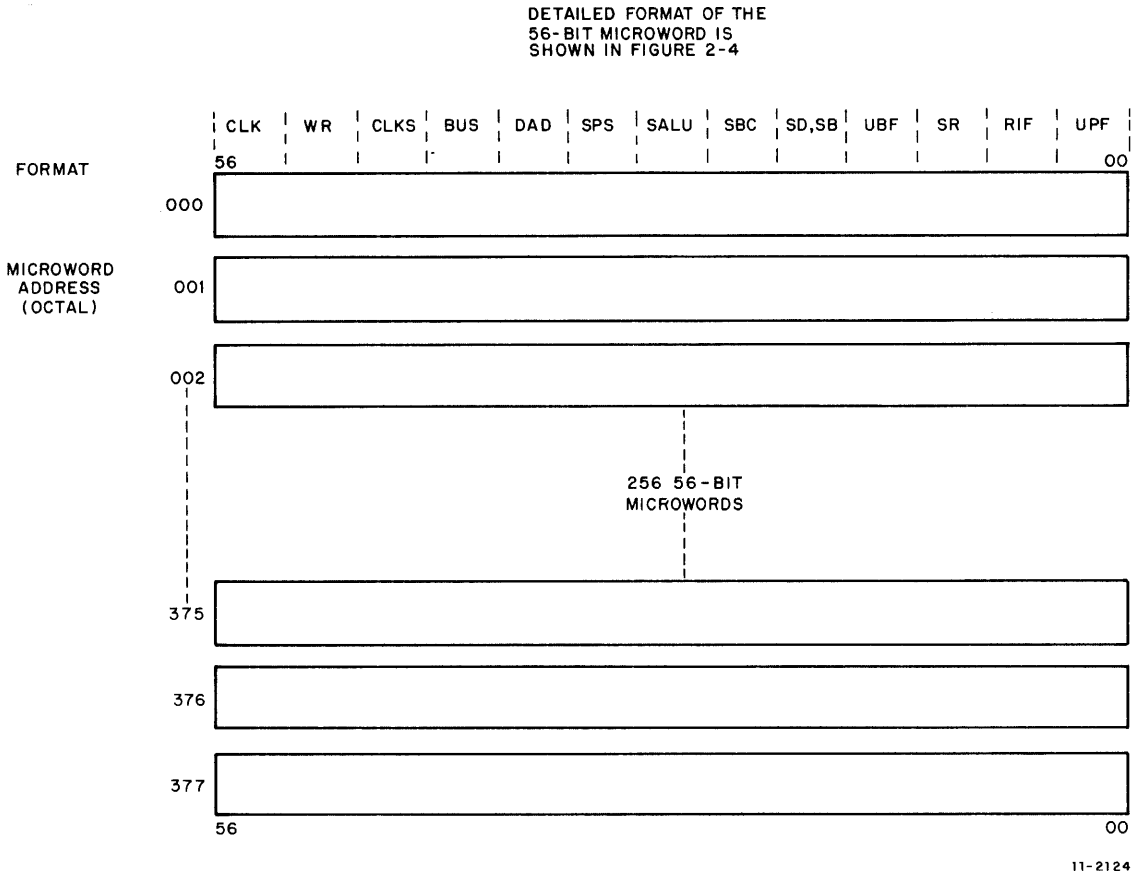


Figure 2-3 Basic ROM Structure

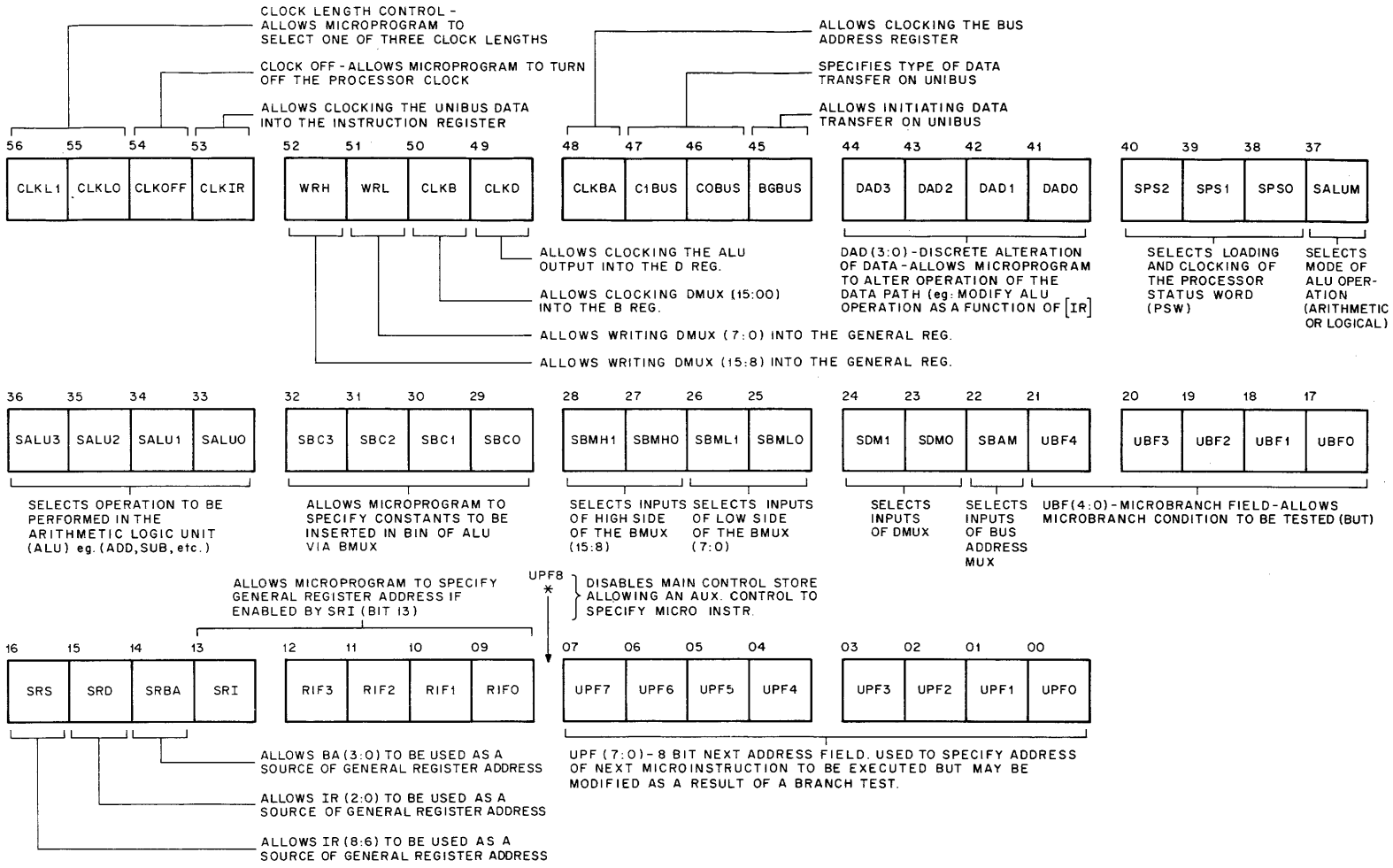


Figure 2-4 Microword Format

Table 2-1
Function of Microword Bits (U WORD)

U Bit	Mnemonic	Meaning and Function
56 55	CLKL1 CLKL0	Clock length control. Permits the microprogram to select one of three clock lengths.
54	CLKOFF	Permits the microprogram to turn off the processor clock.
53	CLKIR	Permits clocking Unibus data into the Instruction Register (IR).
52 51	WRH WRL	Permit writing the Data Multiplexer data into the general registers. WRH writes the high-order byte; WRL writes the low-order byte.
50	CLKB	Permits clocking the entire Data Multiplexer (full word) into the B Register.
49	CLKD	Permits clocking the ALU output into the D Register.
48	CLKBA	Permits clocking the Bus Address Register.
47 46	C1BUS C0BUS	Specify the type of data transfer on the Unibus.
45	BGBUS	Initiates data transfer on the Unibus.
44 43 42 41	DAD3 DAD2 DAD1 DAD0	Discrete alteration of data. Permit the microprogram to alter operation of the data path. For example, modifying the Arithmetic Logic Unit (ALU) operation as a function of the Instruction Register.
40 39 38	SPS2 SPS1 SPS0	Select loading and clocking situations on the Processor Status (PS) word.
37	SALUM	Selects the mode of ALU operation (mode can be either arithmetic or logical).
36 35 34 33	SALU3 SALU2 SALU1 SALU0	Select the operation to be performed by the ALU, such as add, subtract, etc. This selection can be modified by the DAD code noted above.
32 31 30 29	SBC3 SBC2 SBC1 SBC0	Permit the microprogram to specify the constants to be inserted into the B input of the ALU by way of the B Multiplexer.
28 27	SBMH1 SBMH0	Select the inputs of the high-order byte of the B Multiplexer.

Table 2-1 (Cont)
Function of Microword Bits (U WORD)

U Bit	Mnemonic	Meaning and Function
26 25	SBML1 SBML0	Select the inputs of the low-order byte of the B Multiplexer.
24 23	SDM1 SDM0	Select the inputs of the D Multiplexer.
22	SBAM	Selects the inputs of the Bus Address Multiplexer.
21 20 19 18 17	UBF4 UBF3 UBF2 UBF1 UBF0	Represent microbranch field. Select the microbranch condition to be tested. This test is referred to as the Branch Microprogram Test (BUT).
16	SRS	Permits bits (8:6) of the Instruction Register to be used as the source of the general register address.
15	SRD	Permits bits (2:0) of the Instruction Register to be used as the source of the general register address.
14	SRBA	Permits bits (3:0) of the Bus Address Register to be used as the source of the general register address.
13	SRI	Enables RIF (3:0) of the microword for general register address.
12 11 10 09	RIF3 RIF2 RIF1 RIF0	Permit microprogram to specify general register address, provided these bits are enabled by SRI (bit 13).
07 06 05 04 03 02 01 00	UPF7 UPF6 UPF5 UPF4 UPF3 UPF2 UPF1 UPF0	Represent an 8-bit next address field that is used to specify the address of the next microinstruction to be executed. However, it may be modified as the result of a BUT. The U08 bit is for UPF8 and is provided for the KE11-E option.

CHAPTER 3
BLOCK DIAGRAM DESCRIPTION

3.1 SCOPE

This chapter introduces the KD11-A Processor architecture by describing the basic block diagram which illustrates all of the major logic elements and interconnections within the processor.

The block diagram (drawing D-BD-KD11-A-0) has been divided into three major functional groupings: INTERFACE, DATA PATHS, and MICROPROGRAM CONTROL. All of the components in each of these segments are covered in detail in succeeding paragraphs. Paragraph 3.5 contains a tabular listing of all components on the block diagram and includes a brief physical description as well as related inputs and outputs. This abbreviated summary can be used as a quick reference once the more detailed description of the block diagram is understood, or it can be used for a quick overview of the KD11-A Processor by those who are already familiar with PDP-11 processors and microprogramming techniques.

The block diagram format provides blocks with major functions titled, and with major data and control interconnections, as well as specific clock and microcontrol signals indicated. In the lower right hand corner of each block is a K- reference which indicates the module schematic or schematics on which the contents of the block can be found. For example, K1-7 indicates sheet 7 of the K1, or M7231 module print; K2 indicates several sheets on the K2, or M7232 module print. The details of logic operation are contained in Chapter 5.

3.2 INTERFACE

The first section of the processor shown on the block diagram is the INTERFACE section which is used to interconnect the KD11-A Processor with other components of the PDP-11/40 system. Each of the functional blocks shown on the INTERFACE portion of the block diagram is covered in the following discussion.

3.2.1 KY11-D Programmer's Console

The KY11-D Programmer's Console is an integral part of the PDP-11/40 system and provides the programmer with a direct system interface. The console allows the user to start, stop, load, modify, or continue a program. Console displays indicate data and address flow for monitoring processor operations. The console control logic is part of the processor INTERFACE section, while the Switch Register, the DATA display, and the ADDRESS display are part of the KY11-D console.

The Switch Register is located on the KY11-D console and consists of the manually-operated switches with resistor pull-ups gated through 8881 drivers to the Unibus. During console operation, the Switch Register is addressed on the Unibus via a microroutine. When the Switch Register address is decoded, the driver gates are enabled, allowing the contents of the Switch Register to appear on the Unibus.

The DATA display indicates the output of the processor Data Multiplexer (DMUX) which selects information from a variety of sources within the processor (e.g., the Unibus, the ALU output, and BUS RD). The display consists of light-emitting diodes (LEDS) and associated current limiting resistors mounted in the programmer's console. The indicators are connected to the processor by cables. The output line of the Data Multiplexer [D MUX (15:00)] always controls the display, but since the multiplexer can select multiple inputs onto its output line, displayed information is varied.

The ADDRESS display indicates the contents of the processor Bus Address Register (BA Register). This display also consists of LEDS and current limiting resistors mounted on the console and connected to the processor by cables. Note that there is no multiplexing involved with the ADDRESS display as there is with the DATA display. Although it is possible to load specific data into the Bus Address Register for display, the display usually indicates the last used Unibus addresses.

Console control logic is associated with the programmer's console operational switches that provide such manual functions as start, halt, load address, examine, deposit, and continue. The console contains the manual switches and associated set-reset flip-flops used for preliminary contact bounce filtering. However, primary console control is handled by the processor either by means of the microprogram, or by combinational logic and flag flip-flops. The microprogram senses switch activation and branches to the specific routine required. The flags accommodate the special needs of the START and CONT switch sequences as well as the incrementation requirements of consecutive EXAM or DEP sequences.

The remaining functional components of the INTERFACE portion of the processor are the Unibus timing and control, the bus terminator and connector module, and the Unibus drivers and receivers.

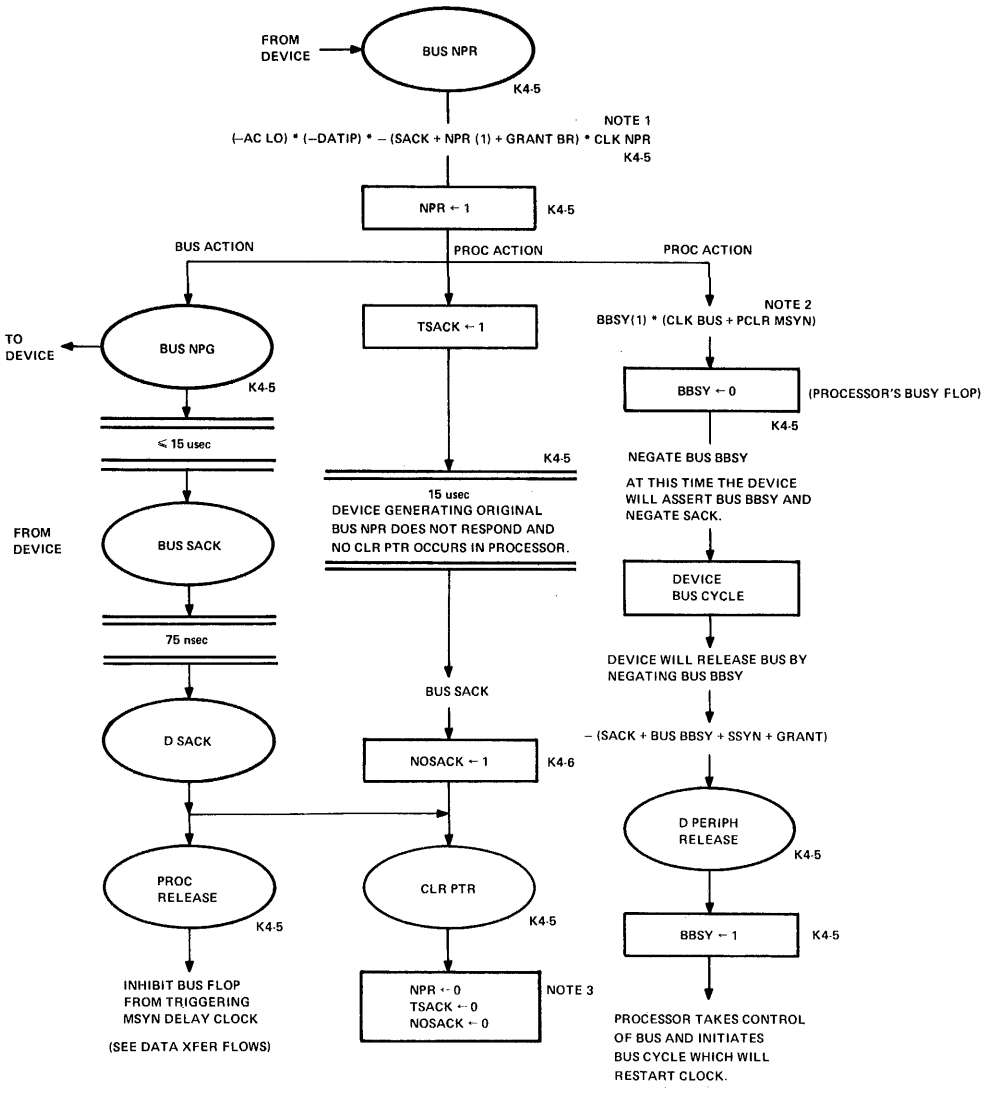
3.2.2 Unibus Timing and Control

The Unibus timing and control logic provides the required processor control of the Unibus, controls data transfer functions, bus ownership functions, and other miscellaneous functions. The control logic includes drivers and receivers for Unibus signal lines as well as timing and priority logic. Combinational logic, pulse circuits, and discrete flip-flops provide control for data transfers (DATI, DATIP, DATO, DATOB) between the processor and the Unibus with associated error checking (odd address, memory parity, stack overflow) and correction (data timeout).

In addition to the data transfer function, the Unibus timing and control logic provides the necessary control for bus ownership, transfer of bus ownership for non-processor requests (NPRs) and bus requests (BRs), and the timeout function for non-response conditions. The logic also provides power-fail timing related to BUS AC LO, BUS DC LO, and BUS INIT signals. Combinational logic, which includes a number of one-shot timing circuits, sequences these signals for power on and power off conditions.

Bus ownership is arbitrated by the processor on a priority basis. The highest priority is the NPR, followed by BR7, BR6, BR5, and BR4. In order for a device to gain bus ownership, its priority must be greater than that of the then current bus master. A flowchart showing the process of arbitrating bus ownership for NPRs is shown in Figure 3-1 and for BRs in Figure 3-2. The timing sequence on the Unibus for NPR transfers is shown in Figure 3-3, and the timing sequence for BRs is shown in Figure 3-4.

A device requests bus ownership by asserting its associated request line, either NPR or BRn. The request is acknowledged when the processor asserts a corresponding bus grant line, NPG or BGn. The bus grant signal causes the requesting device to then assert SACK, which is sent back to the processor. The processor has been asserting the BBSY signal up to this time, and when it receives SACK in response to BGn, it drops BBSY. On an NPR, BBSY is dropped on the next CLK BUS or P CLR MSYN after the NPR flag is set. However, the requesting device reasserts BBSY as soon as the processor drops it, and the requesting device is now the bus master. Several devices may share a common BR priority and request bus ownership simultaneously. In that event, the device closest to the processor has the higher priority since each BG and NPG line is serially wired through every device on the Unibus. When a device asserts a BR or NPR line, it also opens the BG or NPG line to the device next to it, on the side opposite from the processor.



11-1680

Notes:

1. NPRs are clocked frequently to determine if NPR Service needed. NPG occurs.
   a. BGBUS (1) – microword specifies a data transfer
   b. ENPR CLK – from EIS/FIS option
   c. CLK IR – microprogram in FETCH
   d. BUT26*P3 – microprogram just entered SERVICE
   e. –AWBY*SET CLK – clock restarting after bus cycle
   f. P MSYN – A device has asserted BUS MSYN.

2. Microprogram either just starting or finishing a data XFER Bus Cycle or in Service when processor gives up the bus to the NPR device.
   a. CLK BUS – just starting processor Data XFER – clock will be turned off in this U word or succeeding U word.
   b. PCLR MSYN – just finishing a Data XFER Bus Cycle – clock is on but will go off when micro-program reaches next Data XFER microroutine.
   c. SERVICE – give up the bus to do NPRs on BUT26 and on AWBBY.

3. SACK timeouts will not cause a trap but will generate CLR PTR which cleans up the NPR Priority XFER control flip-flops, and will restart Processor Clock.

Figure 3-1 KD11-A Priority Transfer Timing and Control for NPRs

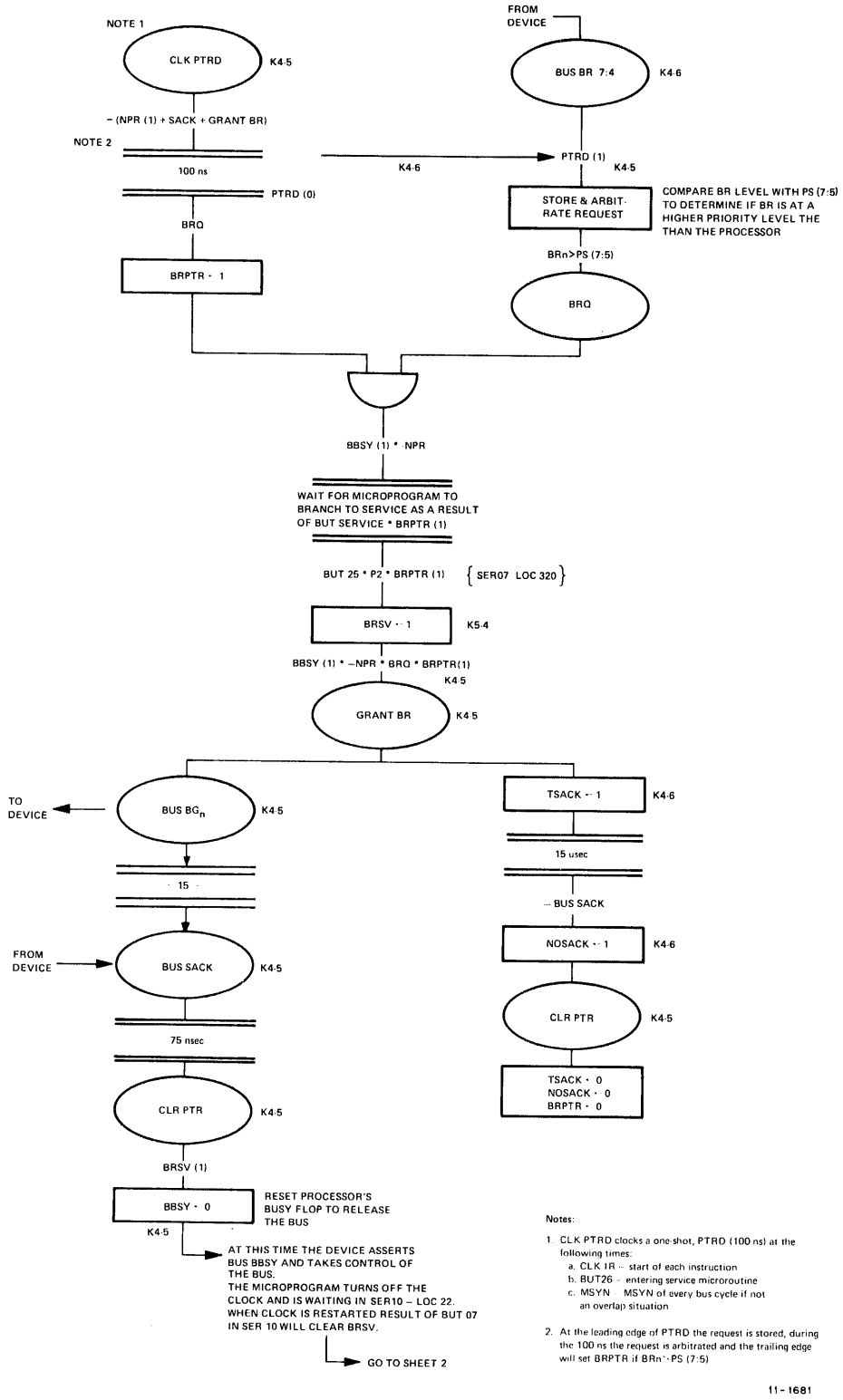


Figure 3-2 KD11-A Priority Transfer Timing and Control for BRs (Sheet 1 of 2)

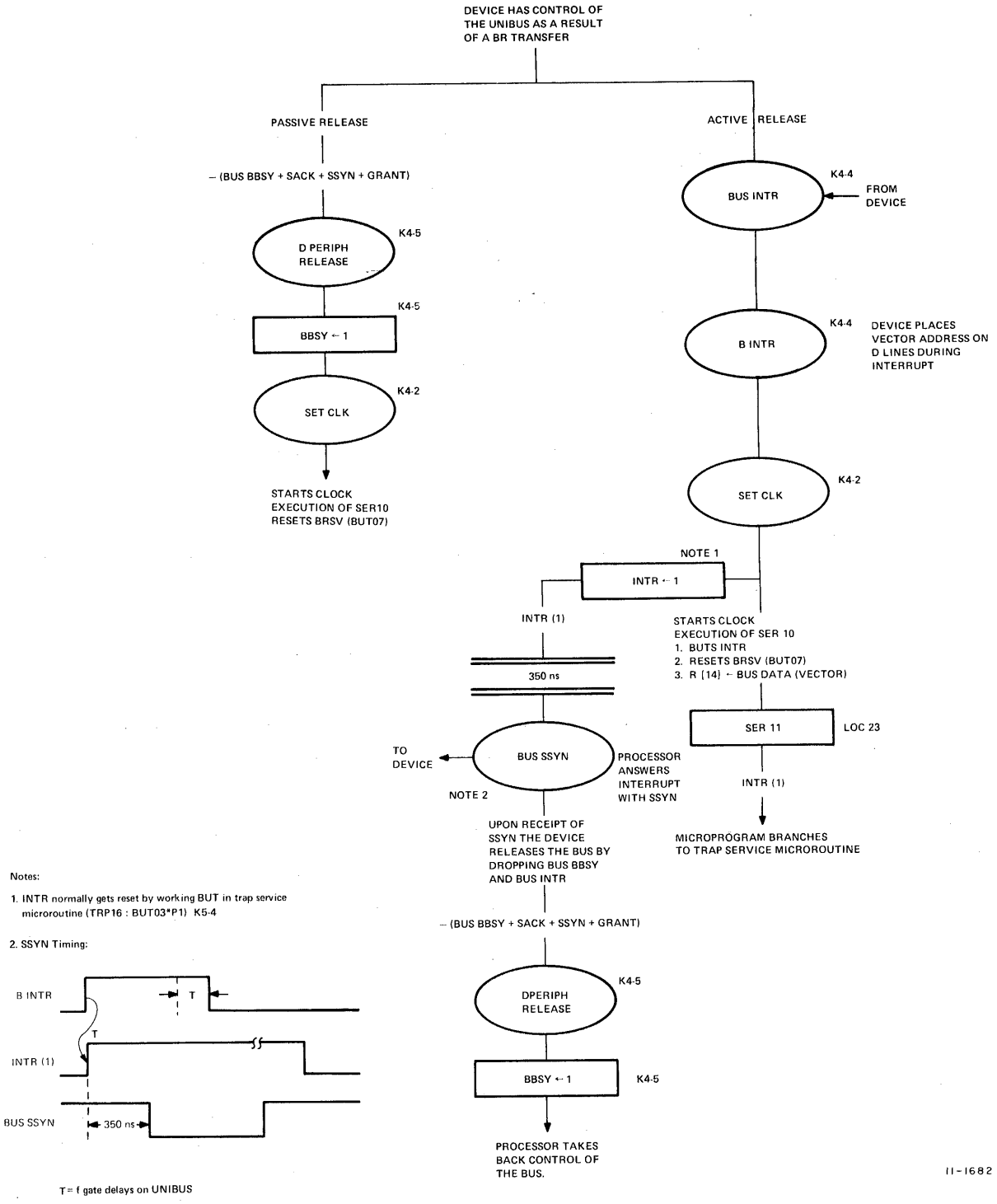


Figure 3-2 KD11-A Priority Transfer Timing and Control for BRs (Sheet 2 of 2)

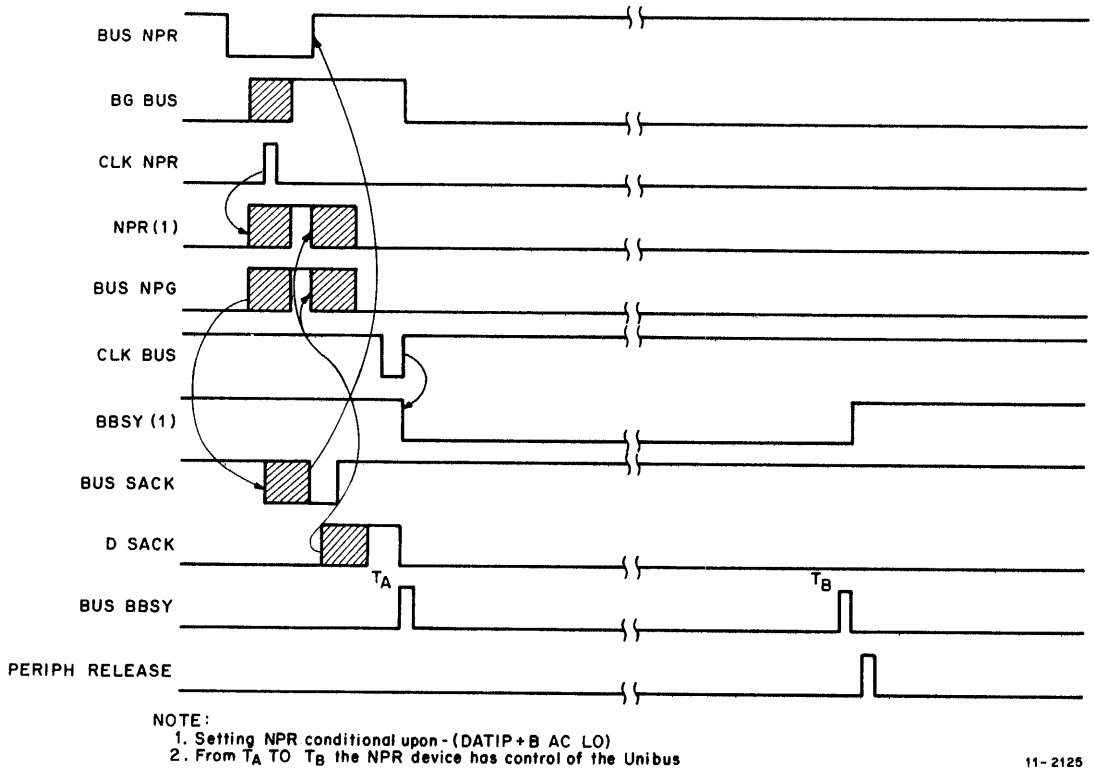


Figure 3-3 NPR Priority Transfer Timing Sequence

After a device has become bus master, it proceeds to transfer data on the Unibus. The sequence is shown on the flowchart in Figure 3-5. The timing of data transactions are shown in Figures 3-6 and 3-7, for DATI and DATO, respectively.

The power up sequence is shown on the timing diagram in Figure 3-8. BUS AC LO and BUS DC LO are generated by the power supply and indicate the status of the input ac power and the derived dc power. When BUS DC LO goes high, it triggers a 20 ms one-shot, PWRUP INIT. The PWRUP INIT signal initializes all the processor logic and all the Unibus devices. It also causes the microaddress 377 to be generated and set into the UPP Register in order to load it with 0s. The trailing edge of PWRUP INIT triggers the POWER RESTART signal if AC LO is not asserted. The trailing edge of POWER RESTART causes a starting microaddress to be generated and set into the UPP Register, starts the processor clock, and inhibits the initiation of a power down sequence for 3 ms, should BUS AC LO be asserted.

The timing sequence of the power down sequence is shown in Figure 3-9. Powering down causes the power supply to assert BUS AC LO followed by BUS DC LO. The BUS AC LO signal sets the LOWAC flip-flop, assuming the 3 ms PWRDN DELAY signal is not present. With LOWAC (1) set, the next P1 or P3 clock pulse causes CLK PWRDN to be generated, setting the PWRDN flag. The LOWAC flip-flop is reset on the next P1 or P3 clock pulse when PWRDN (1) is asserted. At the completion of the current instruction, the PWRDN flag causes the microprogram to branch to SERVICE and subsequently to the TRAP SERVICE microroutine. The power fail trap routine can call a subroutine from memory that causes all volatile information to be saved and the program to be halted at a known location for restarting. Meanwhile, the BUS AC LO signal causes a 15 ms AC LO signal to be generated, locking up the original AC low condition which may or may not still be present. The BUS AC LO signal inhibits further NPR transactions. Also, a 7 ms DELAY DC LO signal was triggered at the same time. The trailing edge of this signal pulses BUS DC LO, which in turn generates PWRUP INIT, if BUS DC LO is not being asserted by the power supply. At the trailing edge of PWRUP INIT, it is possible to enter the power up sequence if BUS AC LO indicates normal power again on the bus. (Detailed examples of the power down and power up sequences are given in Chapter 5 in the discussion of the logic on Sheet K5-8.)

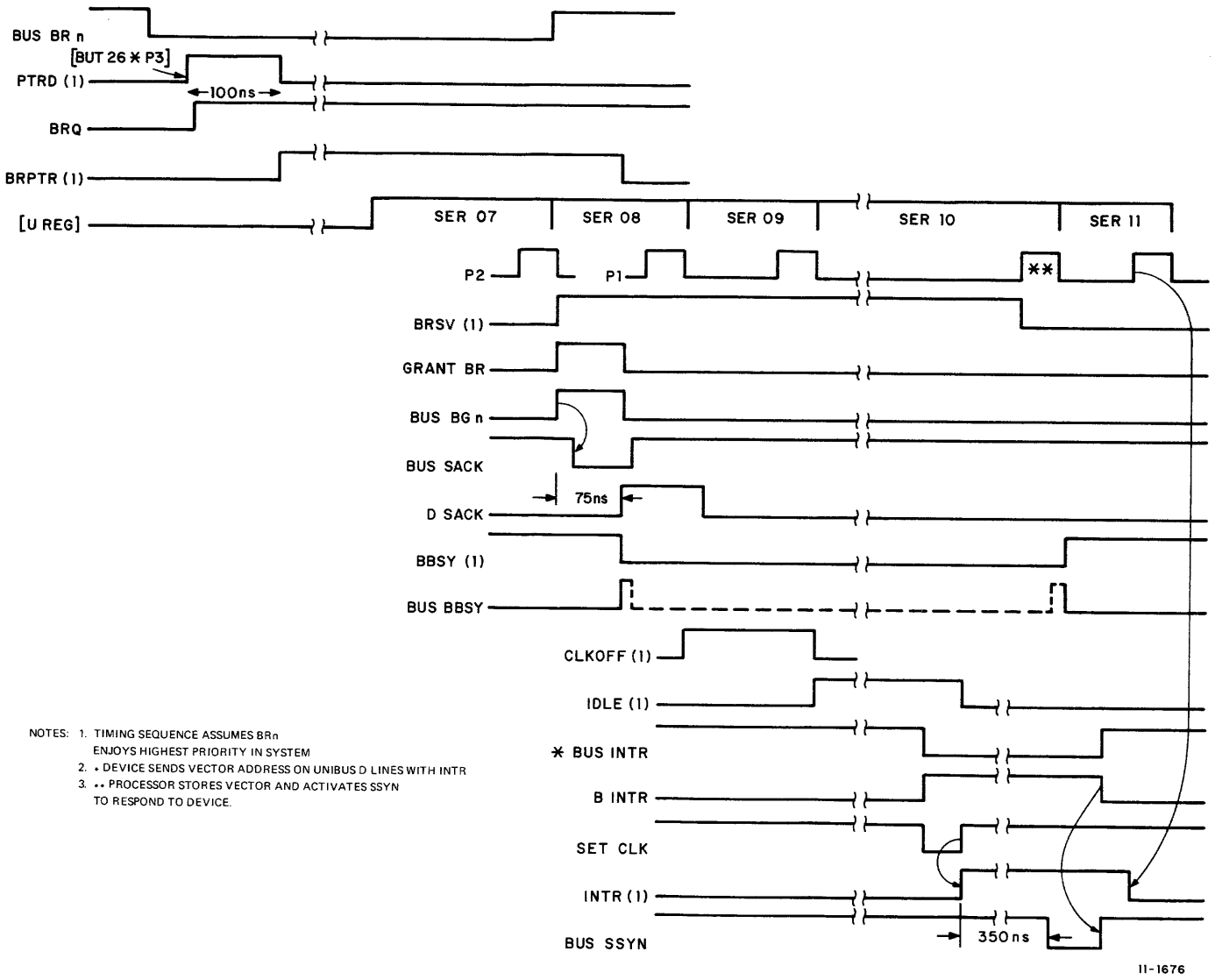
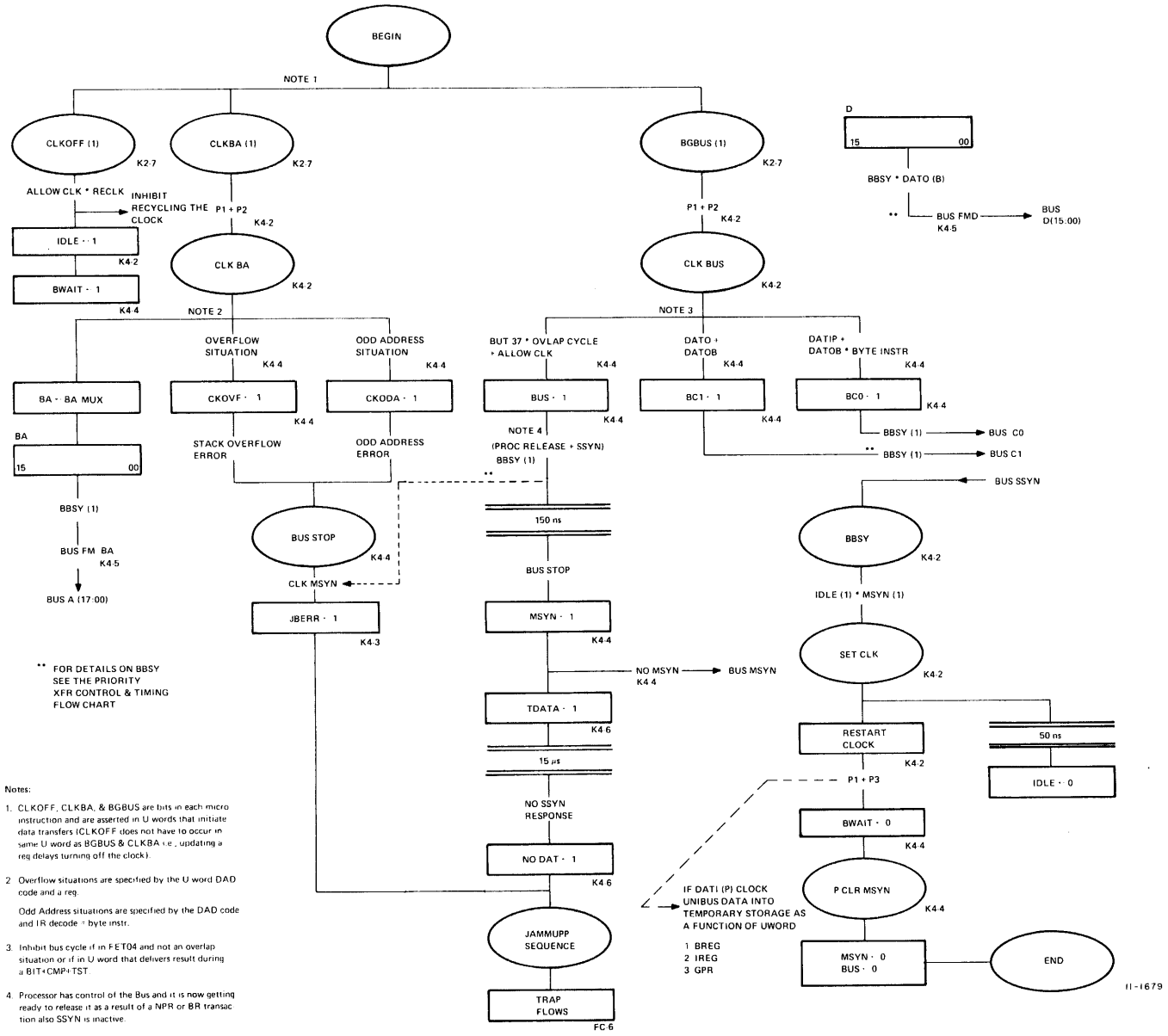


Figure 3-4 BR Priority Transfer Timing Sequence

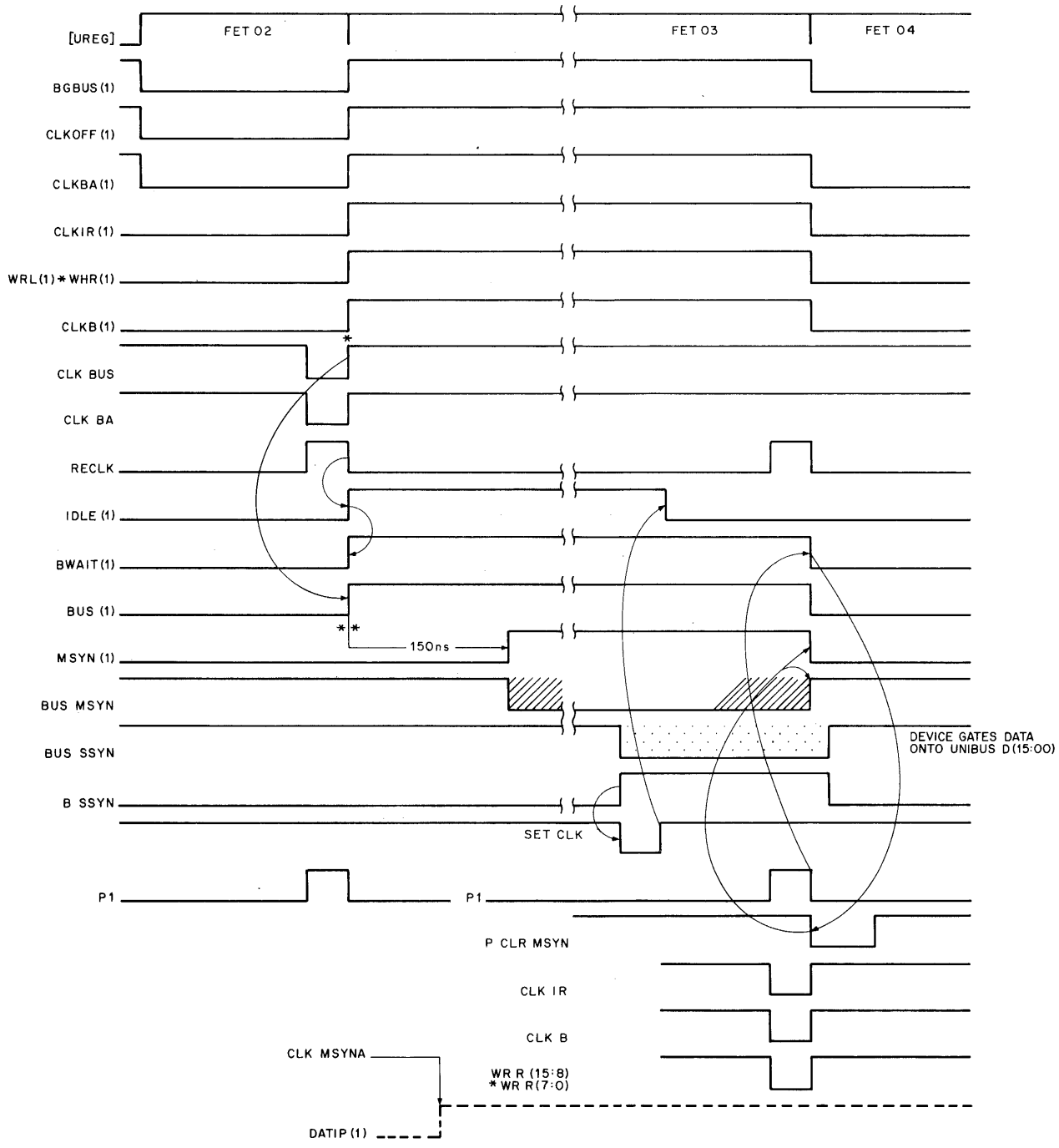


Notes:

1. CLKOFF, CLKBA, & BGBUS are bits in each micro instruction and are asserted in U words that initiate data transfers (CLKOFF does not have to occur in same U word as BGBUS & CLKBA i.e., updating a reg delays turning off the clock).
2. Overflow situations are specified by the U word DAD code and a reg.

   Odd Address situations are specified by the DAD code and IR decode = byte instr.
3. Inhibit bus cycle if in FETO4 and not an overlap situation or if in U word that delivers result during a BIT+CMP+TST.
4. Processor has control of the Bus and it is now getting ready to release it as a result of a NPR or BR transaction also SSYN is inactive.

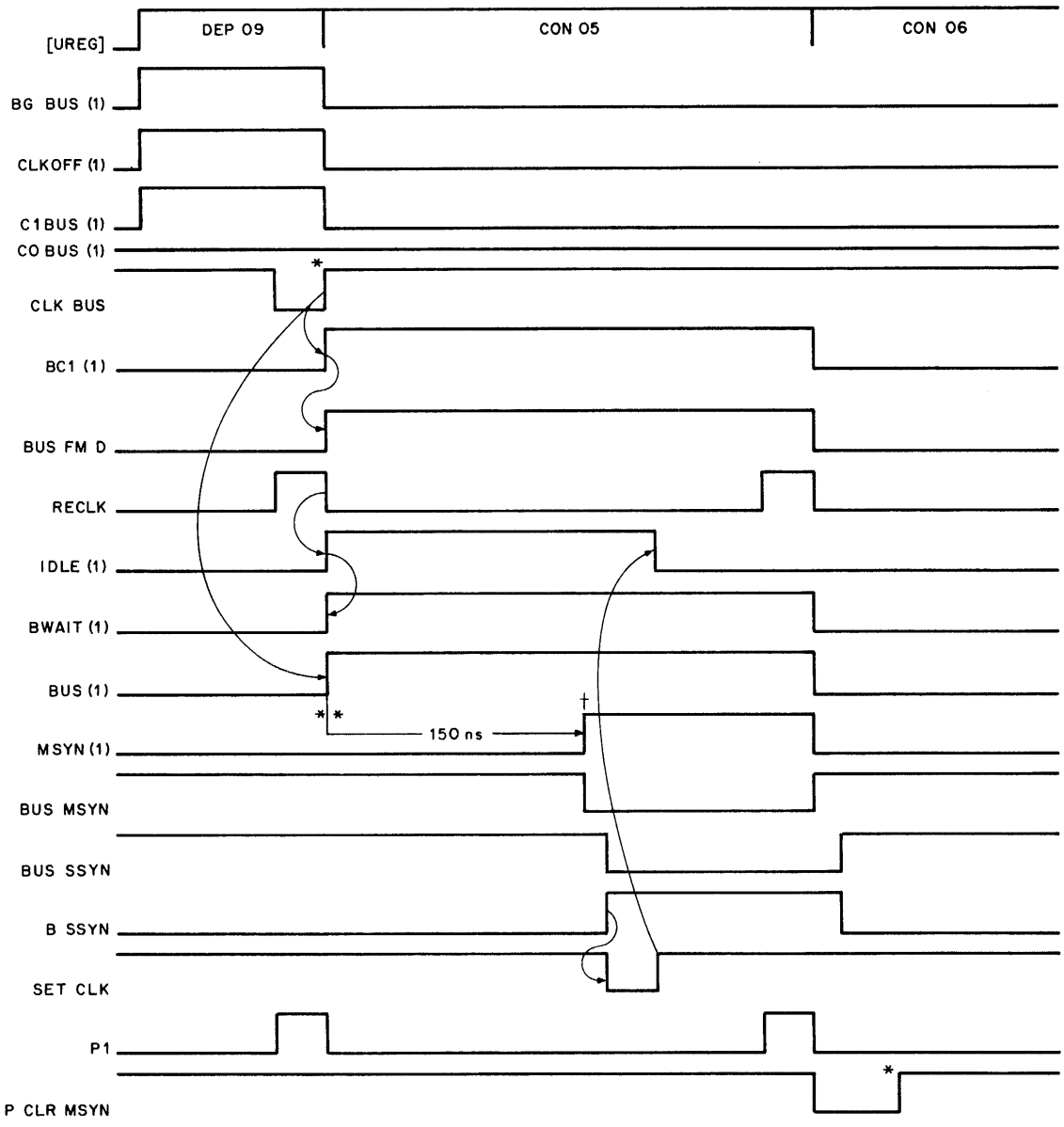
Figure 3-5 KD11-A Bus Data XFER Timing and Control



NOTES:
1. EFFECTS OF GATE DELAYS AND F.F. RESPONSE NOT SHOWN
2. ASSUMES KD11-A HAS CONTROL OF THE UNIBUS [BBSY (1)]
3. * CLOCK NPR REQUESTS: IF NPR (1) THEN BBSY ← 0
4. ** TRIGGERING DELAY TO CLOCK MSYN CONDITIONAL UPON MACHINE STATE: ← (PROC RELEASE + SSYN) * BBSY (1)
5. MSYN BEING SET TRIGGERS 15 USEC BUS TIMEOUT DELAY
6. INHIBIT SETTING MSYN IF BUS STOP ACTIVE: [RED ZONE OVFL + ODA ERR] TRAP
7. SCALE: 1 INCH ≈ 100 NSEC.

11-1677

Figure 3-6 KD11-A DATI(P) Bus Transaction Timing Diagram



NOTES:
1. EFFECTS OF GATE DELAYS AND F.F. RESPONSE NOT SHOWN
2. ASSUMES KD11-A HAS CONTROL OF THE UNIBUS [BBSY(1)]
3. • CLOCK NPR REQUESTS. IF NPR (1) THEN BBSY ← 0
4. •• TRIGGERING DELAY TO CLOCK MSYN CONDITIONAL UPON MACHINE STATE: ← (PROC RELEASE + SSYN) • BBSY (1)
5. † MSYN BEING SET TRIGGERS 15 USEC BUS TIMEOUT DELAY
6. INHIBIT SETTING MSYN IF BUS STOP IS ACTIVE: (RED ZONE OVFL + ODA ERR) TRAP
7. SCALE: 1 INCH ≈ 100 NSEC.

11-1678

Figure 3-7 KD11-A DATO(B) Bus Transaction Timing Diagram

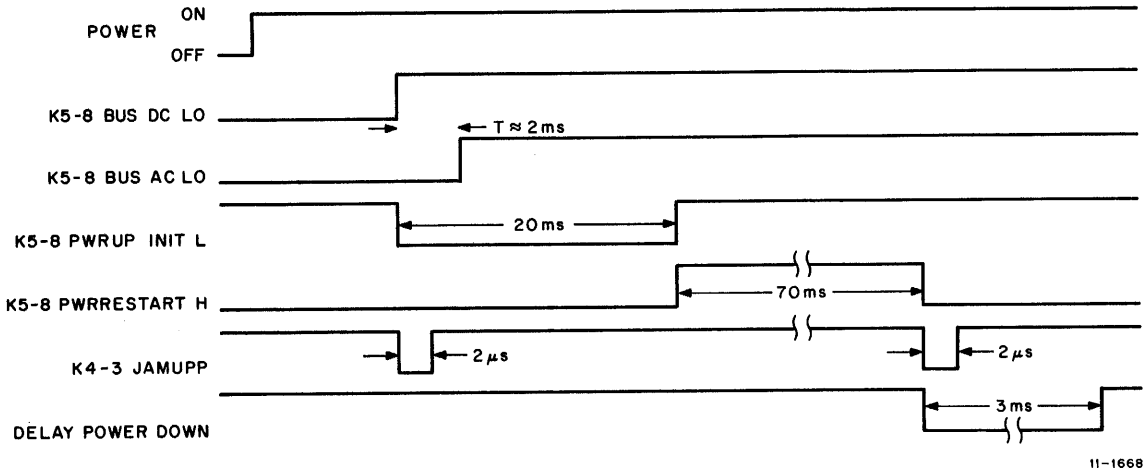


Figure 3-8 KD11-A Power Up Timing Sequence

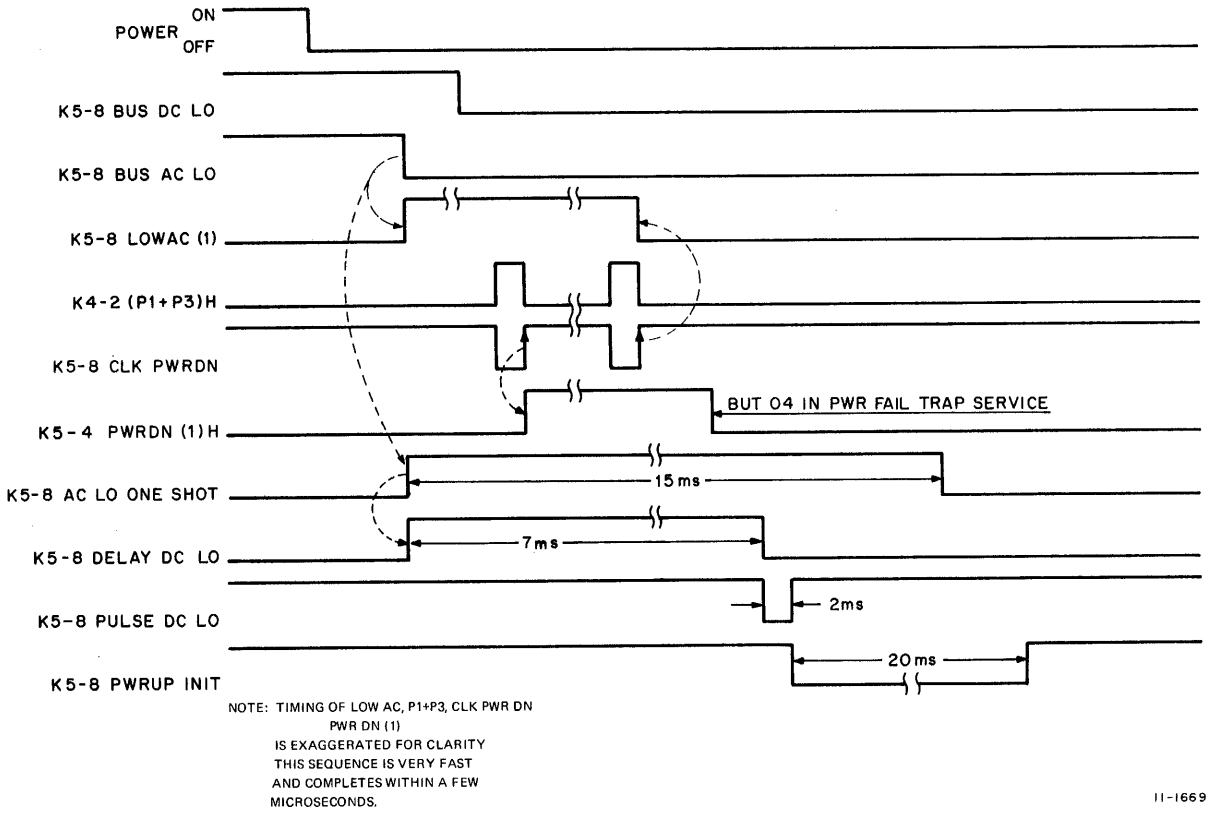


Figure 3-9 KD11-A Power Down Timing Sequence

The microprogram interfaces directly with the Unibus timing and control logic. The start or error checking flip-flops are loaded, either directly or conditionally from the microword. The loading of Unibus data and the deactivation of MSYN occur as a function of the next microword after a DATI or DATIP transfer operation. The processor transmits address and data information to the Unibus under control of the microprogram. Note, however, that bus ownership, the power fail logic, and the Unibus data transfer operate asynchronously and are independent of the microprogram.

The interface portion of the processor contains both bus transmitters (8881 gates) and bus receivers (380 gates) to provide the necessary conversion so that processor and Unibus signals are compatible. The transmitters (drivers) permit the processor to place groups of signals on the bus; the individual signals are noted on the block diagram on the output line of the associated gate. These signals include the outputs of the Bus Address (BA) Register, the D Register, the Processor Status (PS) Register, and the Switch Register. Inputs to the processor from the bus are gated through the bus receiver to the D Multiplexer, which then routes the signals to the proper component within the data paths.

The final functional component in the INTERFACE section of the processor is the bus terminator and connector module, which interconnects system units and also provides the termination required by the Unibus. In the KD11-A Processor, a single set of slots (A09, B09) is provided for the Unibus interface and the processor is single ended. Note that the Unibus Terminator module (M930) is located in, and powered from, the last device on the bus.

3.3 DATA PATHS

The DATA PATHS portion of the KD11-A Processor manipulates, stores, and routes data within the processor.

The prime element of the data path logic is the ALU which operates, both logically and arithmetically, upon input data from the interface portion of the processor. To a certain extent, data path logic is ordered upon the ALU because of the requirements to provide data to each of its inputs and to store, or otherwise use, its output. The ALU and all other components in the processor data paths are described in the following paragraphs.

For the purposes of the following discussion, the term "Scratch Pad Register" refers to one of the 16 internal processor registers shown on the block labeled REGISTER (REG). These registers are also referred to as the General Registers.

3.3.1 Data Paths, Multiplexers and Registers

The Scratch Pad Register supplies operands for the ALU. These operands either come directly from an instruction source or destination mode operation, or they are stored in the Scratch Pad Register during address calculations. In either case, the ALU receives a direct input from the BUS RD (15:00) line. This input is referred to as the *A input.*

In the Scratch Pad Registers, only one address may be accessed at a time, and simultaneous read and write operations are not permissible. In order to provide the two ALU operands (when both operands come from the Scratch Pad Register), it is necessary to provide temporary storage. This storage is provided by the B Register; the contents of the B Register can be fed through the B Multiplexer (B MUX) into the B input of the ALU.

The ALU A input provides variable operands; the B input provides variable operands, constants, and swapped and sign-extended byte operands. The A input usually comes from the Scratch Pad Register on the wire-ORed BUS RD lines (as shown by the dotted OR gates on the block diagram). The Processor Status Register of the basic machine and certain multiplexers with the KE11-E and KE11-F options are also connected to the A input.

The B input comes from the B MUX which receives its input from either the B constants or the B Register. The B Register, in turn, receives its input from the D Multiplexer (D MUX) which has four possible inputs. Therefore, the B input to the ALU comes from a variety of sources with two levels of multiplexing. These various inputs are discussed in the following paragraphs.

The four inputs to the D MUX are: Unibus data lines BUS D (15:00), which permit the ALU to receive operands from other devices within the system; the buffered BUS RD (15:00) lines, which permit operands from the Scratch Pad Register; the output of the D Register, which is the output of the ALU and can permit the result of a previous arithmetic operation to be used as an operand; and the shifted output of the D Register.

The desired D MUX output can be stored in the B Register, which, in turn, can be fed to the ALU by means of the B MUX. Note that the buffered BUS RD signal can be fed through the D MUX into the B Register. This data path is of special interest in the machine instruction for the register-to-register operations, where both the A and B inputs of the ALU must come from the Scratch Pad Register. For example, the first operand passes through the D MUX into the B Register for storage. The second operand can then be fed to the A input of the ALU, and the first operand fed to the B input by means of the B MUX.

The B constants, which are applied through the B MUX to the ALU, provide elementary values (such as 1_8 and 2_8) for incrementation or decrementation throughout machine operation. They also provide other values such as the Switch Register address, more complex constants such as trap vectors or masks for manipulating instruction offsets, and the conditional constants which are a function of machine status and jumper selection.

The B input to the ALU can be either the B constants value or one of the four possible functions of the B Register. The four B Register functions are:

a. B Register – The contents of the register are applied directly to the ALU. Therefore, BIN (15:00) of the ALU equals B (15:08) and B (07:00).

b. B Extend – The B Register contents are gated so that bit 07 (MSB of the low-order byte) provides an extension for the high-order byte. Note that in this case, the value in the high-order byte is either all 1s or all 0s depending upon bit 07 of the B Register; the low-order byte is the B Register directly.

c. Byte Duplication – Either the low-order byte or the high-order byte may be duplicated. Therefore, BIN (15:00) of the ALU equals either B (15:08) and B (15:08), or B (07:00) and B (07:00).

d. Byte Swapping – The high-order and low-order bytes may be exchanged. Therefore, BIN (15:08) of the ALU equals B (07:00) and BIN (07:00) equals B (15:08) for the high byte and the low byte, respectively.

The ALU provides an altered data output that is used for Unibus addresses and data, and by internal processor registers such as the Scratch Pad Register and the Processor Status Register. The output of the ALU is stored in the D Register and/or the Bus Address Register.

The D Register storage capability permits data which has been operated upon in the ALU to be fed around to the B MUX for further manipulation, thus permitting data to be stored in another register (the B Register). This additional path and storage capability is important because it is necessary for single or double operand register operations and is very often necessary in iterative operations.

Operation of the ALU is also determined by the carry-in (CIN logic) and carry-out (COUT MUX) signals. The carry-in signal does not come directly from the microprogrammed word but is a function of the microprogrammed word and the conditions (usually the Instruction Register) that are enabled at specific locations in the microprogrammed flow.

The Carry-out multiplexer (COUT MUX) provides multiplexing of the specific carry information normally used in the PDP-11. The signals that can be selected are: COUT 15, COUT 07, ALU 15, and PS(C). The COUT 15 signal represents the carry from a word operation; the COUT 07 signal represents the carry from a byte operation. These

signals are used for condition code inputs and rotate/shift operations. The ALU 15 signal is the bit 15 output of the ALU which is used for rotate/shift operations. The PS(C) signal is the carry bit from the Processor Status Register. The signal selected by the COUT MUX is clocked into an extension of the D Register which is called D(C). This storage extension is used in rotate/shift operations and in certain arithmetic operations.

A summary of the functions of the BA MUX, the D MUX and the B MUX, and their associated control signals is given in Table 3-1.

Table 3-1
PDP-11/40 Data Path Multiplexer Control

Signal	Function
BA MUX	
Select BA Mux, bits (15:00)	
SBAM(0)	Select BUS RD data
SBAM(1)	Select ALU output
D MUX	
Select D Mux, bits (15:00)	
SDM1(0) * SDM0(0)	Select BUS RD data
SDM1(0) * SDM0(1)	Select Unibus data
SDM1(1) * SDM0(0)	Select D Register
SDM1(1) * SDM0(1)	Select D Register shifted right with D(C) shifted into bit position 15.
B MUX	
Select B Mux Low, bits (07:00)	
SBML1(0) * SBML0(0)	Select B (07:00)
SBML1(0) * SBML0(1)	Select B (07:00)
SBML1(1) * SBML0(0)	Select B (15:8)
SBML1(1) * SBML0(1)	Select BC (07:00)
Select B Mux High, bits (15:08)	
SBMH1(0) * SBMH0(0)	Select B (15:8)
SBMH1(0) * SBMH0(1)	Select B (07)
SBMH1(1) * SBMH0(0)	Select B (07:00)
SBMH1(1) * SBMH0(1)	Select BC (15:8)
(a + b) * e	Selects B Register
(a + b) * f	Selects B Register with sign extension in high byte [B(15:8) ← B(7)]
c * g	Swaps bytes
c * e	Duplicates high byte in low byte
d * h	Selects constant
a + b * g	Duplicate low byte in high byte

3.3.2 Decoding

The address and data decoding logic is a combinational logic network that decodes the output of both the D and BA Registers. When the D Register output is decoded, the decoder senses whether or not the output (for both byte and word segments) is 0 [D (15:00) = 0 H]. The Bus Address Register is decoded to determine if a processor address has occurred, or if an address is less than specified values. It should be noted that the decoding logic decodes the BA Register and not the Unibus address. In the first case, the processor addresses, which represent only those internal registers that can be accessed by the processor itself, are used to gate Unibus responses for bus operations. If the decoded address of the Processor Status Register or the console Switch Register, then either PS ADRS H or SR ADRS H is true. Other addresses also exist. If the decoded address is less than the specified value, then a stack overflow violation may occur and the BOVFL signal is true. Stack limit errors are either yellow zone (warning) or red zone (fatal) indications.

3.3.3 Arithmetic Logic Unit

The Arithmetic Logic Unit (ALU) is the heart of the data path logic. It performs 16 Boolean operations and 16 arithmetic operations on two 16-bit words. The ALU is controlled by six input signals. One signal, ALUM H, selects either the logic or arithmetic mode of operation. Four signals (ALUS0 through ALUS3) select the desired function. The sixth signal is the output of the carry-in (CIN) logic. Basically, the ALU receives two 16-bit words as inputs (AIN and BIN) and performs the operation selected by the six control signals. The output of the ALU is, therefore, altered data which is used for Unibus addresses and data, and is also used by the internal processor registers such as the Scratch Pad Register or the Processor Status Register. The output of the ALU is stored in the D Register or the BA Register for use.

3.3.4 PS Register

The Processor Status (PS) Register is an 8-bit register that stores information on the current priority of the processor [bits (07:05)], the result of the previous operation (condition codes bits N, Z, V, C), and an indicator for detecting the execution of an instruction to be trapped during program debugging (T bit). The PS Register is located between two basic data paths: D MUX (15:00) and BUS RD (15:00). The register is loaded from the D MUX. In addition, the condition codes control logic provides inputs to the N, Z, V, and C bits. The register output is either gated directly onto the Unibus (in cases where the processor has addressed the Unibus as an absolute address) or is gated onto the BUS RD (15:00) line for use by the processor data paths. This latter case is used, for example, by the condition code instructions which alter the contents of the Processor Status Register.

3.3.5 Register (REG)

The 16 internal processor registers comprise the Scratch Pad Register. Eight of these are programmable general registers which include the Program Counter (PC) and Stack Pointer (SP). In the KD11-A Processor, the additional eight registers (not accessible to the program) are used for a variety of functions as shown on the block diagram. Such functions include: intermediate address (TEMP), source and destination data (SOURCE, DEST), a copy of the Instruction Register (IR), the last interrupt vector address (VECT), registers for console operation (TEMPC, ADRSC), and a stack pointer for operation of the KT11-D Memory Management Option (SP USER).

In summation, the data path logic is the fundamental section of the processor; it performs data storage, modification, and routing functions. The other two sections of the processor (interface and control) exist primarily to support the data path logic.

An important aspect of the data path logic is its expandability. The D MUX signals represent an outgoing bus and the BUS RD lines are a wired-OR input bus. Just as the Scratch Pad Register and the Processor Status Register are connected between these two signal buses, other devices can also be connected between them. For example, the KE11-E Extended Instruction Set option and the KE11-F Floating Instruction Set option are connected between these two signal buses for arithmetic expansion of the basic processor.

3.4 MICROCONTROL

The final section of the block diagram is the microcontrol logic which provides the required control signals for the data path logic and the interface logic. The prime element of the control logic is the read-only memory (ROM) which provides the microwords. The bits in each microword (U WORD), in turn, control machine operation as described in Chapter 2. Other elements within the MICROCONTROL section includes microaddress and microaddress modification logic that receives inputs from the ROM, the Instruction Register with associated decoding logic, various processor flags, and basic machine timing and flag control logic.

When an instruction is fetched from an external data storage location, the instruction enters the processor from the Unibus, passes through the D MUX, and is loaded into the Instruction Register under microword control. The output of the Instruction Register is decoded by combinational logic (IR DECODE) to provide the microbranch code (BUBC) for several branch conditions and the discrete auxiliary signals required by the condition code logic and ALU control logic. The last sections of logic are discussed immediately because of their interaction with the DATA PATHS section. The operation of the basic microcontrol follows.

3.4.1 Condition Codes Input

The condition codes are used to store information about the results of each instruction so that this information can be used by subsequent instructions. The information recorded in the condition code bits (N, Z, V, C) of the Processor Status Register differ for each instruction type, and often for the part of the instruction being executed. The decoded output of the IR DECODE logic and the select processor status (SPS) code of the microword determine which conditions are to be presented as the data input to the Processor Status Register. In addition, the SPS code determines when the Processor Status Register should be loaded directly from the D MUX.

3.4.2 ALU Control

The ALU control combinational logic receives the DAD (discrete alteration of data) code from the microword as a function of the IR decode logic. In general, the SALU and SALUM microword bits directly alter operation of the ALU; however, during the latter part of an instruction, where common instruction flow paths exist for several instructions, the DAD code is combined with the Instruction Register to alter operation of the ALU.

There are 32 possible ALU functions, depending on the state of SALU (03:00), ALUM and CIN. Some of these functions are contained in Table 3-2.

3.4.3 Flag Control

The flag control logic is closely related to the IR decode logic because certain instructions require specific flags, such as WAIT and HALT. Other flags exist for service of peripherals and console request, as well as for error conditions. Flip-flops within the flag control logic interact directly with the microbranch logic to provide the required branch conditions in the machine flow to provide flag service.

3.4.4 U Branch Control

In a microprogrammed computer, the next ROM address (next machine state) is dependent on a number of previous conditions. The purpose of the microword branch control (U BRANCH CONTROL) logic and the Branch Microprogram Test (BUT) Multiplexer is to select the next proper machine state. The microbranch control provides some of the inputs to the BUT decoding logic. The microbranch control combines the diverse instruction decoding of the Instruction Register and encodes it into two, three, four, or five bits of a microaddress alteration, called "basic microbranch codes," for specific BUTs [BUBC (BUT XX)]. For most of the complicated branches, such as the first instruction branch or some of the subsequent source or destination instruction branches, these codes are fairly extensive. They may also be fairly simple, consisting of only three bits or, in some cases, three bits of another BUT encoded with another single condition. The latter case is particularly true with the INSTR 2 BUBC and the (BYTE and INSTR 2) BUBC.

Table 3-2
Table of Combinations
74181 – Arithmetic Logic Unit

Selection				ALUM=H Logic Functions	Active High Data ALUM=L Arithmetic Operations	
S3	S2	S1	S0		CIN = 0	CIN = 1
L	L	L	L	F=$\overline{A}$	F=A	F=A plus 1
L	L	L	H	F=A+B	F=A+B	F=(A+B) plus 1
L	L	H	L	F=$\overline{A}$B	F=A+$\overline{B}$	F=(A+B) plus 1
L	L	H	H	F=0	F=minus 1 (2's COMPL)	F=0
L	H	L	L	F=$\overline{AB}$	F=A plus A$\overline{B}$	F=A plus A$\overline{B}$ plus 1
L	H	L	H	F=$\overline{B}$	F=(A+B) plus A$\overline{B}$	F=(A+B) plus A$\overline{B}$ plus 1
L	H	H	L	F=A+B	F=A minus B minus 1	F=A minus B
L	H	H	H	F=A$\overline{B}$	F=A$\overline{B}$ minus 1	F=A$\overline{B}$
H	L	L	L	F=$\overline{A}$+B	F=A plus AB	F=A plus AB plus 1
H	L	L	H	F=$\overline{A+B}$	F=A plus B	F=A plus B plus 1
H	L	H	L	F=B	F=(A+$\overline{B}$) plus AB	F=(A+$\overline{B}$) plus AB plus 1
H	L	H	H	F=AB	F=AB minus 1	F=AB
H	H	L	L	F=1	F=A plus A*	F=A plus A plus 1
H	H	L	H	F=A+$\overline{B}$	F=(A+B) plus A	F=(A+B) plus A plus 1
H	H	H	L	F=A+B	F=(A+$\overline{B}$) plus A	F=(A+$\overline{B}$) plus A plus 1
H	H	H	H	F=A	F=A minus 1	F=A

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

3.4.5 BUT MUX

The Branch Microprogram Test Multiplexer (BUT MUX) selects sets of address alterations to alter data into the Microprogram Pointer (UPP) which points to the next ROM address. The BUT MUX provides a 5-bit output with the number of possible inputs on the lowest order bits being greater than the number of inputs that can be selected for the higher order bits. This corresponds to the fact that few of the branches involve all five or six bits of address alteration. There are a number of address alterations that involve only one bit, usually the lowest order bit.

The gradation of inputs in the multiplexers is as follows: there are two 6-bit multiplexers for bit 0, a single 16-bit multiplexer for bit 1, 8-bit multiplexers for bits 2 and 3, and a 4-bit multiplexer for bits 4 and 5. Besides this ordering of multiplexers, the inputs to the BUT MUX also determine the required branch. The microbranch control logic provides wide branch encoding situations for instruction situations (INSTR 1, INSTR 2, and INSTR 3) and a 5-bit input is possible for the BUBC signal. In other cases, the Instruction Register itself may be used for a single BUBC bit code when the decision between a bit enabled or not enabled is simply a choice between two different microaddresses. The flag control logic also provides certain inputs which alter only one bit of the microaddress.

The actual selection of which of these inputs (wide or narrow branch, branch on instruction, branch on flag) is to be used, is determined by the microprogram branch field (UBF) of the microword. The UBF field directly selects which inputs of the multiplexers are applied to the microaddress alteration logic (the NOT OR gate on the block diagram).

3.4.6 U WORD Control ROM and U WORD Reg

The heart of the control logic is the microword control read-only memory (ROM) which stores 256 56-bit words. The format and purpose of these control words is described in more detail in Chapter 2. Basically, each of these control words represents a different machine state of the processor. The ROM provides a wired-OR output as indicated by BUS U (56:09) and BUS U (08:00). This wired-OR condition permits easy expansion of the processor as required by the KE11-E and KE11-F options.

The microword output of the ROM is applied to a Buffer Register (U WORD Register) that permits a microword to be used for machine control and selection of the next address, while the ROM itself is obtaining the contents of the next address. Although advantageous from a time standpoint, this implementation slightly increases the complexity of the hardware and concepts.

Each microword from the ROM consists of a control portion and a next address portion. At the beginning of the current machine state, a ROM output microword is clocked into the U WORD Register. The bits in the control portion of the microword select addresses, select multiplexers, and enable clocking gates (these gates enable clock pulses toward the end of the machine state). The bits in the address portion of the microword access the ROM to obtain the next ROM word. At this point, this address is fixed in the microword register and alteration for a BUT has not occurred.

The delay in using the buffer (U WORD Register) is fixed by the settling time of the flip-flops (approximately 15–20 ns). This is significantly better than the 60–90 ns required for addressing the ROM. For this reason, the buffer takes the delayed output of the ROM, clocks it at the beginning of the machine state, and provides it almost immediately (in that machine state) to the rest of the processor (data path, interface, and the microprogram control).

The clock for the U WORD Register is taken directly from the basic processor clocking and is related to the clock length selection bits in the microword control. The clock is a function of a machine cycle and is the last pulse edge of the previous machine cycle.

Each microword is divided into two segments: address and control. The address portion of the word is represented by BUS U (08:00) which is the address of the next ROM word. The control portion is represented by BUS U (56:09) which includes the control bits for the microword. The control bits are applied directly to the U WORD with the address bits passing through a NOT OR gate to the Microprogram Pointer (UPP) portion of the U WORD.

The outputs of the U WORD Register are diverse and are used throughout the processor. Outputs control the basic processor clock, microcontrol branching, and elements of the interface and data path. These outputs are indicated both by the labels on the U WORD Register outputs and by signals prefixed with a K2 on other blocks in the diagram.

3.4.7 Microaddress Alteration

Each microprogrammed word contains the address of the next microprogrammed word to be used by the processor. This address is referred to as the microprogram field (UPF) of the ROM. If this address were always exactly as specified by the ROM, it would receive little attention from the processor. However, alterations to this address are made for branching purposes. Therefore, there must be a method of modifying and storing this address so that the next specified word can be output in parallel with current word control. As shown on the block diagram, the hardware used to perform these functions consists of the NOT/OR gates on the UPF output [BUS U (08:00)], the output of the BUT MUX and the UPP Register. The base address of the UPF can be altered by the BUT MUX inputs, resulting in a different next ROM word address in the UPP Register.

In discussing the address in the microaddress loop, it is important to realize that an altered next address has been stored in the UPP Register and that alterations for the subsequent next address are fed to the NOT/OR gate. Both of these addresses are clocked simultaneously; therefore, the address fed through the NOT/OR gate is clocked into the UPP and the address that had been stored in the UPP is clocked out. Consequently, in any given microword, the control portion of the U WORD is performing manipulations while the UPP address portion of that word is addressing the next ROM word. The last UPP contents, the microaddress of the present U WORD, are stored in the Past Microprogram Pointer (PUPP) for reference. The logic diagram in Figure 3-10 shows, in simplified form, how an address change is performed.

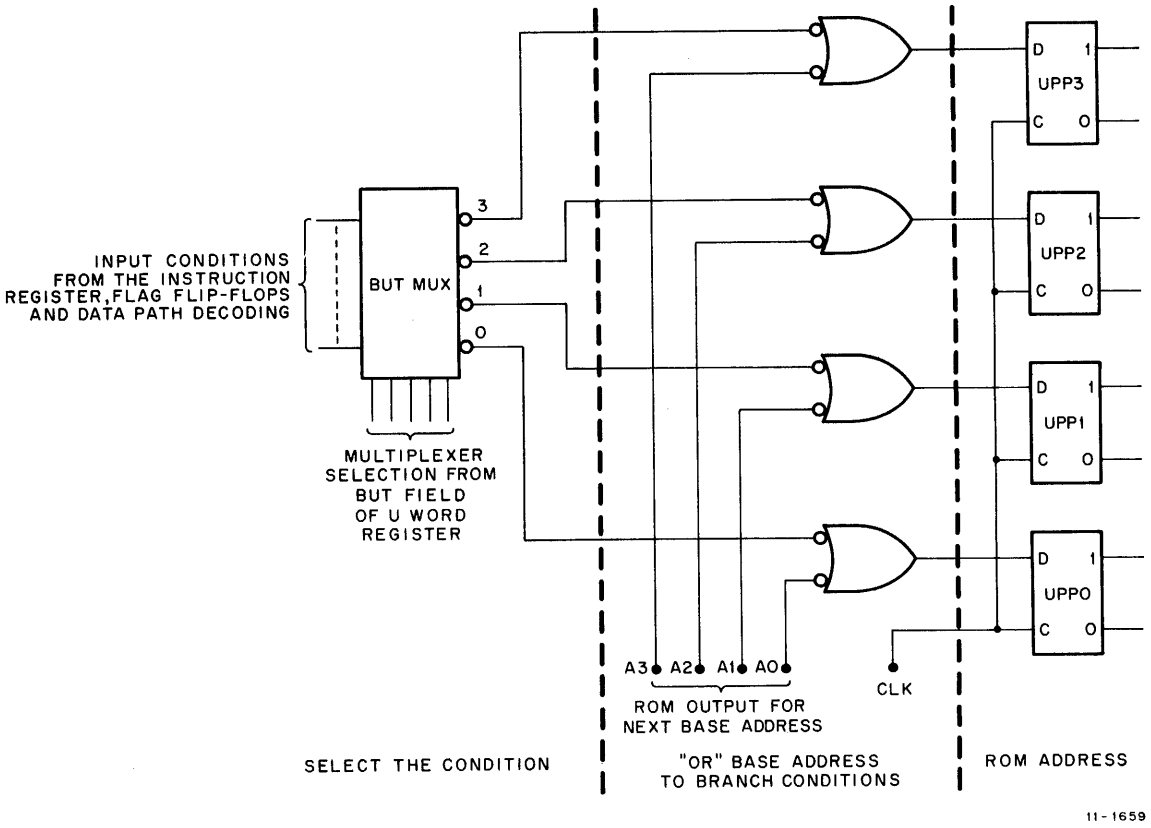


Figure 3-10 U Branch Control Sequence, Simplified Branching Operation

Another address in the address loop is the output of the ROM which has been selected by the next address from the UPP Register. This address does not appear immediately in the machine cycle (as is the case for the UPP next address) because ROM access time is greater than flip-flop settling time. However, it is present about midway through the U WORD state. This ROM output address, which appears on BUS (08:00), is a subsequent next address and is applied through the NOT/OR gate to the UPP Register. The next word data is becoming available across the entire ROM and is to be clocked in after the current machine state ends. If the subsequent next address is fixed (i.e., no branches are required), then there is no real difference between the address and control portion of the ROM/U WORD interface. In effect, the NOT/OR gate simply inverts the already complemented address output of the ROM. However, if a microbranch is to occur, it must occur at this point before the subsequent next address is clocked into the fixed UPP Register. The branch requires a subsequent next address from the ROM with 0s in it; it also requires the BUT MUX logic to input alterations to this address. Both of these occurrences require that the current microword has enabled appropriate control bits in the address and control sections.

Note that the microbranch test in a current word cannot alter the next word. However, it can alter the following word (the subsequent next word) as described below.

Assume that there are three microwords in sequence: A, B, and C. When the current word is A, the address portion of that word is causing word B to be accessed from the ROM (the address portion of word A selects the next word, which is B). Word B contains an address segment which is used for accessing word C and is present on BUS U (08:00). The address portion of word B, however, is a base address so that it can be altered if there is to be a branch. This alteration occurs in the NOT/OR gate while word A selects word B. The address for word C (contained in word B) can be either the base address for C or an altered address for C. For example, the altered address could be C1 or Cn, depending on how wide the branch is.

This technique means that selection of branch conditions and related enabling of that selection to the NOT/OR gate occurs a microword ahead of the word in which the branch takes place. During word A, a decision to branch can affect what word is used for word C, but it cannot affect word B. If a branch to C or Cn is desired, then conditions must be enabled to alter C in word A. By the time the processor goes to word B, the next address for word C is already fixed and stored in the UPP Register.

3.4.8 JAMUPP Logic

The microprogram address loop is also affected by the JAM Microprogram Pointer (JAMUPP) logic which alters the sequential nature of the microprogram. The JAMUPP logic provides a means of jamming an address into the UPP to modify the microprogram for certain conditions such as bus errors, stack overflow, auto restart, etc. This logic provides the next microword address directly, without regard for the previous microword's address. The output of the JAMUPP logic directly sets or directly clears the UPP Register flip-flops to establish the required address. This method differs from the normal NOT/OR gate outputs which are clocked into the UPP Register flip-flops.

3.4.9 PUPP Register

The output of the UPP Register is also fed to the Past Microprogram Pointer (PUPP) Register at each system clock. The PUPP Register maintains a history of the previous UPP and displays its contents on the maintenance console. Note that the PUPP indicates the current microword address. The PUPP Register is clocked each time the microword is clocked and the data input to the register is the address of the next ROM word present in the UPP Register. As the microword changes to the next word, the address of that word is clocked into the PUPP Register. The address of the current microword is therefore available and can be referenced on the maintenance console. The PUPP Register serves to identify the current microword address and to permit access to the ROM listings to determine which control bits should be enabled or disabled, and which operations would be taking place at this time. Note that the register itself does not perform these functions. It is the output of the register on the maintenance console display that permits determination of the current address.

3.4.10 BUPP & SR MATCH

The output of the UPP Register is also fed to the BUPP & SR MATCH logic, which is used for maintenance purposes. This logic compares the contents of the UPP Register [UPP (08:00)] with the low-order bits of the Switch Register [SR (08:00)] and generates a MATCH signal when UPP (08:00) equals SR (08:00). This MATCH signal can be used either as a synchronizing signal to trigger an oscilloscope, to stop the clock (halt the machine) in that word, provided the appropriate switches on the maintenance console are set. For example, to obtain a strobe signal upon entering ROM address 234, this address would first be set in the Switch Register on the programmer's console. When the contents of the UPP Register matched the Switch Register value, the end clocking pulse of that machine state would be enabled as a strobe signal. Because the UPP Register contains the next ROM address, the pulse would occur at the end of the machine state just prior to the microstate addressed in the Switch Register.

3.4.11 Clock Logic

The clock logic and related timing signals are basic to any processor. The clock signals that are generated are either used directly or are gated with enabling signals. These enabling signals are derived directly from either the microword or from machine states (flags, flip-flops, Unibus states, etc.). Data transfers and processor initializations within the processor itself are synchronous; they occur at specific times within machine states. Three different clock cycles are provided by the logic. This synchronous operation is designed for continuous running of the processor as the ROM sequences one microword after another. The processor should, however, be considered as a combination of both synchronous operation and asynchronous operation. The asynchronous nature of the processor is due to the fact that, upon certain conditions, the clock is turned off and waits for a restart. An obvious turn-off situation is that which occurs during Unibus data or bus ownership operations which are specified as asynchronous functions.

There are three functional elements that comprise the processor clock logic: the clock pulse generator, the clock control logic, and the clock enable gates. A simplified block diagram of the clock is shown in Figure 3-11.

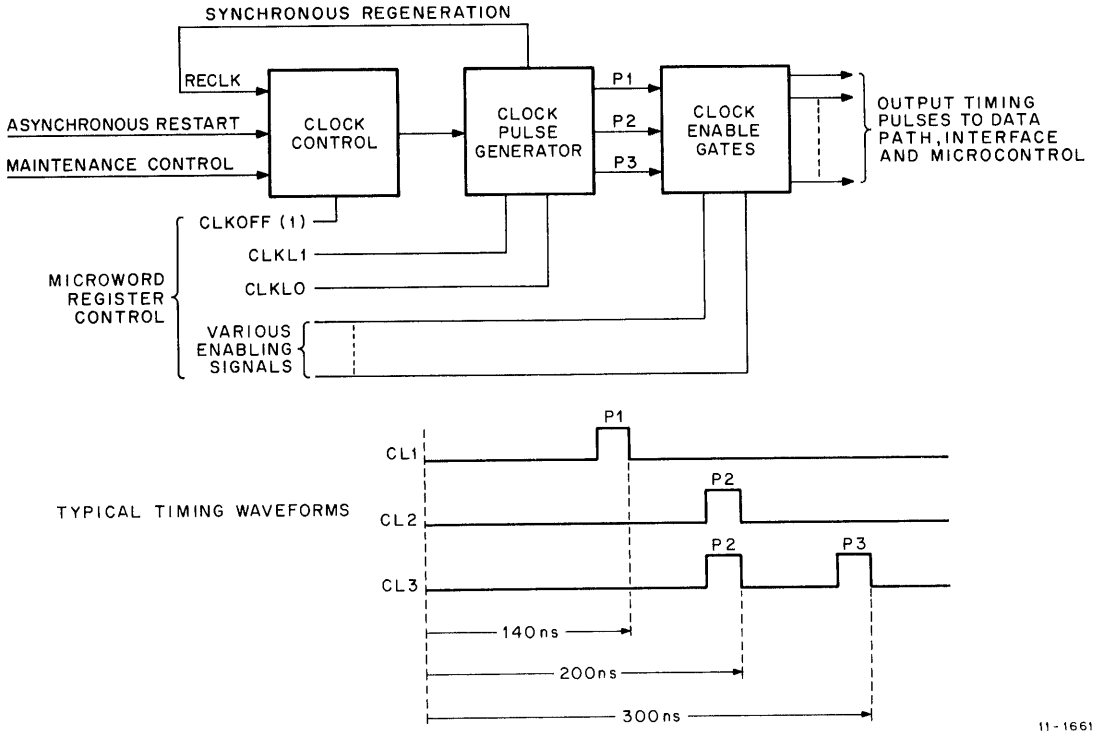


Figure 3-11 KD11-A Processor Clock, Block Diagram

3.4.11.1 Clock Pulse Generator – The clock pulse generator provides the system clock pulses when pulsed by the clock control logic. These clock pulses are used throughout the processor and are combined with the enable signals of the U WORD Register to act on the three major segments of the processor (INTERFACE, DATA PATHS, and MICROCONTROL). Three pulses are generated by the clock pulse logic: the CL1 cycle, which generates a P1 pulse; the CL2 cycle, which generates a P2 pulse; and the CL3 cycle, which essentially combines the CL1 and CL2 cycles and consists of P2 and P3 pulses. The prime purpose of the CL3 cycle is to complete a read/write cycle around the data path loops to allow the transfer to the D Register and from the Scratch Pad Register storage back into the Scratch Pad Register. The specific cycle length (CL1, CL2, CL3) for a microword is determined by microword clock control bits in that word. (See print D-CS-M7234-0-1 and Figure 3-11 for CLK waveforms.)

3.4.11.2 Clock Control – The clock control logic consists of a clock (CLK) and an idle (IDLE) flip-flop. The CLK flip-flop provides a pulse to the clock pulse generator when activated by the input signals RECLK, Asynchronous Restart, and Maintenance Control. The pulse is not generated when the Microword Register signal CLKOFF (1) is active and the IDLE flip-flop is set. The clock is restarted by the Asynchronous Restart or Maintenance Control signals which deactivate the IDLE flip-flop and reinitiate the CLK flip-flop.

3.4.11.3 Clock Enable Gates – The clock enable gates receive the pulses generated by the clock pulse generator. During each machine state, microcontrol bits control the passage of these clock pulses to specific registers. When it is desired to clock a register, the microcontrol word has the appropriate bit enabled and the clock pulse passes through the enable gate to the clock input of the specified register.

Table 3-3
KD11-A Functional Components

Component	Description	Input	Output
ADDRESS Display	Indicator lights located on the KY11-D Programmer's Console.	Contents of the Bus Address (BA) Register.	Displays contents of the BA Register on console ADDRESS display.
Arithmetic Logic Unit (ALU)	Four 74181 IC chips and one 74182 IC chip provide a 16-bit arithmetic logic unit with a lookahead carry. Dependent on mode selected, can perform up to 16 logic functions and up to 16 arithmetic functions. (See ALU TABLE, print D-BD-KD11-A-BD.)	Data: AIN—16-bit wide input from buffered BUS RD bus BIN—16-bit wide input from B MUX CIN—carry insert to LSB of ALU from CIN logic Control: SALUM, SALU (03:00) 5-bit wide control that specifies ALU function.	Data: Provides 16-bit output to either the D Register or to the BA Register through the BA MUX. Status: COUT 7, COUT 15, ALU 15 to input of COUT multiplexer.
Arithmetic Logic Unit Control (ALU CONTROL)	One 8233 IC (dual 2-line to 1-line multiplexer) and combinational logic. Generates control signals that are used to specify the ALU function.	ALU control signals from: microword bits, IR decode logic, and external control (KE11-E).	Five control signals, SALUM, SALU (03:00) that select the ALU function to be performed.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
B Constants	Combinational logic network providing elemental values for incrementation and decrementation. Also provides more complex constants such as trap vectors and masks.	Constants generated are a function of the following inputs: SBC (03:00) from the microword. STPM (04:02) from the trap sensing logic.	Selected constants applied to the B MUX.
B Multiplexer (B MUX)	Eight 74153 multiplexer IC chips. Provides the means of selecting the data input to the B input (BIN) of the ALU.	Control of the high and low bytes are independent signals from the microword. Any one of the following inputs can be selected: a. BC (15:00) (B constants) b. B (15:00) (direct) c. B (15:08, 15:08) (duplicate upper byte of B Register) d. B (07:00, 07:00) (duplicate lower byte of B Register) e. B (07:00, 15:08) (swap bytes of B Register) f. B (15:08=7, 07:00) (sign extend lower byte of B Register)	Provides 16-bit wide input to the B input of the ALU.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
B Register	Four 74174 IC chips provide a 16-bit temporary storage register for the B input of the ALU.	Input is loaded from the output of the D MUX and is therefore dependent on the D MUX selection.	Provides a data input to the B MUX. This input (which is the B Register output) is partitioned into a high (15:08) and low (07:00) byte.
Bus Address Multiplexer (BA MUX)	Four 8233 multiplexer IC chips. The BA MUX loads the BA Register.	Receives 16-bit wide input from either the Register Data bus (BUS RD) or the output of the ALU. A single microword control signal selects one of the two possible inputs. A high signal selects the ALU.	A 16-bit wide output that is loaded into the Bus Address (BA) Register.
Bus Address Register (BA Register)	Four 74174 IC chips that form a 16-bit temporary storage register.	Receives a 16-bit wide input from the BA MUX.	Transmits a 16-bit address to the Unibus. This address is applied through bus drivers to bus address lines BUS BA (17:00). The address is also applied to the address display and is decoded for processor address response.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Bus Register Data (BUS RD)	Four 74H04 IC chips that provide 16 inverters to establish proper input polarity for the A Input (AIN) of the ALU.	Receives input from three sources by means of a wired-OR bus: a. Scratch Pad Register data (16 bits) b. Processor Status Register (8 bits) c. External options (16 bits)	Output provides 16-bit data to either the A input (AIN) of the ALU or to the bus address (BA) multiplexer.
Branch Microtest Decode (BUT DECODE)	Network of combinational logic circuits that decodes the Microbranch Field (UBF) in each microword and generates auxiliary control signals.	UBF (04:00) from the microword.	Control signals, especially to the flag control logic.
Branch Microtest Multiplexer (BUT MUX)	Six multiplexer IC chips: a. three 16-line to 1-line type 74150 multiplexers b. two 8-line to 1-line type 74151 multiplexers c. one dual 4-line to 1-line type 74153 multiplexer	Any one of the following are selected by microword UBF (04:00) field: a. IR Register bits b. branch microbranch control signals c. IR decode signals d. machine status and flags	Control signals that allow modification of the microprogram field, UPF (07:00), prior to clocking the address into the UPP of the Microword Buffer (U WORD).

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Buffered Microprogram Pointer and Switch Register MATCH (BUPP & SR MATCH)	Nine exclusive-OR gates connected as an equivalence detector. Compares the contents of the Microprogram Pointer Register (UPP) with the Switch Register (SR) to generate a MATCH signal. The MATCH signal can be used to stop the clock during maintenance operation or to generate a scope synchronizing signal. Comparing the two registers permits stopping operation or monitoring operation at a specific ROM word.	BUPP (08:00) and SR (08:00)	UPP MATCH signals
Clock Control	Network of combinational logic circuits and delay line controls the CLK and IDLE flip-flops.	CLKOFF from the microword as well as various restart and continue signals.	Control signals to the clock pulse generator.
Clock Pulse Generator	Three delay lines selected by combinational logic circuits to generate the clock pulses specified by the current microword.	Pulse signal from clock control and the clock length signals CLKL0 and CLKL1 from the microword.	Timing pulses P1, P2, and P3. The RECLK signal which provides for continuous microword operation.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Clock Enable Gates	Combinational logic network that routes clock outputs to the INTERFACE, DATA PATHS, and MICRO-CONTROL portions of the processor.	Timing pulse P1, P2, or P3 from the clock pulse generator. Various clock enable signals: CLKIR, CLKBA, CLKB, CLKD, WRH, WRL bits from the current microword.	Various clock signals. (CLK IR, CLK D, CLK BA, etc.).
D Multiplexer (D MUX)	Eight 74153 multiplexer IC chips.	A 2-bit microcontrol field selects one of the following four inputs: a. BUS RD (including the Scratch Pad Register) b. D Register c. D Register shifted right d. Unibus data	The D MUX distributes 16-bit data word to: a. Instruction Register b. Scratch Pad Register c. B Register d. PS Register e. DATA display f. Internal data bus (D MUX) for basic machine and options
D Register	Four 74174 IC chips form a 16-bit temporary storage register.	Output of ALU.	Provides a 16-bit output to the D Multiplexer (D MUX) and to the Unibus data lines [BUS D (15:00)]

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
DATA Display	Four 7380 IC chips that invert the output of the D MUX for display on the console.	16-bit output of the D MUX.	16-bit data to the console DATA indicators.
Decoding (ADRS & DATA)	Combinational logic network that decodes the Bus Address Register and generates internal control signals for addressing processor registers. Sensing is provided for stack overflow situations and zero data in the D Register.	18-bit inputs from Bus Address (BA) Register and the D Register.	Processor status (PS) Address Stack Limit Register (SLR) address (KJ11-A Option) Scratch Pad Register (REG) address Switch Register (SR) address BOVFL STOP and BOVFL signals D Register zero data.
Drivers	Three 74H04 driver IC chips provide 18 buffer gates transmitting the UPP address to the PUPP Register and to an expansion ROM.	Microprogram Pointer (UPP) output of UPP Register.	Basic Microprogram Pointer (BUPP) for application to PUPP register. Expansion Microprogram Pointer (EUPP) for an expansion ROM (KE11-E, KE11-F).
Instruction Register (INSTR REG)	Four 74175 IC chips forming a 16-bit storage register that holds the instruction.	Output of D MUX clocked the instruction fetch sequence.	Output applied to IR decode logic where it is decoded and used to control the microprogram sequence. Some bits used directly for microbranching and Scratch Pad Register selection.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Instruction Register (IR) Decode	Large network of combinational logic circuits that decodes the Instruction Register instruction and generates appropriate control signals to perform the specified function.	16-bit instruction from the Instruction Register.	Generates control signals that are a function of: the operation code, instruction format, and specified register. Primary control signals are sent to the: ALU, microbranch control logic, and the BUT MUX.
JAM Microprogram Pointer (JAMUPP)	Sequential logic network consisting of flip-flops, one-shots, and decoders. This logic permits jamming an address into the UPP to modify the microprogram if certain conditions are present.	Internal control signals dependent on existing condition. Conditions causing JAMUPP are: a. bus errors b. stack overflow (red zone) c. auto restart (PWR UP) d. console switches (INIT)	Set and clear signals to UPP portion of the U WORD. Timing signals to load newly selected ROM word into the Microword Buffer (U WORD).
Processor Status (PS) Register	Four 7474 IC chips providing eight storage flip-flops to hold the processor status word. This word contains condition codes and processor priority.	Input may be either from D MUX (07:00) or may be from condition code logic.	Output may be gated onto Unibus on lines BUS D (07:00) or may be gated for processor use on lines BUS RD (07:00). Individual bits used from branch instruction decode and for microbranching.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Past Microprogramming Pointer (PUPP) Register	Two 74174 IC chips providing a 9-bit storage register for keeping a history of the previous UPP address, which is the present microword address.	Loaded with the contents of the UPP Register at each system clock.	Register contents display on KM11-A Maintenance Console option when used during maintenance operation.
Register (REG) (Scratch Pad Register)	Four 3101 IC chips providing a 16 x 16 read/write facility. Basically, this represents the 16 general-purpose processor registers (referred to as the Scratch Pad Register).	Data: 16-bit input from the D MUX Control: 4-bit address input from REG ADRS input logic 2-bit read/write control from microword	Provides 16-bit data word to BUS RD buffer for transfer to one of the following: a. AIN of ALU b. BA Multiplexer c. D Multiplexer
Register Address (REG ADRS) Input	Combinational logic network used as an address multiplexer to select one of the 16 general-purpose Scratch Pad Registers for reading or writing.	There are four possible sets of inputs. One of the four is selected by the microword signals: a. IR (02:00) – 3-bit destination field from instruction register b. IR (08:06) – 3-bit source field from instruction register.	Provides address selection to the register (REG).

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Register Address (REG ADRS) Input (Cont)		c. RIF (03:00) – 4-bit field from microword directly d. BA (03:00) – 4-bit field from Bus Address Register Microword signals are: SRD – Selects Register Destination, IR (02:00) SRS – Selects Register Source, IR (08:06) SRI – Selects Register Immediate, RIF (03:00) SRBA – Selects Register Bus Address, BA (03:00)	
Microbranch Control (U BRANCH CONTROL)	Large network of combinational logic circuits that provide data signals for modifying the base ROM address.	Instruction Register bits IR decode signals Machine status (i.e., switches, Unibus, control flip-flops, etc.).	Data signals to the BUT MUX. These signals are used to modify the basic ROM address as a function of BUT MUX selection from the microword.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Microword Control (ROM)	A read-only memory storing the KD11-A microprogram. The ROM stores 256 56-bit words. Fourteen ROM IC chips providing storage for the 256 words. Each chip stores 4 bits of the 56-bit word.	Contents of UPP Register selects the next control word to be retrieved from the ROM.	56-bit microword divided into address bits BUS U (08:00), and control bits BUS U (56:09).
Microword WORD Register (U WORD)	A 56-bit storage register consisting of type 74H74 and 74174 IC chips. This register is used to buffer the output of the ROM which provides the signals defining the operation of the KD11-A data path and control.	Output of the NOT/OR gate that receives inputs from the ROM, the BUT MUX, and the EUBC for U (08:00); output of the ROM directly for U (59:09).	UPP (08:00) are the nine low-order bits of the U word which are used to select the next U word. U WORD for U (56:09) have a variety of mnemonics related to their control functions.

Table 3-3 (Cont)
KD11-A Functional Components

Component	Description	Input	Output
Microprogram Pointer (UPP) Register	Five 74H74 IC chips forming an 8-bit address register. The UPP register points to the address of the next microword to be read.	Address of ROM location to be read during current machine cycle. The address loaded is a function of: a. UPF (07:00) of ROM word presently being addressed by the UPP Register. b. Basic Microbranch Control (BUBC) signals for microaddress modification (basic machine). c. Expansion Microbranch Control (EUBC) signals for microaddress modification (optional expansion).	UPP (08:00) – selects one of 256 control words stored in the ROM. It is the address portion of the U WORD Buffer noted above.

CHAPTER 4
MICROPROGRAM FLOW DIAGRAMS

4.1 SCOPE

This chapter describes and explains the microprogram flow diagrams (print D-FD-KD11-A-FD) that are included in the KD11-A Processor print set. These flow diagrams illustrate the operation of the processor on a machine state level; each operation shown on the flow diagram corresponds to one processor time cycle which, in turn, corresponds to one word of the microprogram ROM. The first section of this chapter describes the format, symbology, and layout of the flow diagrams; the second section explains their use.

4.2 HOW TO READ FLOW DIAGRAMS

Virtually all of the information needed to follow and understand the flow diagram is located on the flow diagram itself, however, it is necessary to understand the format of the diagram before this information can be easily used. The diagram contains two basic types of information: the operations performed by each machine state, and the flow of control from each machine state to all of the possible succeeding states.

As shown in Figure 4-1, only three basic symbols are used on the flow diagrams, the most important being the box that represents a specific machine state. This box contains information about the operations that take place during the machine time cycle for the microprogram word represented by the box. In certain cases, it also contains a test operation to determine the path of the control information. The oval represents an entry point in the flow path; the diamond represents an exit point.

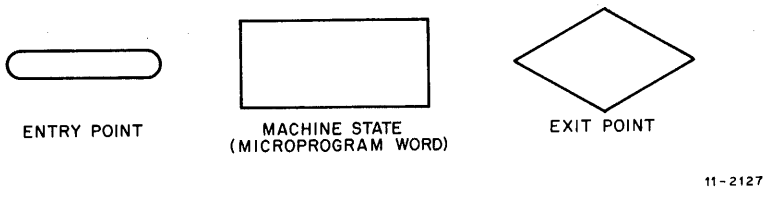


Figure 4-1 Basic Flow Diagram Symbols

In Figure 4-2, a representative example taken from one of the flow diagrams, the flow is shown for logic activated when the console START flip-flop is sensed. The figure is annotated to indicate the type of information found on the flows. Each of these items is discussed separately in the following paragraphs.

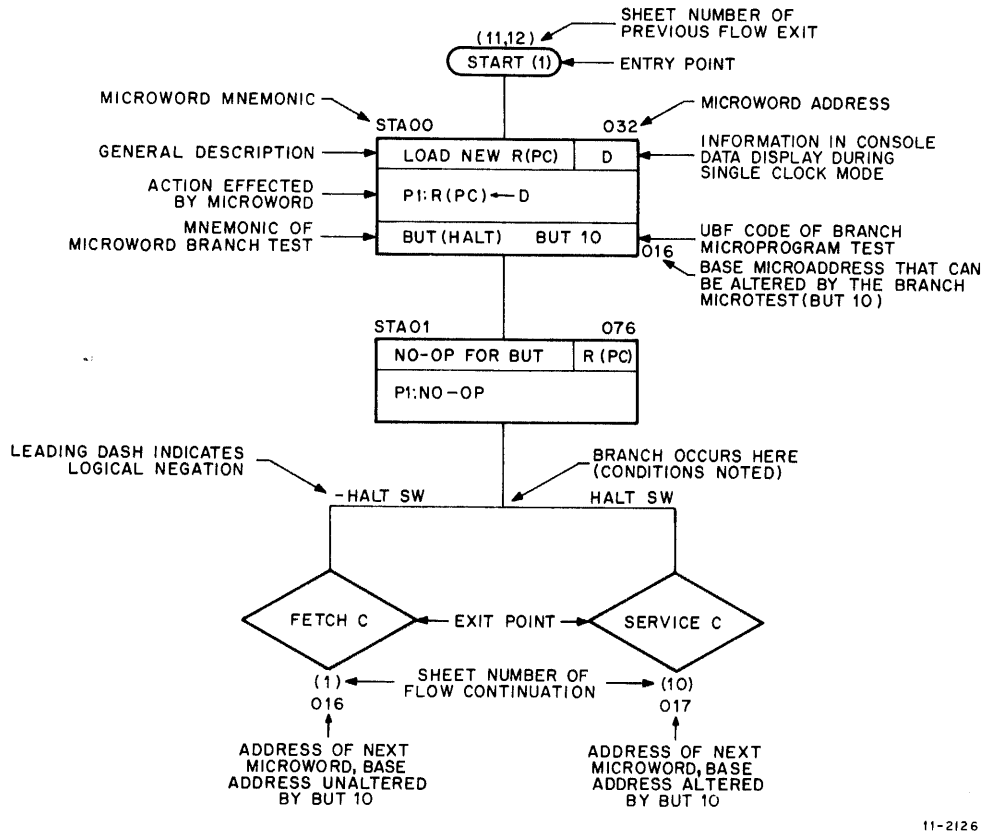


Figure 4-2 Flow Diagram Example

4.2.1 Entry Point

As shown in Figure 4-2, the entry point is labeled START (1). This indicates that the section of the flow beginning at this point is activated when the console START flip-flop is sensed. The numbers in parentheses above the entry point indicate pages of the flow containing previous flow information. Thus, (11) indicates print 11, which is the console loop flow diagram. Following this flow through to the bottom shows that START (1) on print 12 is one of the possible exit points for the console loop flow. The other number (12) above START (1) indicates that this flow can also be entered from a point on print 12. In this case, START (1) is an exit point for the LOAD ADRS switch function, provided BEGIN is true.

4.2.2 Microprogram Word

Each box on the flow diagram indicates one specific microprogram word (machine state). As shown in Figure 4-2, this box contains a variety of information.

Above the box, on the left hand side, is a mnemonic for the microword. In this case it is STA00, indicating it is the first (00) microword in the START (STA) sequence. Note that the numbers used with the mnemonic are decimal numbers and begin with 00. On the right hand side of the box is an octal number indicating the address of this microword in the ROM. Thus, whenever ROM address 032 is used, it is always the STA00 microword.

Directly below the microword mnemonic is a line containing a general description of the function performed by the microword. In this case, it is LOAD NEW R(PC), which indicates that the function of the microword is to load a new value into the Program Counter (PC) Register.

The main description of the microword operation is in a particular form which is explained more fully in Paragraph 4.2.5. In the case illustrated in Figure 4-2, it states: P1: R(PC) ← D. This means that the D Register is being placed (←) into a register R, called Program Counter, R(PC), at clock time P1 in a CL1.

The upper right hand section of the block indicates what information is shown in the console DATA display during this microstate. In this case, the D Register is displayed. Thus, when the maintenance console is being used and the program is being single clocked, the console DATA display allows the value being loaded into R(PC) to be observed. Operation at speed prevents this observation.

The bottom portion of the box contains the Branch Microprogram Test information which determines the sequence of microwords used to perform a specific function. In this case, the Branch Microprogram Test (BUT) is BUT(HALT). The other designation (BUT 10) indicates the octal microbranch field (UBF) code. It is important to note that a BUT in any microword affects *not* the next word, but the word *after* the *next* word.

The purpose of the BUT(HALT) branch test is to determine if the HALT/ENAB switch on the console is set to HALT. This condition is tested by microword 032 (STA00). The branch does not occur until after the next word, which is microword 076 (STA01). If the HALT/ENAB switch is set to HALT, then HALT is true and the flow exits at SERVICE C exit point. If the HALT/ENAB switch is set to ENAB, then -HALT SW is true, and the flow exits at the FETCH C exit point.

A more detailed discussion of BUT instructions is given in Paragraph 4.2.4.

4.2.3 Exit Points

At the bottom of each flow there is a diamond(s) containing the name of the next entry point for the flow. The number in parentheses beneath the diamond indicates the print of the flow diagram(s) containing the entry point. For example, in Figure 4-2, one of the possible exit points is FETCH C, therefore, the (1) indicates print 1 of the flow diagrams. Turning to print 1, it can be seen that FETCH C is one of the entry points. The other exit point on the figure is SERVICE C which is on print 10. On print 10, SERVICE C is one of the possible entry points.

The exit points also have a number located below the flow page reference; this is the octal address of the next ROM word. When the machine is microword STA01 with the microaddress 076 in the PUPP display of the KM11-A Maintenance Console, the UPP display indicates either 016 or 017, depending on the success of the branch microprogram test for BUT(HALT). Note the ORing of the low-order address bit over the base address (016 noted next to the BUT 10 entry of microword STA00) if the branch was successful; the next address would be 017.

4.2.4 Branch Microprogram Test (BUT)

Most machine states (or microprogrammed words) specify a unique succeeding state by means of a microprogram address in the microprogram word. However, the sequence of machine states can be altered. This allows a particular state, or sequence of states, to be shared by various larger sequences. For example, all instruction fetching is performed by one sequence of machine states. Once the instruction has been fetched, then specific sequences are followed, according to the requirements of the fetched instruction.

The BUT instructions may be divided into two functional groups: narrow or wide branch. The first type of BUT is the type previously explained in Paragraph 4.2.2. In this case, the condition of the HALT/ENAB switch is sensed and the branch occurs, depending on whether HALT SW is true or false. An example of a wide branch is shown on print 1 of the flow diagrams. In this case, BUT 37 [labeled BUT(INSTR 1)] is a function of Instruction Register encoding and the program may branch to any one of 25 different locations.

The name of the BUT indicates the possible branches that can be taken as a result of the BUT. For example, refer to page 6 of the flow diagrams at the first machine state after the TRAP A entry. The BUT in this machine state is BUT (MM FAULT), indicating that it is testing for faults in the KT11-D Memory Management option. The line after the next machine state follows one of two paths: MM FAULT or -MM FAULT. The BUT is further defined in Note 2 on the diagram.

Another example of the narrow BUT occurs after the RTS entry point on the same flow diagram. This test is called BUT (SERVICE C + FETCH C). Looking at the flow after the next machine state, it can be seen that the program can branch to either the SERVICE C or FETCH C exit point.

When a BUT instruction lists two or more possible branches as OR conditions, the priority is always from left to right. For example, in the expression BUT (SERVICE B + FETCH OVLAP + FETCH B), the service request always takes precedence over both the fetch overlap and normal fetch cycle entry. The expression also indicates that fetch overlap takes precedence over a normal fetch cycle.

Table 4-1 contains a listing of all the BUT instructions along with their microword addresses and microword mnemonics. Flow diagram sheet numbers are also included.

Notes on BUT instructions are included on each page of the flow diagrams. The notes pertain to the BUTs on that specific page and are used to clarify points not always obvious from the flows themselves. For example, there is a BUT on page 8 of the flow diagrams that is called BUT (NOWR + BYTEWR + WORDWR). By the conventions used, it is known that after the next machine state there is a branch to one of three places and that these three paths are labeled NOWR, WORDWR, and BYTEWR. However, the note on the flow diagram indicates that these branches provide for different register write operations as a function of the Instruction Register (IR) decoding.

In a number of instances, the machine state general description indicates that it is a NO-OP FOR BUT. This means that the previous entry requires an immediate branch before entering any other state but, because a branch can occur only after the next machine state, it is necessary to add a non-operational state after the BUT microword. This is the purpose of a NO-OP FOR BUT.

Some of the notes on the flow diagrams refer to a *working BUT*, which means that it performs a specific task and may or may not cause the flow to branch. As an example of a working BUT, refer to the second machine state in the RESET flow shown on page 6 of the flow diagrams. The BUT in this machine state is called: BUT (CBR2); INIT; DELAY. This BUT senses the HALT switch for a console bus request and branches as a function of HALT SW or -HALT switch. In addition to the branching, it also activates the INIT and RESTART delay, thereby making it a working BUT. Another working BUT is shown on the same page as the last machine state under the TRAP D sequence. This BUT is called BUT (REG DEP). This particular BUT is used in the sequential clearing of various TRAP request flags but does not cause any branching. The branching shown below the machine state is caused by the previous BUT [BUT (CBR1)].

Table 4-1
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
00	NO-OP	Everywhere except where a BUT is used.	–	–
01	CBR1	332	TRP20	6
02	CBR2	25	RST01	6
03	REG DEP	62	DEP01	12
		70	DEP07	12
		333	TRP21	6
04	REG EXAM	53	EXM01	12
		56	EXM04	12
		140	TRP16	6
05	BEGIN	51	LAD01	12
06	SWITCH	26	CON04	11
		315	CON13	11
07	INTR	22	SER10	10
10	HALT	32	STA00	12
11	MM FAULT	10	TRP03	6
12	D = 0	44	CON08	11
		342	SOB01	7

Table 4-1 (Cont)
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
13	Not Used			
14	Not Used			
15	JSR + JMP	151	JMP00	5
		153	JMP05	5
		154	JMP03	5
		155	JMP06	5
		235	JMP02	5
		302	JMP10	5
		305	JMP15	5
16	SERVICE C + FETCH C	110	NBR00	7
		117	SCC00	7
		125	MOV22	4
		277	RSR10	9
		306	JMP12	5
		324	RTS02	6
		340	BRA01	7

Table 4-1 (Cont)
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
16 (Cont)		344	SOB03	7
		345	SOB05	7
		350	CCC01	7
		356	MRK04	5
		366	DOP11	8
		367	DOP12	8
		374	SSL09	9
17	IR03	166	DST07	3
		167	DST06	3
		176	MOV06	4
		177	MOV05	4
20	BYTE + SERVICE + FETCH	160	MOV19	4
		170	MOV18	4
21	IR03, BYTE AND SOURCE	206	MOV08	4

Table 4-1 (Cont)
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
22	BYTE AND SOURCE	171	MOV00	4
		172	MOV01	4
		174	MOV02	4
		207	MOV11	4
23	Not Used			
24	CBR + HALT	255	CON12	11
25	BR, WAIT + FETCH	20	SER07	10
26	REQUESTS	2	SER01	10
		6	SER04	10
		17	SER02	10
		114	SER00	10
		123	SER03	10
27	SERVICE B + FETCH OVLAP + FETCH B	003	MOV21	4
		132	SXT00	8
		135	SWB01	7
		271	RSR01	9
		273	RSR04	9

Table 4-1 (Cont)
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
27 (Cont)		363	DOP01	7
		364	DOP03	7
		370	DOP14	8
		371	DOP16	8
		372	SSL01	7
		373	SSL03	7
30	Various Switches	45	CON10	11
31	NOWR + BYTEWR + WORDWRITE	102	DOP02	7
		104	SSL02	7
		105	SSL00	7
		120	DOP15	8
32	Not Used			
33	OB + INSTR 4	260	DST03	3
		266	DST14	3
34	INSTR 4	237	DST16	3

Table 4-1 (Cont)
BUT CHART

BUT Number (octal)	BUT Name	Microword Location	Microword Mnemonic	Flow Diagram Sheet
35	OB + INSTR 3	240	SRC03	2
		247	SRC14	2
36	INSTR 3	137	SRC16	2
37	INSTR 1	4	FET04	1

4.2.5 Operation Symbols

Previous paragraphs have discussed the basic symbology and format of the KD11-A flow diagrams. Another set of symbols to understand is the ISP notation which provides the detailed description of each machine state. Although ISP is covered in the *PDP-11/40 Processor Handbook,* this paragraph explains KD11-A flow diagram usage.

In KD11-A ISP notations, a few general rules are helpful. The first item appearing in each statement always has a specific clock pulse, which indicates at which clock time the machine state operation occurs. The clock pulse is always P1, P2, or P3 and is associated with CL1, CL2, and CL3, respectively. A statement describing the machine state operation follows the clock pulse. These statements are always read from *right* to *left.* For example:

P2: D ← R0

In the above statement, D indicates the processor D Register and R0 indicates one of the eight general registers. The above statement is read: at clock time P2, the D Register is loaded with the contents of the R0 Register, or D gets R0.

A variation of the above is used when a register address appears in parentheses after the designation R (register). For example:

P1: B ← R(SF)

The above statement is read: at clock time P1, the contents of the register, addressed by the IR source field, is loaded into the B Register. This type of notation is used because a number of registers or locations may be used to store SOURCE data. An example of this notation is shown on print 2 of the flow diagrams. This print carries a note which states that the Source Register is selected by the IR (Instruction Register).

A more complex example of machine state operation statements is:

P2: D ← *f* DAD {R(SF) AND B} ; DAD 14

Before reading this expression, it is necessary to know that the symbol *f* indicates "as a function of," that the term to the right of the semicolon is a separate statement, and that the items in brackets are considered first. Thus, the statement is read: D gets a function of the register, specified by the source field and the contents of the B Register. The function is determined by the DAD (discrete alteration of data) code and its operation on the ALU; the DAD 14 function is used. The user can look up DAD 14 in the U WORD TABLES in print D-BD-KD11-A-BD to find the function of DAD 14. The table indicates that DAD 14 is used for ALU CNTL *f* IR; in other words, the Instruction Register determines what function the ALU is to perform.

There are times that two or more completely separate actions occur at the same time pulse. The different actions are either separated by a semicolon, or by placing them on different lines, or both. For example:

P2: BA ← R(DF); DATI
    D ← R(DF) PLUS 2

This indicates that three separate actions take place at clock time P2. First, the register defined by the destination field of the IR is loaded into the Bus Address Register. Secondly, a DATI bus transfer is begun. And finally, the register defined by the destination field plus 2 is loaded into the D Register.

Note that the usual use of parentheses is to further define the preceding symbol. R(PC) means that the register used as the Program Counter in the Scratch Pad Registers is being referenced. This is true for all situations except R(DF), R(SF), and R(BA) where specific address bits in the IR (for destination field and source field) or the bus address are used to select a Scratch Pad Register. A note to this effect occurs on print 1 of the flow diagram.

The above example would be exactly the same if all three actions had simply been separated by semicolons:

P2: BA ← R(DF); DATI; D ← R(DF) PLUS 2

or if each separate action had been placed on a separate line:

P2: BA ← R(DF)
DATI
D ← R(DF) PLUS 2

Statements that have an equal sign, such as SBC=7, DAD=10, BUS CODE=06, etc., are explanatory statements that list the codes internally generated during performance of the operation specified in the box. The meaning of these codes can be determined by referring to the page of tables in the block diagram prints, D-BD-KD11-A-BD. For example:

P3: PS(C) ← D00; SPS=1

The above expression indicates that the value on bus data line D00 is to be loaded into bit C of the processor status (PS) word during clock pulse time P3. The explanatory expression after the semicolon (SPS=1) indicates that a specific U WORD code is used to perform this function. By referring to the table, it can be seen that SPS code 1 is used to clock bit C of the PS word.

4.3 FLOW DIAGRAM EXAMPLES

Once the format of the flow diagrams is understood, it is possible to follow the flows through any instruction sequence. Examples of following an operation through the flow diagrams are given in Tables 4-2 and 4-3.

In the example in Table 4-2, the following instruction (not micro) program is present:

Program Address	Contents
5000	ADD (1), (2)
R(SF) = (R1) = 300 (R1)	5
R(DF) = (R2) = 400 (R2)	5

In effect, the operation adds two numbers together. The instruction ADD (1), (2), which is 061112 in octal form, is loaded at location 5000. The first number to be added (R1) is the number 5 (octal) stored at address 300. The second number (octal 5) is stored at address 400.

Based on the above conditions, Table 4-2 lists all microwords in the flow when performing this operation. The table also includes a description of what is happening during each machine state. If the table is followed carefully while referring to the flow diagrams, the operations should be apparent.

Table 4-3 describes a subtract operation and is identical to Table 4-2 in format except that the description column has been eliminated to allow the reader to determine if he can follow the table and the flows by himself.

Two tables are included in this chapter as an aid in finding specific microwords on the flow diagrams. Table 4-4 is a numerical listing of all microwords in the ROM and includes the mnemonic, a general statement of the function, and the page of the flow diagrams on which it is shown.

Table 4-5 lists all microwords in alphabetical order according to the microword mnemonic. The only other entry in this table is the ROM address. Once the ROM address is found on Table 4-5, then Table 4-4 can be used to find the microword on the flows.

Table 4-2
Flow Diagram Example 1

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation	Description
FET02	016	001	5000	P1: BA ← R (PC); DATI; CLKOFF; SPS=0	The contents of the PC is loaded into the Bus Address Register; a Unibus data transfer is performed to bring the instruction into the processor. The address of the instruction [ADD (1), (2)] is displayed.
FET03	001	004	061112	P1: IR, R(IR), B ← UNIBUS DATA	The instruction (Unibus data) is loaded into the B Register, a Scratch Pad Register, and the Instruction Register. The Unibus data for the instruction is displayed.
FET04	004	005	5000	P2: D, BA ← R(PC) PLUS 2; DATI IF OVLAP FETCH; BUT INSTR 1	The value of PC plus 2 is loaded into both the Bus Address and D Registers. No DATI is performed for OVERLAP FETCH. Branch test BUT (INSTR 1) is performed, which is the first wide branch for all instructions. Value of current PC is displayed.
FET05	005	141	5002	PC1: R(PC) ← D	Program Counter is updated by moving data in the D Register (which contains next PC+2) into the PC. The new PC is displayed. Note that the display of D in a given microword is a display of what is in D at the beginning of the microword – not what will be clocked into it this microword. The next microword to be used is at ROM address 141 for a source mode 1 (SM1) calculation on sheet 2. This address occurs as a function of the IR and BUT (INSTR 1) with ALLDOP * -SM0 indicating a double operand instruction with a non-zero source mode.

Table 4-2 (Cont)
Flow Diagram Example 1

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation	Description
SRC00	141	247	300	P1: BA ← R(SF); DATI; DAD=01; MM=14	The register specified by the source field (address of the source operand) is loaded into the Bus Address Register. The source address is displayed. **NOTE** **It would be normal to expect the location of this microword to be 100 because that was the value of the previous UPP. However, the UPP was modified by BUT (INSTR 1) as a function of the instruction, resulting in ROM address 141 for this microword.**
SRC14	247	250	061112	P1: NO-OP; CLKOFF BUT (OB+INSTR 3)	This is a no operation word to provide a BUT. Clock is turned off to wait for Unibus response to DATI.
SRC15	250	161	5	P1: B, R(SOURCE) ← UNIBUS DATA	The source operand (the number 5) is taken from external memory and stored in a temporary register R(SOURCE). The value of the operand is displayed. The next microword to be used is at ROM address 161 for a destination mode 1 (DM1) calculation on sheet 3. This address occurs as a function of the IR and BUT (OB+INSTR 3) with PARTDOP * -SM0 * DM0 indicating a double operand instruction with a non-zero destination mode.

Table 4-2 (Cont)
Flow Diagram Example 1

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation	Description
DST00	161	266	400	P1: BA ← R(DF); DATIP; DAD=07; MM=01	The register specified by the destination field (address of the destination operand) is loaded into the Bus Address Register. The destination address is displayed. Note that this microword address was modified by BUT (OB+INSTR 3). Referring to the flow diagram, the output of SRC15 followed the path marked -OB because an odd byte was not being processed.
DST14	266	267	061112	P1: NO-OP; CLKOFF BUT (OB+INSTR 4)	This is a no operation word to allow for a BUT. Clock is turned off to await Unibus response to the DATIP.
DST15	267	225	5	P1: B, R(DEST) ← UNIBUS DATA	The destination operand (the number 5) is taken from external memory and stored in a temporary register, R(DEST), and in the B Register. The value of the operand is displayed. The next microword address (225) results from the BUT (OB+INSTR 4) for the PARTDOP * -DM0 condition.
DOP03	225	367	5	P2: D ← f DAD R (SOURCE) AND B (DATO+DATOB) DAD=17; MM=01	The source operand and the B Register (storing the destination operand) are loaded into the D Register as a function of DAD. In other words, the source and destination operands are added and moved to the D Register. The source operand is displayed.

Table 4-2 (Cont)
Flow Diagram Example 1

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation	Description
DOP12	367	375	12	P1: ALTER COND CODES CLKOFF; DAD=12; SPS=3 BUT (SERVICE C + FETCH C)	The condition codes are altered and the result of the addition of the source and destination operands is displayed. (Note that adding octal 5 to octal 5 results in octal 12.)
DOP20	375	016	12	P1: NO-OP	This is a no operation required by the BUT in the previous word. The BUT determines whether the processor is to enter the SERVICE or FETCH flows.
FET02	016	001	5002	P1: BA ← R(PC); DATI; CLKOFF; SPS=0	Fetch of next instruction.

Table 4-3
Flow Diagram Example 2

Program Conditions	
Address	**Contents**
5000	SUB #20, @ #6000
5002	20
5004	6000
5006	NEXT INSTRUCTION
6000	30

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation
FET02	016	001	5000	P1: BA ← R(PC); DATI CLKOFF; SPS=0
FET03	001	004	162737	P1: IR, R(IR), B ← UNIBUS DATA
FET04	004	005	5000	P2: D, BA ← R(PC) PLUS 2; NO OVLAP FETCH BUT (INSTR 1)
FET05	005	142	5002	P1: R(PC) ← D
SRC01	142	240	5002	P2: BA ← R(SF); DATI; DAD=01; SBC=03 D ← R(SF); PLUS 1; MM=14
SRC03	240	250	5004	P1: R(SF) ← D; CLKOFF BUT (OB+INSTR 3)
SRC15	250	163	20	P1: B, R(SOURCE ← UNIBUS) DATA
DST04	163	264	5004	P2: BA ← R(DF) DATI D ← R(DF) PLUS 2 P3: R(DF) ← D; CLKOFF **NOTE** **New D content does not occur until end of microword.**
DST12	264	265	6000	P1: B, R(DEST) ← UNIBUS DATA

Table 4-3 (Cont)
Flow Diagram Example 2

Microword Mnemonic	ROM Address (PUPP)	Next ROM Address (UPP)	DATA Display	Operation
DST13	265	266	6000	P1: BA ← R(DEST) DATIP; DAD=01; MM=01
DST14	266	267	162737	P1: NO-OP; CLKOFF; BUT (OB+INSTR 4)
DST15	267	227	30	P1: B, R(DEST) ← UNIBUS DATA
DOP05	227	365	20	P1: B ← R(SOURCE)
DOP06	365	367	30	P2: D ← R(DEST) MINUS B; DAD=10 DATO; MM=01
DOP12	367	375	10	P1: ALTER COND CODES; CLKOFF; DAD=12; SPS=3; BUT (SERVICE C + FETCH C)
DOP20	375	016	10	P1: NO-OP – If no service request, go to FET02

Table 4-4
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
000	FET01	Fetch next instruction	1
001	FET03	Store instruction	1
002	SER01	Clock for PTR	10
003	MOV21	Sign extend byte data	4
004	FET04	Modify register (PC)	1
005	FET05	Restore modified register (PC)	1
006	SER04	Clock for PTR	10
007	TRP08	Get new status	6
010	TRP03	Form, store trap vector	6
011	CON01	Display register (PC)	11
012	SER06	Await bus busy	10
013	FET00	Fetch next instruction	1
014	SER09	Wait for interrupt	10
015	SER05	No-op for BUT	10
016	FET02	Fetch next instruction	1
017	SER02	Clock for PTR	10
020	SER07	Clock again for PTR	10
021	SER08	No-op for BUT	10
022	SER10	Store vector, flags	10
023	SER11	No-op for BUT	10
024	CON03	Display register	11
025	RST01	Reset delay and INIT	6
026	CON04	Test for switch	11
027	CON07	Contact bounce count	11
030	CON05	No-op for console entry	11
031	EXM06	Get data, timeout flag	12
032	STA00	Load new register (PC)	12
033	LAD03	Display zero data	12
034	DEP00	Load console address	12
035	EXM00	Load console address	12
036	CNT00	No-op after a BUT	12
037	LAD00	Get address data from SR	12
040	RST02	No-op for BUT	6
041	CON02	Await bus busy	11
042	RST04	No-op for fetch entry	6
043	RST03	No-op for console entry	6
044	CON08	Test count	11
045	CON10	Test which switch	11
046	CON06	No-op for BUT	11
047	CON09	Increment count	11
050	CON11	Load last console address	11
051	LAD01	Store data as console address	12
052	LAD02	Display console address	12
053	EXM01	Console flags	12
054	EXM02	Conditional plus 1	12
055	EXM05	Read register of bus address	12
056	EXM04	Console flag	12

Table 4-4 (Cont)
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
057	EXM03	Conditional plus 1	12
060	EXM07	Store data	12
061	EXM08	Display data	12
062	DEP01	Console flags	12
063	DEP02	Conditional plus 1	12
064	DEP03	Conditional plus 1	12
065	DEP04	Get deposit data from SR	12
066	DEP05	Store deposit data	12
067	DEP06	Load console address	12
070	DEP07	Load deposit data	12
071	DEP08	No-op for BUT	12
072	DEP09	Deposit data	12
073	DEP10	Deposit data	12
074	DOP21	Alter codes	8
075	SSL11	Alter codes	9
076	STA01	No-op for BUT	12
077	TRP15	Enable new status	6
100	FET07	No-op after a BUT	1
101	RTI00	Get new register (PC), modify	6
102	DOP02	Put destination into B	7
103	DOP00	Put source into B	7
104	SSL02	Operate upon destination	7
105	SSL00	Put destination into B	7
106	RSR00	Operate upon destination	9
107	RSR02	Put destination into B	9
110	NBR00	No-op after a BUT	7
111	BRA00	Add half of offset	7
112	MRK00	Double offset	5
113	TRP12	Deskew word for DATO	6
114	SER00	Clock for PTR	10
115	TRP09	Store new status	6
116	CCC00	Mask register (IR) for PS mask	7
117	SCC00	Mask register (IR) for PS mask	7
120	DOP15	Put destination into B	8
121	DOP13	Put source into B	8
122	CON00	Display register (PC)	10
123	SER03	Clock for PTR	10
124	RTS00	Get new register (PC)	6
125	MOV22	Alter condition codes	4
126	TRP06	Form, store trap vector	6
127	RST00	Get reset data display	6
130	SOB00	Decrement count	7
131			
132	SXT00	Extend sign	8
133			
134	SWB00	Put destination into B	7
135	SWB01	Swap bytes	7

Table 4-4 (Cont)
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
136	FET06	Modify, store register (PC)	1
137	SRC16	Duplicate upper byte	2
140	TRP16	Load new status	6
141	SRC00	Get source data	2
142	SRC01	Get source data, modify	2
143	SRC04	Get address, modify, restore	2
144	SRC02	Get source data, modify	2
145	SRC05	Get address, modify, restore	2
146	SRC06	Get index data, modify	2
147	SRC09	Get index data, modify	2
150	TRP07	Form, store trap vector	6
151	JMP00	Get destination address	5
152	JMP01	Post modification	5
153	JMP05	Get address, modify, restore	5
154	JMP03	Get destination address, modify	5
155	JMP06	Get address, modify, restore	5
156	JMP08	Overlap, modify register (PC)	5
157	JMP07	Overlap, modify register (PC)	5
160	MOV19	Get destination data	4
161	DST00	Get destination data	3
162	DST01	Get destination data, modify	3
163	DST04	Get address, modify, restore	3
164	DST02	Get destination data, modify	3
165	DST05	Get address, modify, restore	3
166	DST07	Overlap, modify register (PC)	3
167	DST06	Get index data, modify	3
170	MOV18	Get destination data	4
171	MOV00	Load destination address	4
172	MOV01	Load destination address, modify	4
173	MOV03	Get address, modify, restore	4
174	MOV02	Load destination address, modify	4
175	MOV04	Get address, modify, restore	4
176	MOV06	Overlap, modify register (PC)	4
177	MOV05	Get index data, modify	4
200	MOV16	Store data	4
201	MOV17	Store data	4
202	MOV14	Load byte data	4
203	MOV13	Load byte data	4
204	MOV20	Store destination data	4
205	MOV15	Store justified data	4
206	MOV08	Store index data	4
207	MOV11	Store address data	4
210	MOV12	Load destination address	4
211	SSL10	Alter code PS(C)	9
212	MOV09	Store indexed destination address	4
213	MOV10	Get indexed address	4
214	TRP05	Store MM vector	6

Table 4-4 (Cont)
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
215	TRP02	Get new status	6
216	TRP04	Get MM vector (250)	6
217	FET08	New instruction from MM	1
220	SSL06	Operate upon destination, store	9
221	SSL04	Negate destination, store	9
222	SSL08	Operate upon destination	9
223	SSL07	Negate destination	9
224	DOP07	Operate upon B, source, store	8
225	DOP03	Operate upon B, source, store	8
226	DOP04	Put source into B	8
227	DOP05	Put source into B	8
230	DOP09	Operate upon B, source	8
231	DOP10	Operate upon B, source	8
232	RSR06	Operate upon destination	9
233	RSR08	Operate upon destination	9
234	SXT01	Extend sign, store	8
235	JMP02	Get destination address	5
236	SSL05	Swap bytes, store	9
237	DST16	Duplicate upper byte	3
240	SRC03	Restore modified base	2
241	SRC07	Store index data	2
242	SRC08	Get indexed source data	2
243	SRC10	Store index data	2
244	SRC11	Get indexed address data	2
245	SRC12	Store address data	2
246	SRC13	Get source data	2
247	SRC14	No-op for BUT	2
250	SRC15	Store source data	2
251	SRC17	Store justified data	2
252	FET09	Store new instruction	1
253	SSL12	Alter code PS(C)	9
254	DOP22	Alter code PS(C)	8
255	CON12	Display status	11
256			
257	MOV07	Restore base address	4
260	DST03	Restore modified base	3
261	DST09	Store index data	3
262	DST10	Get indexed destination data	3
263	DST11	Get indexed address	3
264	DST12	Store address data	3
265	DST13	Get destination data	3
266	DST14	No-op for BUT	3
267	DST15	Store destination data	3
270	DST17	Store justified data	3
271	RSR01	Store destination	9
272	RSR03	Operate upon destination	9
273	RSR04	Duplicate byte, store	9

Table 4-4 (Cont)
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
274	RSR05	Alter codes	9
275	RSR07	Store destination	9
276	RSR09	Duplicate byte, store	9
277	RSR10	Alter codes, finish store	9
300	JMP04	Store destination address	5
301	JMP09	Store index data	5
302	JMP10	Get indexed address	5
303	JMP11	Store destination address	5
304	JMP14	Store index data	5
305	JMP15	Get destination address	5
306	JMP12	Get destination address	5
307	JSR00	Modify Stack Pointer	5
310	JSR01	Stack Linkage Pointer	5
311	JSR02	Get new linkage	5
312	JSR03	Store new linkage	5
313	JMP13	Store as new register (PC)	5
314			
315	CON13	Test for switch	11
316			
317	TRP01	Form, store trap vector	6
320	RTI01	Store new register (PC)	6
321	RTI02	Get new status, modify	6
322	RTI03	Store new status	6
323	RTS01	Store new register (PC)	6
324	RTS02	Get top of stack, modify	6
325	RTS03	Store top of stack	6
326	TRP10	Modify stack pointer	6
327	TRP11	Store old status on stack	6
330	TRP13	Modify stack pointer	6
331	TRP14	Store old PC on stack	6
332	TRP20	Get new PC	6
333	TRP21	Store new PC	6
334	TRP18	Get new status	6
335	TRP19	Store new status	6
336	TRP00	Jam register SP to 4	6
337	TRP17	Form, store power up vector	6
340	BRA01	Modify PC	7
341	BRA02	Rest of offset, modify	7
342	SOB01	Test count	7
343	SOB02	Mask IR register for offset	7
344	SOB03	Subtract half of offset	7
345	SOB05	No-op for BUT	7
346	SOB04	Subtract half of offset	7
347	SOB06	No-op for BUT	7
350	CCC01	Complement PS mask bits	7
351	CCC02	AND PS mask to PS	7
352	SCC01	OR PS mask to PS	7

Table 4-4 (Cont)
Microwords (Numerical Order)

ROM Address	Microword Mnemonic	General Function	Flow Print
353	MRK01	Modify PC with offset	5
354	MRK02	Form new Stack Pointer	5
355	MRK03	Stack points to old R5	5
356	MRK04	Load R5 with old R5	5
357	MRK05	Load PC with old PC	5
360	DOP19	Alter codes, word store	8
361	DOP18	Alter codes, byte store	8
362	DOP17	Alter codes, no store	8
363	DOP01	Subtract B from destination	7
364	DOP03	Operate upon B, source	7
365	DOP06	Subtract B from destination, store	8
366	DOP11	Duplicate lower byte, store	8
367	DOP12	Alter codes, finish store	8
370	DOP14	Subtract B from destination	8
371	DOP16	Operate upon B, source	8
372	SSL01	Negate destination	7
373	SSL03	No-op for BUT	7
374	SSL09	Duplicate byte, store	9
375	DOP20	No-op for BUT	8
376	RSR11	No-op for BUT	9
377			

Table 4-5
Microwords (Alphabetical Order)

Microword Mnemonic	ROM Address	Microword Mnemonic	ROM Address	Microword Mnemonic	ROM Address
BRA00	111	DOP00	103	EXM00	035
BRA01	340	DOP01	364	EXM01	053
BRA02	341	DOP02	102	EXM02	054
		DOP03	225	EXM03	057
		DOP04	226	EXM04	056
CCC00	116	DOP05	227	EXM05	055
CCC01	350	DOP06	365	EXM06	031
CCC02	351	DOP07	224	EXM07	060
		DOP08		EXM08	061
		DOP09	230		
CON00	122	DOP10	231		
CON01	011	DOP11	366	FET00	013
CON02	041	DOP12	367	FET01	000
CON03	024	DOP13	121	FET02	016
CON04	026	DOP14	370	FET03	001
CON05	030	DOP15	120	FET04	004
CON06	046	DOP16	371	FET05	005
CON07	027	DOP17	362	FET06	136
CON08	044	DOP18	361	FET07	100
CON09	047	DOP19	360	FET08	217
CON10	045	DOP20	375	FET09	252
CON11	050	DOP21	074		
CON12		DOP22	254		
CON13	315			JMP00	151
				JMP01	152
		DST00	161	JMP02	235
CNT00	036	DST01	162	JMP03	154
		DST02	164	JMP04	300
		DST03	260	JMP05	153
DEP00	034	DST04	163	JMP06	155
DEP01	062	DST05	165	JMP07	157
DEP02	063	DST06	167	JMP08	156
DEP03	064	DST07	166	JMP09	301
DEP04	065	DST08		JMP10	302
DEP05	066	DST09	261	JMP11	303
DEP06	067	DST10	262	JMP12	306
DEP07	070	DST11	263	JMP13	313
DEP08	071	DST12	264	JMP14	304
DEP09	072	DST13	265	JMP15	305
DEP10	073	DST14	266		
		DST15	267		
		DST16	237		
		DST17	270		

Table 4-5 (Cont)
Microwords (Alphabetical Order)

Microword Mnemonic	ROM Address	Microword Mnemonic	ROM Address	Microword Mnemonic	ROM Address
LAD00	037	RSR00	106	SOB00	130
LAD01	051	RSR01	271	SOB01	342
LAD02	052	RSR02	107	SOB02	343
LAD03	033	RSR03	272	SOB03	344
		RSR04	273	SOB04	346
		RSR05	274	SOB05	345
MOV00	171	RSR06	232	SOB06	347
MOV01	172	RSR07	275		
MOV02	174	RSR08	233	SRC00	141
MOV03	173	RSR09	276	SRC01	142
MOV04	175	RSR10	277	SRC02	144
MOV05	177	RSR11	376	SRC03	240
MOV06	176			SRC04	143
MOV07	257	RST00	127	SRC05	145
MOV08	206	RST01	025	SRC06	146
MOV09	212	RST02	040	SRC07	241
MOV10	213	RST03	043	SRC08	242
MOV11	207	RST04	042	SRC09	147
MOV12	210			SRC10	243
MOV13	203	RTI00	101	SRC11	244
MOV14	202	RTI01	320	SRC12	245
MOV15	205	RTI02	321	SRC13	246
MOV16	200	RTI03	322	SRC14	247
MOV17	201			SRC15	250
MOV18	170	RTS00	124	SRC16	137
MOV19	160	RTS01	323		
MOV20	204	RTS02	324	SSL00	105
MOV21	003	RTS03	325	SSL01	372
MOV22	125			SSL02	104
		SCC00	117	SSL03	373
		SCC01	352	SSL04	221
				SSL05	236
MRK00	112			SSL06	220
MRK01	353	SER00	114	SSL07	223
MRK02	354	SER01	002	SSL08	222
MRK03	355	SER02	017	SSL09	374
MRK04	356	SER03	123	SSL10	211
MRK05	357	SER04	006	SSL11	075
		SER05	015	SSL12	253
		SER06	012		
NBR00	110	SER07	020		
		SER08	021	STA00	032
		SER09	014	STA01	076
		SER10	022		
		SER11	023	SWB00	134
				SWB01	135

Table 4-5 (Cont)
Microwords (Alphabetical Order)

Microword Mnemonic	ROM Address	Microword Mnemonic	ROM Address
SXT00	132	TRP10	326
		TRP11	327
TRP00	336	TRP12	113
TRP01	317	TRP13	330
TRP02	215	TRP14	331
TRP03	010	TRP15	077
TRP04	216	TRP16	140
TRP05	214	TRP17	337
TRP06	126	TRP18	334
TRP07	150	TRP19	335
TRP08	007	TRP20	332
TRP09	115	TRP21	333

CHAPTER 5
LOGIC DIAGRAM DESCRIPTION

5.1 INTRODUCTION

Detailed logic discussions are presented in Paragraphs 5.3 through 5.7 for each of the basic KD11-A Processor modules. These discussions correlate with the previous information on the block and flow diagrams.

The format of the discussion is ordered toward quick reference with each module and each module print identified separately. Detailed information on specific output logic signals is coupled with information on overall logic operation. The balance between these two varies as a function of the logic.

5.2 PRINT FORMAT

Certain print formats are used in the Circuit Schematics and Wire List of the KD11-A Processor, and its processor options (KE11-E, KE11-F, KT11-D, and KJ11-A). Since information is resident in these formats, they are noted in the following paragraphs.

5.2.1 Circuit Schematic Format

5.2.1.1 Logic Flow – Logic flow is from left to right with inputs on the left and outputs on the right. All inputs of a given name are interconnected on a given print unless different module pins exist. Signals which output to module pins are brought to the extreme right. Signals which do not have module pins may not be brought to the extreme right. In any case, signal names are grouped in vertical columns wherever possible. Connectors with input signals have the signal name to the right of the connector, output signals are referenced to the left of the connector.

5.2.1.2 Module Pins – Module pins are redundantly noted for each signal occurrence. If a signal occurs on several sheets of a module, the module pin appears for each entry. Module pins are presented in their backpanel context with machine slot and section noted. For example, F07D1 refers to the D1 module pin in section F of slot 07.

5.2.1.3 Print Prefixes – Print prefixes are provided for each signal to identify the print upon which the signal was generated. Since a most usual manner of logic debugging involves the tracing of signals back to their source, the print prefixes are most important. For example, K1–7 BOVFL L signal indicates a source print of K1–7, which is sheet 7 of the K1 print set for the M7231, DATA PATHS module. Print prefixes for the various modules are correlated as follows.

Module	Print Prefix	Option
M7231	K1	
M7232	K2	
M7233	K3	KD11-A Processor
M7234	K4	
M7235	K5	
M7236	KT	KT11-D Memory Management
M7237	KJ	KJ11-A Stack Limit Register
M7238	KE	KE11-E Expanded Instruction Set
M7239	KF	KE11-F Floating Instruction Set

Sheet information for each print prefix occurs as a dash and number after the print prefix. BUS print prefixes occur when multiple sources for a signal can exist; these signals are usually associated with wired-OR signal connections.

5.2.1.4 Signal Level Indicators – Signal level indicators are provided by print suffixes of H for high and L for low. These indicators attempt to relate a level with signal activation. The high and low levels in the KD11-A Processor usually correlate with TTL logic levels. For example, K3–6 WAIT L indicates that the line, so labeled, will be low when the situation WAIT exists.

Two exceptions and qualifications to this nomenclature exist. The BUS U (56:09) L signals from the ROM have a low indicator because of the wired-OR nature of the bus; in reality, the U WORD Buffer and ROM are active for high levels out. Clock signals such as K4–2 CLK IR H are active on the positive transition of the signal as they clock D-edge type flip-flops.

5.2.1.5 Flip-Flop Outputs – Flip-flop outputs are allowed two forms for a single signal output. The 1 output of a flip-flop can be represented as (1) H or (0) L with corresponding references for the 0 output of (0) H or (1) L. This nomenclature recognizes the duality of any given logic signal, but in the KD11-A Processor, it is allowed only on the flip-flops. Signals such as K3–8 CIN00L are not presented as K3–8 -CIN00 H, where the leading dash represents negation of the signal name.

5.2.1.6 Inhibit Situations – Inhibit situations are noted on the input to logic gates when the signal level indicator of the input signal does not match the state indicator on the logic gate input. This technique allows the assignment of a singular name to a logic line, with the duality of names resolved in a gate inhibit. Instead of trying to match an input state indicator with a signal level indicator and a negated name, a direct inhibit appears in the conflict between the state indicator and the singularly named signals with assigned signal level indicators.

5.2.1.7 Parentheses and Colons – Parentheses and colons are used to indicate inclusive groups of bits. A signal BUS U (56:09) L indicates the BUS U signals for bits 56 through bit 09. This grouping of bits occurs in actual signal names used on the prints; it is also used to group for discussion, signals of like nature that appear singularly on the prints.

5.2.1.8 Parentheses and Commas – Parentheses and commas are used to specify singular bits in a signal. The signal K4–3 CLR UPP (7, 6, 2) L indicates a clearing operation on bit 7, bit 6, and bit 2.

5.2.1.9 Basic and Expansion Signals – Basic and expansion signals in the machine are noted with leading Bs or Es. For example, K1–7 BOVFL STOP H is a signal generated in the basic KD11-A Processor, while KJ–2 EOVFL STOP H is a signal generated in an option or expansion of the basic processor.

5.2.1.10 Logic Symbols – Logic symbols for the KD11-A Processor include simple logic gates and flip-flops, and complicated medium and large scale integration (MSI and LSI) gates. Symbols for the latter devices tend to be rectangular with function information and grouping provided on the symbols. Truth tables are provided on the appropriate logic prints.

5.2.1.11 System Information – System information is provided on a number of logic prints in the form of tables and waveforms.

5.2.1.12 Jumper Information – Jumper information is provided on each print for option connection. Fixed formats on the etch board also provide information. Jumper numbers (W1, W2, etc.) are etched in the rest (or basic) position where two jumper positions are possible. Note on the STATUS board that W notations for the PWR UP jumpers provide the basic vector of 24.

5.2.1.13 Cable Connection – Cable connection information is provided on each print and on the etch boards. Special attention must be given to shield location as noted on the prints and on the etch.

5.2.2 Wire List Format

5.2.2.1 Alphabetical Searches – Alphabetical searches for signal names are eased by the listing of signal names without their print prefixes. The print prefix can still be determined by noting the source print in the REMARK column. The print prefix is needed in identifying the signal on the logic prints.

5.2.2.2 Print References – Print references are noted in the DRAW column for all prints on which a signal occurs. Multiple sheet entries within a print set are noted without commas between the sheet references. For example, the entry K4–235 indicates that the signal occurs on sheets 2, 3, and 5 of the K4 print set (no print sets have more than nine sheets).

5.2.2.3 Etch Backpanel – Etch backpanel information is contained in the wire list and is identified by an H in the Q column and a P in the REMARK column. EXCEPTION column notations for etch connections should be ignored.

5.2.2.4 Forward Searching – Forward searching for logic interaction (where the signals are used) is best done through the wire list. All signals for a given name are noted with appropriate print references.

5.3 M7231, DATA PATHS, K1 MODULE

The DATA PATHS module includes the following logic:

a. The Arithmetic Logic Unit (ALU) with an A input and a B input with multiplexer (B MUX) and register (B), and an output register (D).

b. The Bus Address Register (BA) with its input multiplexer (BA MUX) and output drivers to the Unibus.

c. Processor address decoding upon the internal Bus Address Register. This address decoding is not on the Unibus; therefore, these addresses only respond to processor (console or program) addressing.

d. D Register decoding for sensing when portions or all of D is zero.

e. The Scratch Pad Register (REG) with its associated addressing selection under direct microword control.

f. A console interface with drivers for data display of the D MUX signals, input receivers of the Switch Register setting on the console, and XOR matching circuit between the low order Switch Register (SR) settings and the Buffered Microprogram Pointer (BUPP) to determine when the microprogram matches the console switch settings.

This print shows the data path for the data bits 03 to 00. It has the ALU and its associated A and B input logic as well as the output D Register and Bus Address Multiplexer (BA MUX).

K1-2 D MUX (03:00) H signals at the output of the D Multiplexer provide a main data path in the machine with inputs to the B Register associated with the ALU and to each of the other processor registers including the Scratch Pad (REG) and the Processor Status (PS) Registers. These signals are also available on the back panel and are used in processor options (KE11-E, KE11-F, KT11-D).

The following inputs are combined or multiplexed into the D MUX signal: the D Register at the output of the ALU, the buffered UNIBUS BUS D signals, a right shifted output of the D Register, and the buffered BUS RD signals. The latter signals (BUS RD) are from the outputs of the various processor registers located in the DATA PATHS section of the machine and include Scratch Pad (REG), Instruction (IR), and Processor Status (PS) Registers, as well as other processor option registers. The D MUX signals are displayed on the DATA display on the console.

K1-2 COUT03 L signal is the carry out of the third bit of the ALU. It is a signal derived in the carry bridging network of the 74182. This carry bridging is used to allow a faster settling of the 74181 by looking ahead to determine if carries exist.

K1-2 D (03:00) (1) H signals output the D Register as noted, input to driver 8881 gates to the UNIBUS, and feed around to the D MUX on the input to the B side of the ALU. The D Register is essential in the data path because of the need to hold data for Unibus operations and for rewrite to the Scratch Pad Register (REG). The latch nature of the Scratch Pad Register requires a storage device in the data loop to avoid a simultaneous read and write situation in the scratch pad. The K1-2 D00 (1) H signal is also used for carry data (K3-9 C DATA H) in a Rotate Right instruction.

BUS D (03:00) L provide the Unibus BUS D signals through appropriate drivers (8881s) with gating signal K4-5 BUS FM D H.

K1-2 CLR D H signal is noted as an output only to avoid repetition of the pull-up resistor throughout the next three prints. The clear input of the D Register is essentially tied high and is not used as a signal. The D Register, as many of the other registers in the KD11-A Processor, are never cleared; they are assumed to have erroneous information until proper information is clocked into them.

K1-2 BA MUX (03:00) H signals are the inputs to the Bus Address Register (BA) located on the module (print K1–6). Multiplexed signals allow selection of either the output of the ALU or the buffered bus RD signals as the input to the Bus Address Register. The choice of these inputs is a function of microcontrol for a flow operation in process. The buffered bus RD input is provided for speed for operations in which the machine waits on bus operation, with the address coming from the Scratch Pad Register (REG). The ALU input accommodates those situations in which data is altered before use.

K1-3 D MUX (07:04) H
K1-3 D (07:04) (1) H
BUS D (07:04) L
K1-3 BA MUX (07:04) H
K1-3 COUT07 L

With the exception of bit references, these signals are similar to signals on the K1-2 print.

K1-3 D07 (1) H signal provides sign information for byte data and is used as an input to the condition codes on print K5-2.

K1-3 RD07 H signal is the highest order bit of byte data for the A input of the ALU. It is used in the status module (print K5-2) to determine overflow conditions in the ALU.

K1-3 B MUX07 H signal is the high order bit of the byte input to the B input of the ALU. It is used in the status module (print K5-2) to determine the overflow condition.

K1-3 ALU07 H signal is the direct output of the ALU prior to the B Register. It is provided for test purposes.

K1-4 D MUX (11:08) H
K1-4 D (11:08) (1) H
BUS D (11:08) L
K1-4 BA MUX (11:08) H
K1-4 COUT11 L

With the exception of bit references, these signals are similar to signals on the K1-2 print.

K1-5 D MUX (15:12) H
K1-5 D (15:12) (1) H
BUS D (15:12) L
K1-5 BA MUX (15:12) H

With the exception of bit references, these signals are similar to signals on the K1-2 print.

K1-5 D(C) (1) H signal is fed around into the D MUX on this same sheet and is used in a rotate right situation. The D(C) flip-flop is an extension of the D Register for the carry bit. The COUT MUX allows microcontrol selection of the carry output of the ALU on its inputs for word or byte, the carry bit of Processor Status PS(C), and bit 15 of the ALU output for shift or rotate.

K1-5 COUT MUX (L) signal is the selection of the aforementioned signals on the input to the D(C) flip-flop. The signal provides a test point.

K1-5 COUT15 L signal is the carry output of bit 15 of the ALU. It is used as one of the inputs for the COUT MUX and is used in the KE11-E option.

K1-5 RD15 H signal inputs to bit 15 of the arithmetic input on the A side. This signal is used in this module as an input to the D MUX and BA MUX, as are all of the buffered BUS RD signals. It is also used as an input to the condition code logic for determination of word overflow conditions (print K5-2).

K1-5 B MUX15 H provides the bit 15 input to the ALU on the B side. This intermediate signal is used on the STATUS board in the condition code logic to determine word overflow conditions (print K5-2).

K1-5 ALU15 H is the output of bit 15 of the ALU and is used as an input to the D Register, the BA MUX, and the COUT MUX.

K1-5 B15 (1) H signal is bit 15 of the B Register. It is used as an input to the B MUX on this print and the SWAP BYTE input on print K1-3. The KE11-E and KE11-F options also utilize this signal.

This print contains the Bus Address Register (BA) and the bus driving gates to the Unibus. The bus address signals for bits 16 and 17 are derived from bits 13, 14, and 15 for the basic KD11-A. The Unibus drivers, BUS A (15:00), on the print can be disabled when the KT11-D option is installed; then the above bus address bits are provided by logic on the M7236 module of the option.

K1-6 BA (17*16) H signal is a direct function of the bus address bits 15, 14, and 13 being set. This signal is used for display purposes (print K1-9) and with the KT11-D option.

BUS A (17:00) L Unibus signals provide the bus address signals from the processor and are gated by K4-5 BUS FM BA H. If the KT11-D option is installed, a jumper (W10) grounds the enabling signal of the 8881 gates for bus address bits (15:06).

K1-6 BA (15:00) (1) H signals are the direct output of the Bus Address Register. These direct outputs are used for driving the Unibus gates on this print, for decoding of processor addresses (print K1-9), for data display (print K5-7), and for generation of Unibus addresses for the KT11-D option (if installed). Some of the signals are used with the KJ11 option for comparison against the Stack Limit Register. Low order signals for bits 03 to 00 are used as one of the REG selection address inputs. Additional uses for the BA00 are odd byte address sensing, allowance of odd byte (print K4-4), and byte branching information for the BUBC codes (print K3-7).

The decoding of the Bus Address Register prior to the Unibus limits the use of these addresses to processor references. The addresses are not derived from the Unibus and it is not possible for a peripheral to access these addresses. The Bus Address Register is also decoded to determine the absoluteness of an address for stack overflow sensing. Decoding logic is also provided on the D Register to sense the status of its contents on byte or word basis. The table at the right of the sheet provides correlation between the mnemonic for an address and the octal value of the address in the bus address register. Notes are also provided for jumper selection.

K1-7 PS ADRS H decodes the processor status address to enable the combinational logic inputs to the Processor Status Register (print K5-2) and sequencing of a BUS SSYN response by the processor (print K4-6). The address is also utilized in the microbranch code to ensure a reservice of possible bus requests because of a change in machine or processor status (print K5-7). The signal is provided by the KT11-D option when installed; the jumper (W1) is removed and the option provides the processor status address.

K1-7 SLR ADRS H addresses the Stack Limit Register and is normally disabled by a jumper (W2) to ground, unless the option KJ11-A is installed. When KJ11-A is installed, the signal provides selection of that register and the sequencing of a BUS SSYN through logic on print K4-6. The signal is also wired to the KT11-D slot so that if this option is installed it can provide the address. The jumper (W2) is removed and the KT11-D provides the Stack Limit Register address.

K1-7 REG ADRS H provides the scratch pad Unibus addresses used during console operation. The jumper (W3) allows generation of the signal from this source or from the KT11-D option, if installed. The signal, when present, is utilized in the branch circuits of console operation to effect proper incrementation and access under console operation (print K3-2). Access to the Scratch Pad Register during instruction operation is through Instruction Register decode. Direct specification of registers is done by the address selection logic on print K1-8 under microprogram control.

K1-7 SR ADRS H provides Switch Register address decoding and allows derivation, either from this module or by removal of the W4 jumper from the KT11-D option. The generation of this address, as with other processor addresses, results in the sequencing of BUS SSYN through the logic shown on print K4-6.

K1-7 BA (06:03) = 0 H signal is a test point for determining that those bits of the bus address are zero.

K1-7 BA (07:05) = 1 L signal is a decoding of the segment of the bus address bit 07–05, and is used with the KJ11-A option.

K1-7 BA (15:08) = 1 L signal is a decoding of the Bus Address Register to determine content and not an address situation. This output is for test purposes.

K1-7 BOVFL STOP H signal detects a red zone stack violation and is used to interrupt the microflow (print K4-4).

K1-7 BOVFL L signal is used to sense a yellow zone stack violation and is used to set a trap servicing flag (print K5-4).

K1-7 D (15:00) = 0 H indicates that those bits of the D Register are zero. This signal is used as an input to the condition codes for the Z bit of processor status (print K5-2); it is also used as an input to the microbranch logic (print K3-2). This signal is also used in the KE11-E and KE11-F options.

K1-7 D (03:00) = 0 H signal is used for a partial indication that the byte data of bits D (07:00) is zero for the inputs to the conditional codes of processor status (prints K5-2).

This print contains the Scratch Pad Register (REG), basic to the PDP-11 architecture. There is address selection enabling either direct and complete selection by the microprogram control, or selection by the Instruction Register source field, the Instruction Register destination field, or the lower bits of the Bus Address Register. Some variations in the addressing are affected by jumpers (W5, W6, and W9) when the KT11-D option is installed. The registers themselves take data from the D MUX signals and output data onto the BUS RD lines. The resistor terminator for this wired-OR BUS RD bus are shown on this print.

BUS RD (15:00) L common input bus in the data paths has several sources. Its input is shown on prints K1-2 through K1-5, with sources from the Scratch Pad Register (REG) (this print), from the processor status system (PS) (print K5-2), and from the KE11-E and KE11-F options, as well as the KT11-D options.

K1-8 R (X6+X7) H signal senses selection of either register 6, 7, 16, or 17 (octal addresses) and is used to force an increment of 2 on any byte operations referencing register 6 or 7 in the instruction set (print K3-8). It is also used to enable the check overflow logic when the Processor Stack Register is accessed (print K4-4). This last use requires the K1-8 RADRS0 L signal.

K1-8 RADRS (03:00) L. These are the actual signals applied to the Scratch Pad Register ICs. They are the complement of the address applied to the selection AND/NOR gates. The signal K1-8 RADRS 0 L is used in conjunction with K1-8 R (X6+X7) H to specifically indicate the Stack Pointer (REG 06) to enable the check overflow logic in the bus timing circuitry (print K4-4).

This print shows the connector interface to the KY11-D console. The DATA display is provided with type 380 gates buffering the D MUX (15:00) signals. In addition, the Switch Register signals from the console are brought in and enabled by the K1-4 BUS FM SR H signal through type 8881 gates to the Unibus BUS D (15:00). A matching circuit is provided which compares the lower order Switch Register settings [SR (08:00)] with the basic microprogram pointer (BUPP) signals. Upon a match, pulse K1-9 P MATCH L is generated. This pulse occurs at the beginning of the microword specified in the Switch Register. The signal K1-9 UPP MATCH H is used with the maintenance console for a HALT on match. (See restrictions below.)

K1-9 SR (17:16) H signals for the bits 17 and 16 Switch Register settings come from the console through this module and are provided to the KT11-D option, which allows a physical address involving these address bits (print KT-9) during console operation.

BUS D (15:00) L Unibus signals are enabled by K4-4 BUS FM SR H to allow the Switch Register settings onto the Unibus.

K1-9 UPP MATCH H signal provides a level output when the Switch Register settings bits (08:00) match the buffered UPP signals. This signal is used on the timing board in conjunction with the KM11-A Maintenance Console option to halt the machine at the specified microaddress. The limitation is that the matching address must have a CL2 or CL3 preceding it.

K1-9 P MATCH L is a timing pulse which occurs at the end of a machine cycle, gated against the aforementioned match signal to provide a scope triggering pulse at the beginning of the selected word. This occurs independently of the maintenance module and is of value in situations where the machine will not be halted. For both of these instructions, the maintenance console section should be referenced for specifics of operation.

5.4 M7232, U WORD, K2 MODULE

The M7232 module for the U WORD (U is used in place of the Greek letter Mu for micro) provides the central portion of the microcontrol. It contains the basic processor's read-only memory (ROM), the U WORD Register (various mnemonics per function), the Past Microprogram Pointer Register (PUPP), and certain driving buffering gates for signals BUPP (08:00) and EUPP (08:00). Connectors at the back module edge interconnect to the KE11-E option for expansion of the basic microword ROM.

The U WORD logic on the M7232 module is very regular. The ROM has 256 words, each of which has 56 bits. Output signals from the ROM, BUS U (56:00), feed a resistor terminator with a wired-OR input from the module connectors located on the rear of the module. These inputs are for optional expansion of the ROM beyond 256 words. The BUS U signals feed the U WORD Register that controls the machine. The first segment of the ROM (07:00), on prints K2-2 and K2-3, is concerned with the microaddress and has 74H10 gates between the ROM output and the U WORD Register [UPP (08:00)]. These gates allow the next base address, BUS U (08:00), to be modified, if necessary, by basic microbranching logic [K3-2 BUBC (04:00)] or expanded microbranching [KE-4 EUBC (04:01)]. Additional logic exists on the output of the UPP portion of the U WORD for driving expansion ROMs and for storage of the present microaddress, PUPP (08:00).

The use of the BUS U (56:00) L signals and the low state indicator on the ROM gate output indicate the physical wired-OR nature of these signals. No absolute correlation should be made between low active and the state of the ROM output or the U WORD Register. For U (56:09), a high level indicates a true or active signal, both at the ROM output and in the U WORD Register. This is presented in the microflow diagrams on sheets 9 through 12. For U (08:00), the microaddress portion of the microword, a low output at the ROM output represents an active or true signal. This complementing of the ROM pattern occurs because of the inversion in the 74H10 gate prior to the U WORD Register, UPP (08:00). In the U WORD Register, a high level represents an active or true signal. The flow diagrams on sheets 9 through 12 show the complement of the ROM output for the UPF field; this is done to allow the address reference from the flow diagrams without the need to complement. Note that the UPF field represents the complemented ROM output of the next address without reference to microbranch inputs. If there are not microbranch inputs, the UPF field represents the U WORD Register, UPP (08:00).

The output signals noted in detail for this module are primarily those of the U WORD Register. These signals, with the alterations possible to the UPF field, are directly compatible to the BASIC U WORD noted in the block diagram of the U WORD and the tables shown on engineering drawing D-BD-RD11-A-BD. This drawing also provides numerous tables of the microprogram control fields, noting the codes, the effect, and occasionally the purpose.

The signals on prints K2-4 through K2-8 are presented in order from bottom to top. This is in order of ascending BUS U allocation, and represents a logical presentation of the functions.

K2-2 BUPP (03:00) H signals are the Buffered Microprogram Pointer signals for the noted bits. They are used to address expansion ROMs in the KT11-D, KE11-E, and KE11-F options. They are also displayed on the KM11-A Maintenance Console and are matched against the Switch Register for a HALT or for scope timing pulses (print K1-2). Note that the microword address present here is the address of the next microword. Alterations for microbranching will have already occurred prior to the input to the UPP Register, if they were to occur. Only a microjam (JAM CLK on print K4-3) can alter this address by setting or clearing the UPP Register. This change is then reflected in the signals.

K2-2 PUPP (03:00) (1) H signals are the output of the Past Microprogram Pointer Register. It provides the microprogram address of the microword presently in the U WORD Register. The PUPP Register is displayed in the KM11-A Maintenance Console option for the basic machine. This register is necessary to record the present microword address since the microword in the U WORD Register contains only the next address. As this word is changed [K4-2 CLK (UPP*PUPP) H], the next address (now the present address) is transferred to the PUPP Register. Most searches in the microflow diagrams (prints 9 through 12 of the M7232) for detailed operation will use the PUPP address as the starting point.

K2-2 CLR PUPP L signal is used only to hold the CLR input of the PUPP Register high. It is labeled for reference on print K2-3 to eliminate repetition of the pull-up resistors.

K2-3 BUPP (08:04) H and K2-3 PUPP (08:04) (1) H are similar to the corresponding signals on the K2-2 print with the exception of bit references.

K2-4 CLR U (16:09) signal represents a pull-up resistor to the clear lines noted for the U WORD Register.

K2-4 RIFO (0) H is used on the address input to the Scratch Pad Register for a special situation in which the KT11-D Memory Management option provides the user Stack Pointer instead of the usual REG 06 Stack Pointer. The RIF notation is discussed in the following paragraph.

K2-4 RIF (03:00) (1) H are the Register Immediate Field signals which allow direct microprogram addressing of the 16 Scratch Pad Registers (REG) when the Select Register Immediate microcontrol is enabled. Direct microword selection of a Scratch Pad Register occurs at several points in the microflow. It is used during FETCH and in the immediate address mode to explicitly select the program counter for incrementation. Trap sequences RTI and RTS instructions directly address the Stack Pointer and Program Counter registers. REG 0 is selected during the execution of the HALT instruction and during console operations.

K2-4 R125 PULL UP H signal is an identification of the resistor pull-up noted for use on print K2-3.

K2-4 SRI (1) H is the Select Register Immediate signal which is used in the Scratch Pad Register selection logic (K1-8). It enables the register immediate field provided by the microword to directly select a Scratch Pad Register (REG).

K2-4 SRBA (1) H signal is the Select Register Bus Address signal which enables the lower four bits of the Bus Address Register to select a Scratch Pad Register. This selection is used in the EXAMINE and DEPOSIT microflow for console operation. It is specifically used when an internal Scratch Pad Register address is accessed. The nomenclature used in the flow diagrams is R(BA), which indicates an internal register addressed by the Bus Address Register.

K2-4 SRD (1) H is the Select Register Destination signal which is used in the scratch pad address logic to enable the destination field of the Instruction Register IR (02:00). This field is used for various instructions that have destination addresses.

K2-4 SRS (1) H is the Select Register Source signal which is used in the Scratch Pad Address logic to enable the source field of the Instruction Register IR (08:06). This field is used in binary instructions.

NOTE

The use of discrete 74H74 flip-flops for the U WORD Buffer for REG addressing controls reflects the emphasis on reducing access time for data.

K2-5 UBF (04:00) (1) H are the Microbranch Field signals which enable various test conditions for branching the microprogram address. The actual switching of various conditions is done in the multiplexers shown on print K3-2. A total of 32 possible microbranch tests are enabled throughout the microflows and are directly noted in the flow diagrams by the BUT notation (Branch Microprogram Test) and in the table on print D-BD-KD11-A-BD. The enabled tests may consist of a number of bits or a single bit and may relate to the Instruction Register or to a single flag flip-flop.

The UBF field is also decoded on the STATUS board (print K5-3) to provide enabling signals during the microwords in which this field of a specific BUT is present. The decoded signals are used to clear or set flags relating to the tests upon which a Branch Microprogram Test is being performed.

K2-5 SBAM (1) H is the Select Bus Address Multiplexer signal and is used on the data path to control the Bus Address Multiplexer (prints K1-2, 3, 4, 5). It selects either the buffered BUS RD data or the output of the ALU for input to the BA Register. A high level in this microcode control bit enables the ALU output to the Bus Address Register data input.

K2-5 SDM (01:00) (1) H are Select D Multiplexer signals and are used in the data path (K1-2 through K1-5) to select the D MUX signal from four possible sources. The output code, or enabling levels, provided by this field can be directly associated with the 74153 multiplexer logic symbol and truth table located on the DATA PATHS prints. Essentially, a 00 SDM code selects the A input (BUS RD); a 01 SDM code selects the C input (D Register); a 10 code selects the buffer Unibus data (BUS D); and a 11 SDM code selects the D input (right shifted D Register).

K2-5 SBML (01:00) (1) H are Select B Multiplexer Low microcontrol signals which provide selection signals to the lower eight bits of the B MUX in the DATA PATHS shown on prints K1-2 and K1-3. The separation of the B MUX into an upper and lower portion for microcontrol allows additional flexibility in handling byte, sign extend, or swap byte situations. The code enables the following to the B input of the ALU:

a. An SBML code of 00 selects the low byte of the B Register directly.

b. An SBML code of 01 selects the low byte of the B Register directly and is used for sign extension in the upper byte [B MUX (15:08)].

c. An SBML code of 10 selects the upper byte of the B Register for a swap byte instruction. For example, bit 8 of the B Register is put in the bit 0 position of the ALU; this is true in sequence for the other bits.

d. An SBML code of 11 selects the B constants [BC (07:00)] as inputs to the ALU.

K2-5 SBMH (01:00) (1) H are the Select B Multiplexer High signals which provide selection signals to the upper byte of the B MUX in the DATA PATHS on print K1-4, 5. The code enables the following to the B input of the arithmetic logic.

a. An SBMH code of 00 selects the B Register directly for the B input.

b. An SBMH code of 01 selects the sign extension bit (B07) for the B input.

c. An SBMH code of 10 selects the lower byte of the B Register for a swap byte situation.

d. An SBMH code of 11 selects the B constants. These constants are not discrete for each bit input on the higher byte, but instead consist of a discrete input for bit 11 and a composite input BC (15:12, 10:08) H for the other inputs.

K2-6 SBC (03:00) (1) H are Select B Constant signals which provide selection through combinational logic (print K5-5) of a possible 16 constants for use by the B Multiplexer. The encoding of constants selection is utilized to conserve microcontrol storage (ROM) bits. A table of the constants is provided on print K5-5 and the block diagram print D-BD-KD11-A-BD.

K2-6 SALU (03:00) (1) H are the Select Arithmetic Logic Unit signals which provide direct microcode selection of the functions that the ALU will perform. These signals are selected by logic on print K3-8 for direct ALU control unless a DAD microcode of 11XX is present. (See the DAD table on print D-BD-KD11-A-BD. The ALU table on the same print also notes the Instruction Register and ALU interaction.)

K2-6 SALUM (1) H is the Select Arithmetic Logic Unit Mode signal which is used in the same way as the SALU codes previously mentioned. It is used directly on the logic shown on print K3-8 and comprises a DAD code which allows IR selection of the ALU function.

K2-6 SPS (02:00) (1) H are the Select Processor Status signals which provide an encoded combination for various functions on the processor status word. These operations on the processor status word are not unlike the encoding of the microword for the discrete alteration of data (DAD) codes. A table of SPS codes and functions is noted on the block diagram print D-BD-KD11-A-BD. Specific bits perform certain functions, and there is also a total decoding of these bits to perform other functions. The code is used in the logic shown on print K5-2 to directly select the inputs to the Processor Status Register, in the logic shown on print K5-2 to directly select the inputs to the Processor Status Register, and in the logic shown on print K3-9 to effect the condition code inputs.

K2-7 DAD (03:00) (1) H are the Discrete Alteration of Data signals which provide an encoded portion of the microword used throughout the machine to allow exceptions. It is used with Unibus cycles to check for odd address or stack overflow, and in the ALU logic to allow alteration of the code as a function of the Instruction Register. The DAD code is also used within the console loop for setting and clearing EXAM and DEP flags on consecutive operations. A summary of usage is provided in the DAD table on the block diagram, U WORD, and tables, print D-BD-KD11-A-BD.

K2-7 BGBUS (1) H is the Begin Bus signal which forms a clock bus signal with a P1 or P2 pulse (print K4-4). This clock bus signal, in turn, is used to clock the initiating signals shown on print K4-4 to begin a bus cycle, and also to load registers with various error and stack conditions which should be checked on each operation. Signal BGBUS, shown on print K4-5, is used to clock the NPR signals.

K2-7 C (01:00) BUS (1) H signals consist of the C1 BUS and C0 BUS signals usual to the Unibus. These control signals are clocked into holding flip-flops shown on print K4-4 for use throughout the bus cycle.

K2-7 CLKBA (1) H is the Clock Bus Address signal and it provides an enabling signal for the pulses (print K4-2) used to clock the Bus Address Register.

K2-7 CLKD (1) H is the Clock D signal which provides an enabling signal for the pulses (print K4-2) used to clock the D Register.

CLKB (1) H is the Clock B signal which provides an enabling signal for the pulses (print K4-2) used to clock the B Register.

K2-7 WRL (1) H is the Write Low signal which provides a signal to enable a Write signal (print K4-2) to the Scratch Pad Register for the lower byte.

K2-7 WRH (1) H is the Write High signal which provides a signal to enable a Write signal (print K4-2) to the Scratch Pad Register for the upper byte.

K2-8 CLKIR (1) H is the Clock IR signal which provides an enabling signal for the pulses (print K4-2) used to clock the Instruction Register. It is enabled only during the FETCH cycle.

K2-8 CLKOFF (1) H is the Clock OFF signal which is used in the logic shown on print K4-2 to provide direct microprogram control of clock continuance. When this bit is enabled, the Clock IDLE flip-flop is clocked on while the Clock RUN flip-flop is clocked off. The CLKOFF microcontrol stops the processor directly after the microword in which it appears. The processor then waits for an external asynchronous start signal on the set input of the RUN flip-flop.

K2-8 CLKL (01:00) (1) H are the Clock Length Code signals which provide selection of the cycle lengths used in the current microword. This signal (print K4-2) controls the pulse stream within the delay line chains. A CLKL code of 00 or 01 causes a Clock Length 1 (CL1) signal. If the CLKL code 00 is used, a special overlap situation may be in effect. A CLKL code of 10 effects a Clock Length 2 (CL2), a CLKL code of 11 effects a Clock Length 3 (CL3). The normal duration of these respective cycles are:

CL1	140 ns
CL2	200 ns
CL3	300 ns

K2-8 CLKL (01:00) (0) H signals are the complement of the clock length code and are used to ensure direct and rapid gating of the basic clock signals within the delay line chains.

CONNECTORS – On this sheet, the interconnection to the KE11-E and KE11-F options are shown. The BUS signals noted as inputs (signals to the right of the connector) are wire-ORed throughout the module to the basic ROM output. EUPP (07:00) output signals (to the left of the connector) provide the address signals for the expansion ROM.

Sheet 9 (ADR000-077)
Sheet 10 (ADR100-177)
Sheet 11 (ADR200-277)
Sheet 12 (ADR300-377)

The U WORD Microprogram Listing presents the read-only memory (ROM) content of the M7232 module and of the KD11-A Processor. The format is as follows:

Octal notation is used throughout the listing for word addresses and the contents of the individual microprogram fields.

Addresses of the U WORD are presented downward in octal numerical sequence under the ADR (address) column. The addresses correspond to those noted in the flow diagrams (D-FD-KD11-A-FD). Each address presents the complete microword for that address in the same horizontal line.

Functions of the U WORD are listed across the top of each table. These functions represent individual bits in the U WORD and are presented in fields. The fields are associated with the individual U WORD bits in the block diagram, U WORD, and tables print D-BD-KD11-A-BD.

The mnemonics for the U WORD fields (right to left) are as follows:

UPF U(08:00) — Microprogram Pointer Field represents the next microword base address in the present ROM word. This field is complemented at the output of the ROM. The field is uncomplemented in the U WORD Buffer Registers (UPP 08:00) but may have microbranch alterations already made to the ROM output. At this point, the address becomes that of the next ROM word and is used to address the ROM. The transfer of this address to the PUPP Register when the next ROM word enters the U WORD Buffer Register facilitates comparison of the U WORD and the Microprogram Listing. In the single clock mode of the maintenance console option (KM11-A), the PUPP address can be used in the ADR column to find the current controlling microword. The Microprogram Listing can also be correlated with the flow diagram from its microword address.

NOTE

With the exception of the UPF field, the function and states of the other fields are directly (uncomplemented) represented at the output of the ROM and in the U WORD Buffer. Details of operation have already been presented in the logic discussion on the U WORD Buffer signal outputs.

RIF U(12:09) — Register Immediate Field selects a scratch pad address when enabled by the Select Register Immediate bit (U13) of the SRX field.

SRX U(16:13) — Select Register provides an address mode for Scratch Pad Register selection where X can be Select Register Immediate (SRI), Select Register Bus Address (SRBA), Select Register Destination (SRD), or Select Register Source (SRS).

UBF U(21:18) — Microbranch Field enables the logic which can alter the UPF microword address to allow a branching of the microprogram flow. The U WORD Buffer for this field is UBF (04:00). A correlation is made between the Branch Microprogram Test (BUT) number and its purpose on print D-BD-KD11-A-BD.

SBA U22	Select Bus Address directly controls the Bus Address Multiplexer on the input to the Bus Address Register. When enabled, the U WORD Buffer signal, SBAM, selects the ALU output instead of the BUS RD signal output.
SDM U(24:23)	Select D Multiplexer directly controls the selection of inputs on the D Multiplexer (D MUX). Its octal codes 0, 1, 2, and 3 correlate respectively to the A, B, C, or D inputs in the logic symbol.
SBM U(28:25)	Select B Multiplexer directly controls the selection of the inputs on the upper and lower byte sections of the B Multiplexer (B MUX).
SBC U(32:29)	Select B Constants controls the logic which generates B constants, which are then selected by the B Multiplexer. The code, the constants, and the purpose of the codes are presented in the SBC table on print D-BD-KD11-A-BD.
ALU U(37:33)	Arithmetic Logic Unit controls the mode of operation of the ALU in the data paths. The code is not used directly and allows discrete alteration as a function of the Instruction Register. This interaction is shown in the ALU table of the block diagram, U WORD, and tables print D-DA-KD11-A-BD.
SPS U(44:41)	Select Processor Status provides a discrete and encoded microcontrol of the input and clocking of the processor status word. This control is especially concerned with the individual response by the condition code portion to each instruction.
DAD U(44:41)	Discrete Alteration of Data is a microcode field that provides for the alteration of usual usages of data (including microcode data). A usual alteration is the checking for odd addresses or stack limit violations during bus operations initiated by microprogram data. A table of functions and codes is noted on the block diagram, U WORD and tables print D-BD-KD11-A-BD.
BUS U(47:45)	BUS operations for the Unibus are controlled by this microcontrol field. Included are the bus control signals, C1 BUS and C0 BUS, and their initiating signal BGBUS. A table of bus operations (including the non-data transfers) is shown on print D-BD-KD11-A-BD.
CBA U48	Clock Bus Address field provides the direct enabling signal for clocking the Bus Address Register.
CD U49	Clock D field provides the direct enabling signal for clocking the D Register.
CB U50	Clock B field provides the direct enabling signal for clocking the B Register.
WR U(52:51)	Write field provides two directly used microword bits for writing into upper or lower byte of the Scratch Pad Register.
CIR U55	Clock IR field provides the direct enabling signal for clocking the Instruction Register.
CLK	Clock field contains both the clock cycle length control (CLKL 0, CLKL 1) and on-off control (CLKOFF).

The three other columns contained in the Microprogram Listing are:

ADR — The microprogram address of the microword displayed on that line. This address can be obtained from the flow diagram or from direct observation of the PUPP Register with the KM11-A Maintenance Console option.

STATE — The mnemonic used in the flow diagram (D-FD-KD11-A-FD) to provide an immediate identification of a microword. It is possible to refer to a microword in easier terms than its address.

FLOWS — The page in the flow diagram (D-FD-KD11-A-FD) upon which the microword occurs. This reference provides for a backward search from an address to a microword in its flow context.

5.5 M7233, IR DECODE, K3 MODULE

The IR DECODE module contains extensive combinational logic which decodes the Instruction Register (IR), providing discrete instruction signals, as well as re-encoded microaddress information necessary for the microbranches. The Instruction Register is present on this module, as is the BUT Multiplexer. In addition, combinational logic exists for instruction control of the ALU and condition code inputs for the carry (C) and overflow (V) bits of the Processor Status Register.

This print shows the Branch Microprogram Test Multiplexer which combines diverse microbranch tests into a limited number of bits for a next microprogram address. There are essentially six multiplexers, two of which affect bit 0 of the address; the other four multiplexers affect bit 1 through bit 4 of the address. The conditions gated to the address are occasionally singular and named by the actual signal condition, such as JSR. The conditions are often complex and affect more than one address bit; they are named in the standard way, such as K3-5, BUBC0 (BUT37) H. This signal is essentially a basic microbranch code that will affect the 0 bit of the microaddress for the BUT37. A table of these branch microtests and their mnemonics appears on the block diagram print D-BD-KD11-A-BD. BUT37 is the INSTR1 branch occurring in FETCH; it branches to all the various response microflows for instruction implementation. It has an input to each of the five affected address bits. Other BUTs only require one or two bits, and therefore only input into one or two address bits. For this reason, the type of multiplexer related to specific address bits changes. On bit 0 there are two multiplexers which input into two possible inputs in the NOR/OR gate. For the next address bit there is a single 16-input multiplexer. For the next two address bits, there are 8-input multiplexers and the upper two address bits have only 4-input multiplexers.

K3-2 BUT (37:34) L is a decoding of the microbranch field (UBF) for BUTS 37 through 34. It is a single pin run and is provided for test purposes.

K3-2 BUT (3X) signal is a decoding of the microbranch field (UBF). The signal is used to enable the multiplexers on this sheet and on the STATUS module (prints K5-3 and K5-6) as an enabling signal for clocking flag flip-flops. The signal is a partial decoding of branch microprogram test for BUT 30 through 37 octal.

K3-2 BUBC (05:01) L signals represent the basic microbranch code for address bits 5 through 1. They each represent a single input to the NOR/OR gate where they can modify a base address when a branch test is called. These bits provide the inputs for all branch tests, unlike the input for the 0 address bit which required two distinct inputs for lower order and higher order BUTs. Selection of these inputs is a function of the microbranch field from the U WORD applied against the appropriate multiplexers. In conjunction with the basic microbranch code, there are expansion microbranch code bits also input to the NOR/OR gate.

K3-2 BUBC0 (BUT37:20) L provides basic microbranch code 0 for BUT 37 through 20 (the notation is octal). It is used in conjunction with the next signal to the exclusion of the expansion microbranch code for this bit. It is used on K2-2 print in the NOR/OR gate.

K3-2 BUBC0 (BUT17:00) L provides the basic microbranch code for bit 0 for the BUT 17 through 00. The signal is selected as a function of the multiplexer and the UBF field in the U WORD, with the UBF field selecting the BUT being applied against the base address. Thus, bit 0 has many test conditions applied against it, not only in the complex codes but the single bit codes.

This print shows the complete Instruction Register (IR), which has input data from D MUX (15:00). All of the IR is brought to the module edge for expansion and basic machine use. In addition to the IR, the first level of decoding is provided by the 8251 Decoders. The binary instructions, the source mode, the destination modes, and intermediate IR bit patterns are decoded.

K3-3 IR (15:00) (1) H is the (1) side of the IR brought out for use within the basic and expansion machine. It is used on various inputs in the IR DECODE itself, on other prints within the basic machine; it is also used in the KE11-E option, the KE11-F option, and the KT11-D option. The low order bits, in the case of the Source or Destination Registers, are used in the register selection logic associated with the Scratch Pad Register.

K3-3 IR (14:12)=0 L is a partial decoding of the IR for bits 14 through 12 equal to 0 and is utilized on the STATUS module for branch instruction decoding and enabling.

K3-3 SM=1 L K3-3 SM=2 L K3-3 SM=3 L	These signals are partial decoding of the IR (bits 11 through 09) for the source mode equal to the respective number. They are used on the STATUS board (K5-3) for branch instruction decoding and enabling.

K3-3 SM=0 L is a partial decoding of that portion of the IR (bits 11 through 09) for source mode equal to 0. It is used throughout the IR module, on the STATUS board (K5-3) for branch instruction decoding, and on the KT11-D option (print KT-9).

K3-3 SM=7 L is a partial decoding of the IR for source modes equal to 7 (bits 11 through 09). This is a single pin entry and is a test point.

K3-3 IR (08:06)=6 L is a partial decoding of the IR, indicating that the octal code for bits 8 through 6 inclusive is 6. It is used in the KT11-D option.

K3-3 IR (08:06)=0 L is a partial decoding of the IR Register, indicating that bits 8 through 6 are 0. It is used in the KE11-F option.

K3-3 DM=0 L is a partial decoding of the IR for a destination mode 0. It is used in the KT11-D option.

K3-3 CLR IR L signal is a pull-up resistor signal for the clear input of the IR.

This print contains additional decoding of the IR with a relatively fast and direct decoding of the single operand instructions. In addition, the low order bits IR (02:00) are decoded. Combinational logic is provided for the overlap signals with the signals consisting of an overlap cycle and an overlap instruction.

K3-4 IR (02:00)=6 L is a partial decode of the IR Register bit 2 through 0 inclusive, equal to 6 octal. It is utilized by the KT11-D option.

K3-4 OVLAP CYCLE L includes the next signal overlap instruction, as well as additional situations. An overlap cycle is based on the same premise as an overlap instruction; i.e., the next bus address desired in FETCH is the incremented PC.

In certain instructions, time can be saved by beginning the address calculation which uses the incremented PC (this is true in index operations), and, in this case, it is done for destination modes 6 or 7 on single operand instructions of JMP and JSR. It is also done for destination mode 6 or 7 if the source mode of a double operand instruction is 0; it is done for a source mode 6 or 7. Here the exceptions for service between instructions do not prohibit the overlap cycle; the overlap cycles pertaining to internal instruction operations occur. The signal is used on TIMING (print K4-4) to initiate another bus cycle during FETCH.

K3-4 OVLAP INSTR H signal for overlap instructions is active for certain instructions with certain address modes. It is also necessary that specific service requirements and some instruction modes do not exist. Overlap is a situation where, in the FETCH of a given instruction as the PC is being incremented, it is possible to initiate a bus cycle using the incremented PC. This can only be done when it is known that the next bus address desired is a DATI to the incremented PC. If this is true, the cycle can begin while the processor is still busy with the present instruction. The situations where overlap instructions occur are usually single operands with destination mode 0, or double operands with both source and destination modes 0. Exceptions to this are that the Destination Register cannot be REG 07, nor can the Program Counter that is being used as the next address be in the process of change. Other exceptions to overlap involve service requirements for bus requests, power fail, console bus requests (HALT switch), and the TRACE bit in the STATUS word. MOVE instructions for byte operations are not overlapped. This signal is used on STATUS (print K5-4) as a data input to the OVLAP flag. The flag ensures proper re-entry into the FETCH microflow.

K3-4 IR15 H and K3-4 IR15 L are buffered signals provided for the additional drive requirements of this particular bit of the IR.

This signal is basic microbranch code for instruction 1. The print contains combinational logic which further decodes the initial IRD decoding, provided on the previous pages, into specific instruction signals. In addition, some of the instruction signals from this sheet and instructions from subsequent sheets are re-encoded into basic microbranch code (BUBC) for the first instruction branch. This instruction branch is known as INSTR1 for BUT37 and appears on sheet 1 of the flow diagram (D-FD-KD11-A-FD).

K3-5 BUBC (05:00) (BUT37) H is the basic microbranch code for microaddress bits 5 through 0 and is activated on the INSTR1 branch test for BUT37. It is decoded from the IR and is available on the input to the multiplexer. The multiplexer itself on print K3-2 provides the selection for BUT37 and this code is enabled over the base microaddress for this test. This branching code is especially critical and basic to the machine since it is the first instruction branch in FETCH.

K3-5 DOP * -SM0 L is a double operand instruction and not source mode zero encoded together and provided for use within the IR board.

Combinational logic shown on this print further decodes the initial decoding of print K2-3 and provides discrete signals for certain instructions. These instructions are the non-double operand and non-single operand instruction which often require a flag set or a unique function performed. These signals are at the right and at the interior of the print.

Most of the instruction signals noted are mutually exclusive and are active (H) or low (L) as noted. Some signals of interest are noted below.

K3-6 PRIV INSTR L signal provides the KT11-D option with information on privileged instructions (HALT and RESET) to make their implementation in USER mode appear as NO-OPs. Note the inhibiting of the discrete HALT and WAIT signals by the KT02 PS15 (0) H signal.

K3-6 I1K0 (CINSTR) L is an internal intermediate signal for instruction 1 constant for bit 0 for C instructions. It is used as an element of the BUBC0 (BUT37) signal for bit 0 on print K3-5. Like other elements of the BUBC signal, it represents a microaddress re-encoding from the decoded Instruction Register.

Located on this print are various basic microbranch codes (BUBC) for tests other than INSTR1, consisting of different numbers of address bit inputs for different BUTs. The ones shown on the extreme right are as important as the ones shown on the left or center. Essentially, BUBC codes for BUTs 20, 21, 25, 26, 27, 31, 33, 34, 35, and 36 are shown. There are also some additional Instruction Register signals, such as SERVICE, TRACE, and BYTE CODES. The majority of signals (BUBC codes) are used on print K3-2 as inputs to the multiplexers. A table of the BUTs used is on the block diagram, U WORD, and table print D-BD-KD11-A-BD, as well as in Table 4-1.

K3-7 TRACE L signal provides for an immediate trace trap during service if PS(T) is set and the IR does not contain an RTT instruction. The signal is used on this print in the BUBC1 (BUT26) signal and on STATUS prints K5-4, 5 for flag control and trap vector generation.

K3-7 SERVICE H is a definition of the reasons to enter the service section of the microflows after instruction execution. It contains flags and inputs for internal (bus error, basic overflow on the stack, power down, and trace) and external (BUS Request Priority flag, console bus request, and reference to processor status address) situations requiring service. The signal is used on this print in the BUBC1 (BUT20) signal for microbranching and is provided as a test point.

K3-7 BYTE CODES H signal indicates to the condition codes logic (print K5-2) that a byte instruction is in the IR. The signal is used on STATUS (print K5-2) for selection of input data to the condition codes of the Processor Status Register.

K3-7 BUBC (5,03:00) (BUT36) H is the basic microbranch code for microaddress bits 5, 3, and 0 for the BUT36. BUT36 is the INSTR3 branch associated with the next flow sequences after SOURCE calculations.

K3-7 BUBC (5,03:00) (BUT35) H is the basic microbranch code for microaddress bit 5 and bits 3 through 0 for BUT35. BUT35 is the odd byte and INSTR3 branch associated with byte formatting of incoming data or the next flow sequences after SOURCE calculations.

K3-7 BUBC (03:00) (BUT34) H is the basic microbranch code for microaddress bits 3 through 0 for BUT34. BUT34 is the INSTR4 branch associated with the next flow sequences after destination calculations.

K3-7 BUBC (03:00) (BUT33) H is the basic microbranch code for microaddress bits 3 through 0 for BUT33. BUT33 is the odd byte and INSTR4 branch associated with byte formatting of incoming data or the next flow sequences after destination calculations.

K3-7 ODD BYTE L is the combination of a BYTE instruction decode from the IR and a 1 in bit 00 of the Bus Address Register. This signal is used within the IR DECODE module in the microbranching logic of BUBC (BUT33).

K3-7 BUBC (01:00) (BUT20) H is the basic microbranch code for microaddress bits 1 and 0 for BUT20. BUT20 is the Byte, Service, or Fetch branch associated with the end of instruction execution.

K3-7 BUBC0 (BUT31) H is the basic microbranch code for microaddress bit 0 for BUT31. BUT31 is the No Write or Byte Write or Word Write associated with instructions of destination mode zero requiring register rewrite.

K3-7 BUBC0 (BUT27) H is the basic microbranch code for microaddress bit 0 for BUT27. BUT27 is the Service B or Fetch Overlap or Fetch B branch associated with the end of instruction execution where an overlap situation might exist.

K3-7 BUBC (01:00) (BUT26) H is the basic microbranch code for microaddress bits 1 and 0 for BUT26. BUT26 is the Request branch associated with the entry into the SERVICE flow and provides for the proper sequencing and servicing of requests.

K3-7 BUBC (01:00) (BUT25) H is the basic microbranch code for microaddress bits 1 and 0 for BUT25. BUT25 is the Bus Request or Wait or Fetch branch associated with the servicing of these requests in the WAIT loop of SERVICE.

K3-7 BUBC (01:00) (BUT21) H is the basic microbranch code for microaddress bits 1 and 0 for BUT21. BUT21 is the IR03 and Byte or Source branch associated with index address operations in the MOV address calculations.

This print shows two sets of combinational logic. One set is ordered toward the ALU control signals and provides for a multiplexer selection of either the U WORD directly, or control as a function of IR decode. Multiplexer selection is a function of the DAD code. The other set of logic is the carry-in for the ALU and control of the Carry-Out Multiplexer. See Table 3-2 for ALU functions.

K3-8 COMUXS (01:00) H provide the inputs of the COUT Multiplexer Selection (print K1-5) which forms the data input of the D(C) flip-flop. Selection is solely a function of the IR decode and inputs from the KE11-E option; no direct control from the U WORD exists.

K3-8 CIN00 L provides the carry-in for bit 00 of the ALU (print K1-2). Control of this data input is a function of the IR decode and indirect control from the U WORD through the Discrete Alteration of Data (DAD) and Select Arithmetic Logic Unit (SALU).

K3-8 BIT+CMP+TST H is a simple combination of the bit test, compare, and test instruction from the IR decode. It is used with the IR DECODE module and TIMING (print K4-4) to alter DATIP bus cycles to DATI bus cycles for destination data references.

K3-8 ALUS (03:00) H are the direct control for the ALU selection signals on prints K1-2, 3, 4, and 5. The multiplexer selects either direct U WORD control by the SALU signals, or Instruction Register control by either the basic processor or KE11-E option. Multiplexer selection is controlled by the Discrete Alteration of Data (DAD) signals of the U WORD.

K3-8 ALUM H is the direct control of the ALU mode on prints K1-2, 3, 4, and 5. Combinational logic allows U WORD control by the DAD microfield or IR decode.

K3-8 DAD (3*2) L is a decoding of discrete bits in the DAD microfield. It is used in the KE11-E and KE11-F options.

This print shows combinational logic associated with the input data required for the C and V bits of the condition codes. Conditioning of these data inputs is a function of the IR decode and the present processor status.

K3-9 V DATA L is the V DATA input of the overflow bit of the condition code portion of the processor status word. This input reflects direct loading inputs (D MUX01) as well as instruction data inputs: V(ROTSHF), V(COMPARE1), and V(COMPARE2). The signal is used on print K5-2 of STATUS.

K3-9 C DATA H is the C DATA input of the C or carry bit of the condition code portion of the processor status word. This input reflects direct loading inputs (D MUX00), as well as instruction data inputs. The signal is used on print K5-2 of STATUS.

5.6 M7234, TIMING, K4 MODULE

Timing for the KD11-A Processor consists of the basic processor clock for data path and microcontrol, and the Unibus-ordered control for data and bus ownership transfers. Microcontrol techniques are used in each section, but discrete flip-flop, combinational logic, discrete timing (delay or pulse) circuits are necessary. These circuits and logics are discussed in context with the overall timing and not ordered upon output signals.

This print contains the basic processor clock which consists of the CLK flip-flop, pulse-width forming delay line logic, and cycle-length forming delay line logic. Necessary peripheral logic provides on/off control (IDLE flip-flop), asynchronous restart inputs, and output enabling gates.

Assuming sequential, uninterrupted operation, the end of one clock cycle is the beginning of the next clock cycle. The trailing edge of the K4-2 RECLK H signal clocks a 1 to the CLK flip-flop (assuming continuous operation) which activates the pulse forming logic loop with Delay Line 1 (DL1). After the delay, the DL1 loop clears the CLK flip-flop. The CLK flip-flop, therefore, forms a pulse of approximately 40 ns (DL1 time plus gate time). This pulse is now passed through additional delay lines to form the various cycle lengths (CL1, CL2 and CL3).

A CL1 is formed by passing the CLK pulse through Delay Line 2 (adjustable per CLOCK ADJUSTMENT note) to 74H00 gates at E63 (output pins 08 and 11). If a CL1 was specified by the U WORD CLK field, the signal K2-8 CLKL1 (0) H enables the CLK pulse through the upper 74H00 gate (E63, output pin 08) where, after inversion (74H00 gate at E66, output pin 06), it becomes K4-2 P1 H.

A CL2 is formed by the CLK pulse if, after passing through Delay Line 2, the bottom 74H00 gate (E63, output pin 11) is enabled by K2-8 CLKL1 (1) H signal. The upper 74H00 gate (E63, output pin 08) is disabled. The CLK pulse now passes through Delay Line 3 to the 74H00 gates at E72 (output pins 08 and 11). Here a P2 pulse is generated with the upper 74H00 gate (output pin 08) allowing the pulse as an End of Cycle signal to the microcontrol and clock.

If a CL3 is to be formed, the bottom 74H00 gate (E72, output pin 11) enables the P2 pulse to the data path and to the next delay line (DL4). The upper 74H00 gate (E72, output pin 08) is not enabled to allow the P2 pulse as an End of Cycle signal. That signal is provided by the P3 pulse from the 74H00 gate at E72 (output pin 03).

Reference to the CLK WAVEFORMS table allows correlation between the clock output pulses, their relative timing, and the U WORD enabling signals.

The output enabling gates service the three sections of the KD11-A Processor: the INTERFACE, the DATA PATHS, and the MICROCONTROL. The microcontrol clocking signals (CLK U signals, RECLK P END, and PART P END) are ordered toward end-of-cycle pulses. For a CL1, this is a P1 pulse; for a CL2, a P2 pulse; and for a CL3, a P3 pulse. Clocking to the U WORD and the clock logic is not conditioned by any enabling signal and is usually on the final pulse transition. The End of Cycle signals are also used in the flag control logic of STATUS, especially P END and PART P END. Here the signals may be used as set or clear pulses with enabling BUT signals.

The output enabling gates for data path and interface control use a variety of the P1, P2, and P3 pulses. The pulses are enabled singularly or in combination by specific U WORD control bits to provide the several CLK signals noted. The pulse signals are also provided directly for generation of other CLK signals in the basic (STATUS) and expansion (KE11-E, KE11-F, KT11-D) processor. Note that any end (enabled) CLK signal must have only one gate (H series) between the pulse signals (P1, P2, P3) and the end CLK signal; this prevents excessive clock skew.

Continuance of clock cycles, one after another, is determined by the End of Cycle signal, K4-2 RECLK H, and the data input signal to the IDLE flip-flop. If a new clock cycle (microword, machine state) is to begin, the IDLE flip-flop data input is inactive (a high logic level); the CLK flip-flop data input is therefore the inverse (74H00 gate at E73, output pin 03), and the CLK flip-flop is clocked to the 1 state. This begins the pulse forming and delay sequences already noted. If a next clock cycle is not to begin, the IDLE flip-flop data input is active (a low logic level) and the flip-flop is clocked to the 1 state; the CLK flip-flop is not clocked to the 1 state and no pulse forming occurs. Conditions to halt the clock are noted on the inputs to the 74H53 gate at E77; the most common input would be the U WORD control signal K2-8 CLKOFF (1) H. Note that the U WORD is clocked by the last pulse transition of the halting clock cycle, the machine halts in the beginning of the next microword and awaits timing signals.

The restarting of the clock is effected by the combination logic on the set input of the CLK flip-flop. This input has interlocking signals from the CLK pulse forming logic and IDLE flip-flop to ensure that the clock restarts without partial pulses and that the clock is completely off before restart. The actual restart inputs provide for a fast direct restart for data transfer situations (K4-6 B SSYN H input) and a combination of lower priority (time wise) restarts. Common to each of these restart inputs are the enabling conditions for the restart condition and the restart signal.

An additional control flip-flop, MCLK, is provided for single clock manual operation. This flip-flop functions in parallel with the CLK flip-flop and generates the beginning transition to the pulse forming logic. It does this as a function of maintenance console switch activation (KM-2 MCLK L). The IDLE flip-flop is not directly affected by this manual operation mode. The CLK flip-flop is affectively disabled with neither its data or set inputs enabled. Details of maintenance console interaction are contained in Paragraph 7.3 of this manual.

Discontinuities exist in the microprogram flow. The majority of these interruptions are accommodated by halting and restarting the CLK logic (noted for print K4-2); the next microword after the halting signal [usually K2-8 CLKOFF (1) H] is entered and the machine awaits a restart signal. An interruption (or pause) has occurred in the microprogram flow, but sequential flow still occurs after restart.

The CLK JAM logic is ordered toward non-sequential interruptions of the microprogram flow. Error conditions or power-up sequences enable this timing such that the usual microcontrol timing is disabled (K4-3 JAMUPP H signal on IDLE flip-flop input) and special clocking signals are provided to force the microprogram flow to specific microaddresses. The microprogram flow is irrevocably JAMmed to a specific operating flow. The JAMUPP ADDRESSES table on this print correlates the reasons (USE) for the microprogram jam and the new microaddresses (UPP).

The CLK JAM logic has three parts: error sensing and power-up flag flip-flops, asynchronous serial timing logic, and combination logic for the new microprogram address generation.

The flag flip-flops which are clocked to the 1 state for activation are the JBERR flag for odd address bus errors and red zone stack overflow, and the JPUP flag for START switch activation in the HALT mode and Power Restart. The PERJ RS type flip-flop is set when a memory parity error is detected. These three flags, with additional inputs from the NODAT flag (print K4-6) for non-existent bus address error and PWRUP INIT (print K5-8), activate the timing logic.

The JAMUPP one-shot, when activated, provides an enabling signal to the combination logic generating the new microaddress. This logic encodes the various error and power up flags to provide direct set and clear signals to the Microprogram Pointer (UPP) Register. Usual machine timing is disabled (IDLE data input of print K4-2); less important machine flags (TRAP and INTR of print K5-4) are cleared; and the BERR flag and STALL flag are clocked (print K5-4). Deactivation of the JAMUPP one-shot removes the set and clear signals to the UPP Register, and after a delay (δ = 100 ns), provides the K4-3 JAM CLK H signal. This signal clocks the newly selected microword (see JAMUPP ADDRESSES table) into the U WORD Buffer and activates the JAM START one-shot. The pulse output of the JAM START one-shot clears the NODAT flag (print K4-6) if appropriate, and restarts the CLK logic.

The PERR flip-flop stores the fact that a memory parity error has occurred and is used to generate the parity trap vector.

The logic shown on this print is associated with processor Unibus data transfers and the variety of required error checking and cycle alterations. Some decoding of the Unibus BUS C signals is provided for processor and processor option use. The logic consists of control flip-flops (BUS, CKOVF, CKODA, BWAIT, BC1, and BC0) which are activated by U WORD and IR decode input. Delays for skew correction are provided between the bus activating control flip-flop (BUS) and the actual MSYN flip-flops. Appropriate checking logic combines error conditions with error check enabling signals. Bus cycles are aborted or allowed with error conditions affecting the flag flip-flops of STATUS (print K5-4). Tables are provided for the BUS and DAD fields of the microword.

BUS, MSYN, MSYNA Flip-Flop – The BUS flip-flop initiates all processor Unibus cycles. It is clocked to the 1 state by K4-2 CLK BUS H signal (derived from BGBUS of the U WORD), except for the DAD code (1X1X) in the execution flow of the BIT or CMP or TST instruction, and the non-existence of an overlap cycle in the FETCH flow (BUT37 at FET04 microword). The activation of the flip-flop is gated by bus ownership signals in the 74H20 gate (E9, output pin 06). For a bus cycle to occur, the processor must be in charge of the bus [K4-5 BBSY (1) H], no Unibus cycles are in process (K4-6 B SSYN L), and the processor is not giving up bus ownership (K4-5 PROC RELEASE L). With these conditions met, the delays associated with Unibus data skew and address decode are activated. Two delays exist: one for normal Unibus delay to the MSYN flip-flop, and a shorter delay to the MSYNA flip-flop.

During the deskewing delay, error conditions disable the data inputs of the MSYN and MSYNA flip-flops. Normally, however the flip-flops are clocked to the 1 state and drive the Unibus through appropriate gates. Disabling exists for the KT11-D option. The MSYN, MSYNA and BUS flip-flops are pulsed clear from the BWAIT flip-flop.

CKOVF Flip-Flop – The CKOVF flip-flop controls the check of overflow on the processor Stack Pointer. Only certain address modes in certain bus operations need to be checked. This is controlled by the DAD code (X11X) of the U WORD with register selection information; a disabling flag [K5-4 STALL (1) L] from STATUS can inhibit the check. The CKOVF flip-flop is clocked by the K4-2 CLK BA H signal with activation of the flip-flop further conditioned by the KT11-D option and the Unibus cycle type. The check enabling signal enables the error detection signals and provides for possible abortion of the Unibus cycle (inhibits data input to MSYN flip-flop) with corresponding raising of error flags. This occurs here only for red zone stack overflow; the yellow zone stack overflow is handled solely by the error flags.

CKODA Flip-Flop – The CKODA flip-flop controls the check of odd address errors on processor data bus cycles. The flip-flop is always clocked to the 1 state by the K4-2 CLK BA H signal unless a byte instruction exists with a DAD code (XXX1) from the U WORD. Checking, however, is further conditioned by console operation and the KT11-D option. The check enabling signal enables the error detection signals [K1-7 BA00 (1) H or KT-3 FAULT H] and provides for possible abortion of the Unibus cycle (inhibits data input to MSYN flip-flops) with corresponding raising of error flags.

BWAIT Flip-Flop – The BWAIT flip-flop provides the clearing signal (K4-4 P CLR MSYN L) for the processor BUS, MSYN, and MSYNA flip-flops. The flip-flop is set by activation of the IDLE flip-flop (print K4-2); this is usual for processor data bus cycles. The BWAIT flip-flop remains set during BWAIT for the usual peripheral response (K4-6 B SSYN L), which restarts the CLK. Usual deactivation of BUS, MSYN, and MSYNA flip-flops occurs at the end of the first microword [K4-2 (P1 + P3) H] when the BWAIT flip-flop is clocked to the 0 state. Other clearing signals are combined in the pulse logic to accommodate situations where no peripheral response is made (NODAT error, microcontrol JAMUPP for other bus errors) and processor initializing.

BC1 and BC0 Flip-Flops – The Unibus control signals are held in the BC1 and BC0 flip-flops. The flip-flops are loaded from C1BUS and C0BUS bits of the U WORD by the K4-2 CLK BUS H signal (derived from BBUS of the U WORD). Modification of the data input for BC0 is made for byte instructions (to change DATO operation to DATOB) and for BIT or CMP or TST instructions (to change DATIP operation to DATI). Appropriate gates drive the Unibus with additional logic providing conditioning inputs to processor and processor options (KJ11-A especially) which respond through the processor to absolute bus addresses.

Shown on this print are the discrete flip-flops and combinational logic associated with the granting and acceptance of Unibus ownership by the KD11-A Processor. Processor ownership exists with the BBSY flag in the 1 state, and is necessary on power-up, console operation, processor data bus cycles, RESET instruction, power fail, and prior to release of bus ownership for bus requests. The processor usually controls the bus unless it has specifically given up control.

The granting of bus ownership requires that peripheral requests for ownership are acquired by the processor in the appropriate flag flip-flops: the NPR flag for a non-processor request; the BRPTR flag for bus requests with a priority request greater than processor status priority; and the CBR flag for the KY11-D Console HALT switch. Clocking signals combining various inputs are necessary with proper sequencing of Unibus bus ownership signals [BUS SACK L, BUS NPG H, and the BUS BG (07:04) H] on the Unibus (*PDP-11 Peripherals and Interfacing Handbook*).

The major clocks for priority determination and acquisitions of requests are K4-5 CLK NPR H and K4-5 CLK PTRD H. Both clocks contain clocking signals with a BUS MSYN clock necessary for situations when the processor is inactive; no separate continuous clocking exists in the processor for the priority determination logic.

The K4-5 CLK NPR H signal, in addition to the BUS MSYN clock, has clock inputs for clock restart (K4-2 SET CLK L), data bus cycle beginnings [K2-7 BG BUS (1) H], BUT26 in service flow, the activation of MSYN (K4-5 P MSYN H pulse), and CLK IR for overlap situations. Independent of clocking, the data input to the NPR flag is inhibited for power fail (K5-8 B AC LO L), during console operations, and across DATIP/DATO operations. A DATIP flag flip-flop prevents the granting of bus control for non-processor requests across DATIP/DATO cycles as the DATIP address location is still selected by the processor with the probability of a partial read/restore cycle in the peripheral.

BGBUS (1) H produces a pulse which strobes the NPR line each time the processor is ready to initiate a bus data cycle. If any NPR is present, it will be serviced prior to the execution of the processor data cycle.

B MSYN L and SET CLK L clock the NPR flag during and upon restart following a processor bus data cycle. NPRs clocked in by either of these signals will be serviced immediately following the bus cycle.

BUT26 in the SERVICE flow provides clocking of the NPR flag while the WAIT instruction is being executed. ENPRCLK provides an external clock from the KE11-E option. Both of these signals clear the BBSY flag directly so that all NPRs are serviced immediately.

The BBSY flag is clocked each time the processor initiates (K4-2 CLK BUS H) and terminates (K4-4 P CLR MSYN L) a bus data cycle. It is cleared, relinquishing bus control, whenever an NPR is clocked into the NPR flag.

Clocking for the K4-5 CLK PTRD H occurs for BUS MSYN clock and for BUT26 in the SERVICE flow and CLK IR for overlap situations. Associated with this clock is the PTRD one-shot that delays the actual clocking of the BRPTR flag flip-flop until the comparison of peripheral bus request priority levels can be made against processor status priority levels (print K4-6). The result of that comparison is signal K4-6 BRQ H on the data input of the BRPTR flag. The BRPTR flag is used to store the fact that a bus request of higher priority than the present processor priority has been received. It is used as a source of information for branching to SERVICE at the end of an instruction and also for branching to the interrupt service microflow.

Console control of the processor via the HALT switch is gained in a manner similar to the servicing of a bus request. The PTRD one-shot clocks the CBR flag which stores the condition of the HALT switch. If the HALT switch is activated, the processor will branch to SERVICE at the end of the current instruction, at which time BUT26 will cause the machine to enter the CONSOLE microflow.

Three types of bus response are provided by the logic shown on this print: the Bus Grant signals in response to Bus Requests; the SSYN and Bus Address selection of processor registers in response to processor or console bus cycles; and the processor timeout flags for NO SACK and NODAT.

The Bus Grant signals [BUS BG (07:04) H] are generated by comparison logic for the incoming Bus Request signals and the existing Processor Status signals. The results of the comparison are used in the bus ownership logic shown on print K4-5 to determine if the BRPTR flag should be enabled. With the flag enabled, processor service of the flag results in the K4-5 GRANT BR H signal activating one of the BUS BG (07:04) H signals on the Unibus.

Processor register response to absolute bus addresses is not completely specified by microprogram control. Bus address decoding (print K1-7) and Unibus control decoding (print K4-4) are combined to read or write these registers. Timing signals are provided for Unibus response (BUS SSYN L) and clocking of the registers [K4-6 PS (P FM BUS) H]. Note that a read from a processor register usually results in data gated onto the Unibus; a Write to a processor register results in the data being available on the D MUX signals. The Scratch Pad Register (REG) does not respond with SSYN to processor and console Bus Address references; rather, it responds to console references directly, and then under microprogram control.

The processor register addresses that respond with SSYN are PS ADRS (Processor Status Register), SR ADRS (Switch Register), and if the KJ11-A option is present, the SLR ADRS (Stack Limit Register). Normally, the Load Address function addresses the Switch Register for the address data, however, in the case of a BEGIN operation, the Switch Register data gating enable (K4-6 BUS FM SR H) is inhibited. BEGIN is a remote Load Address/Start function, and the address data must be provided by the remote device. A further discussion of the BEGIN function is contained in the logic description of the K5 print.

The timeout flags for no SACK and no data provide a processor response when peripherals fail to respond. The NOSACK flag is set when peripherals granted bus ownership fail to respond; the NODAT flag is set when data bus operation receives no SSYN response. In each case, the timeout duration is 15 μs. The service of the timeout flags differs. The NODAT flag results in microprogram interruption (JAMUPP) and a trap sequence. The NOSACK flag merely allows the processor to regain bus ownership and continue operation. Each timeout may be disabled for maintenance operation. See the note on the print or details of maintenance module operation in Paragraph 7.3 of this manual.

5.7 M7235, STATUS, K5 MODULE

The STATUS module contains miscellaneous combinational logic relating to processor status. This includes:

a. Processor status word with priority bits for comparison to bus request, a TRACE bit, and condition codes N, Z, V, and C.

b. Branch instruction implementation with comparison of the condition codes with IR decoding.

c. Branch Microprogram Test (BUT) decoding with discrete outputs as a function of specific microwords.

d. Flag flip-flops for a variety of machine and error states that require unique servicing.

e. B constants decoding with Special Trap Pointer Marker (STPM) signals for trap vectors.

f. Console flags for START, BEGIN, and proper incrementation on double EXAMs and DEPs.

g. Console interface for the ADDRESS display and control inputs.

h. Power fail and bus delay one-shots for proper sequence of some Unibus signals (BUS AC LO L, BUS DC LO L, BUS INIT L), as well as processor start-up.

The processor status word consists of PS (07:00), with PS (07:05) associated with the priority of machine operation. It is this portion of processor status that is compared against the bus request signals to determine whether a bus request should be granted. These processor status bits are represented by discrete flip-flops and are loaded from the D MUX signals upon a specific LOAD processor status clock. Other bits of the processor status are the PS(T) bit and the condition codes. PS(T) is the TRACE bit and its function is described in detail in the appropriate *Processor Handbook.* Loading of the TRACE bit does not occur as a function of a processor reference to an absolute bus address. The TRACE bit is implicitly altered only in RTI and RTT instructions and in trap sequences.

The condition codes portion of the processor status word consists of bits PS(N), PS(Z), PS(V), and PS(C). These bits are loaded from the D MUX upon a specific LOAD processor status clock from the processor, in addition to conditional inputs as a function of instruction operations and data results from those operations. The conditional inputs for PS(C) and PS(V) are already generated upon the IR DECODE module (print K3-9). The inputs for PS(Z) and PS(N) are generated by the combinational logic on this module. The major conditions of all these inputs are indicated in the Processor Handbook for each instruction.

Other logic on the K5-2 print is the PASTA and PASTC flip-flops necessary for holding PASTA input (to the ALU) and PASTC [PS(C)] information for condition code operations. A multiplexer is used for the selection of input data, usually high byte or low byte for the PASTA flip-flop and other condition code logic [PS(N) and PASTB]. Combinational logic is utilized in the generation of the processor status clocking signal with direct interaction occurring between the clock pulses [K4-2 PS(P1+P3) H], U WORD control [K2-6 SPS (02:00) (1) H], address decoding (K1-7 PS ADRS H), and instruction decoding (K3-6 CC INSTR H).

K5-2 PS (07:05) (1) H are the priority bits of the Processor Status Register and are compared against the bus request signals on the TIMING module (print K4-6). These flip-flops are loaded from the D MUX (07:05) lines when the processor status word is referenced by the processor or console with its absolute bus address.

BUS RD (07:00) L are the signals connecting the Processor Status Register to the internal Processor Register Data Bus. These signals allow the routing of the processor status word through the machine in trap sequences and condition code instructions.

BUS D (07:00) L are the Unibus signals that allow the Processor Status to respond to processor or console requests to its absolute bus address.

K5-2 PS(T) (1) H is the TRACE bit of the processor status word and is used on the IR DECODE module (print K3-7) to generate a branch to SERVICE (no RTT instruction present). Signals K3-7 TRACE L and K3-7 SERVICE L reflect this input with the appropriate flag flip-flop on the STATUS module (print K5-4) being set. The PS(T) bit is not loaded with the rest of the processor status word, it is implicitly altered only on RTI and RTT instructions and during trap sequences.

K5-2 PS(N) (1) H is the negative bit of the condition codes portion of the processor status word. It is loaded as a function of absolute bus address reference to the processor status or under microprogram control in instruction or trap operations. Input data for condition code operation comes from combinations of logic which select upper or lower byte information. The signal is used in combinational logic generating the ALU control signal K3-8 ALUM H on the DATA PATHS module and in the branch instruction logic (print K5-3).

K5-2 PS(Z) (1) H is the zero bit of the condition codes portion of the processor status word. It is loaded as a function of absolute bus address reference to the processor status, or under microprogram control in instruction or trap operations. Input data for condition code operation consists simply of combinational logic to sense word or byte zeroing of the D Register. The signal is used in the branch instruction logic (print K5-3).

K5-2 PS(V) (1) H is the overflow bit of the condition codes portion of the processor status word. It is loaded as a function of absolute bus address reference to processor status, or under microprogram control in instruction or trap operation. Input data for condition code operation is provided by K3-9 V DATA L from the IR DECODE module. The PS(V) signal is used in the branch instruction logic (print K5-3).

K5-2 PS(C) (1) H is the carry bit of the condition codes portion of the processor status word. It is loaded as a function of absolute bus address reference to processor status, or under microprogram control in instruction or trap operations. Input data for condition code operation is provided by K3-9 C DATA H from the IR DECODE module. The PS(C) signal is used in the branch instruction logic (print K5-3), on the input multiplexer for D(C) Register (print K1-5), and in combinational logic for generation of signals K3-8 CIN00 L, K3-9 V DATA L, and K3-9 C DATA H.

K5-2 BUSRD FM PS H gates the processor status word to the Bus Register data lines for condition code instructions and for microcontrol Select Processor Status (SPS) codes of 6 for trap sequences and console display.

K5-2 N DATA L is the input data to PS(N) and provides byte-selected data [D15 (1) H or D07 (1) H] to the combinational logic generating K3-9 V DATA L on the IR DECODE module (print K3-9).

K5-2 LOAD PS L is the enabling signal for the combinational logic on the data inputs of the condition codes to allow the D MUX data signals instead of condition code inputs. The signal is used on this print and on the IR DECODE module (print K3-9).

K5-2 PASTA (1) H is a holding flip-flop for the most significant bit (word or byte) for the AIN input of the ALU. This signal is necessary in the calculation of overflow data (K3-9 V DATA L); storage of the input is required because the condition code calculation occurs after the AIN input is removed.

K5-2 PASTB H is a simple gating of the most significant bit (word or byte) for the BIN input of the ALU. The signal is necessary to the calculation of overflow data (K3-9 V DATA L).

K5-2 PASTC (1) L is a holding flip-flop for the past value of the PS(C) flip-flop. The signal is used in the combinational logic generating signal K3-9 V DATA L for SBC and DEC instructions, and in signal K3-9 C DATA H.

K5-2 SPS (02:00)=7 H is a decoding of the Select Processor Status (SPS) code and is used with the KT11-D option.

Two distinct sets of combinational logic are shown on this print: branch instruction logic for comparison of instruction decoding with condition codes; and BUT decoding of the microprogram field.

K5-3 TRUE BR L indicates that TRUE conditions specified by the Instruction Register for a branch instruction have been met by the condition codes. The signal, when active, provides BUBC signals (print K3-5) to alter flows and implement the instruction.

K5-3 FALSE BR L indicates that FALSE conditions specified by the Instruction Register for a branch instruction have been met by the condition codes. The signal, when active, provides BUBC signals to alter flows and implement the instruction.

K5-3 BR INSTR L is the decode of the Instruction Register for a branch instruction. It is used in the BUBC signals (K3-4 print) for the INSTR1 microbranch.

BUT signals noted for this print are decoded from the microbranch field (UBF) of the U WORD. These decoded signals are used throughout the processor as auxiliary timing signals unique to the microword in which a specific BUT is called. A table on the print correlates the numeric code of a BUT with its mnemonic function; BUTs that are decoded and used for auxilliary purposes (besides branching the microflow) are called "working BUTs." Flow diagram notations (D-FD-KD11-A-FD) indicate when and what functions these BUTs perform. A usual function is to clear and set machine flag flip-flops such as those on STATUS module prints K5-4, K5-6, and K5-8. In these instances, the BUT signal acts as an enabling signal to a timing pulse.

Flag flip-flops for error conditions and machine sequencing are shown on this print. The logic discussion will deal with the interaction and function of each flag flip-flop instead of discussing output signals.

Provided below, from top to bottom, is the sequence of service to the internal processor traps, external interrupts, and HALT and WAIT. This order of sequence is affected by the interaction of the flag flip-flops and is basic to understanding their operation.

BUS ERROR Traps – Odd Address Fatal Stack Overflow (Red) Memory Management Violations to 250 (if KT11-D) and Memory Parity Errors.

HALT Instruction – Console Operation (and certain changes if KT11-D).

TRAP Instructions – Illegal or Reserved Instructions, BPT, IOT, EMT, TRAP.

TRACE Trap – T Bit of Processor Status.

OVFL Trap – Warning (Yellow) Stack Overflow.

PWR FAIL Trap – Power Down.

BERR Flag – The Bus Error flip-flop provides a flag for trap service upon the occurrence of a NODAT or odd address error in a processor Unibus data transfer. The flip-flop is clocked to the 1 state by the activation of the data inputs from the NODAT flip-flop (on TIMING) or the Odd Address Error signal with the clocking signal K4-3 JAMUPP H (which also jams the microflow to a trap routine). The BERR flag output generates appropriate STPM constants from trap vectors and accommodates the ordered sequence of service for the various processor flags. This sequence is noted in the Processor Handbook and is repeated in the introduction to this print.

Certain clearing signals are common to the BERR, TRAP, and INTR flag flip-flops. They are: the processor Initialize (INIT) signal; the External Pulse Clear Trap signal from the KE11-E option; BUT03 in TRP16 microword at microaddress 140 in the trap sequence; and the establishment of a new stack at location 024 for a power-down situation. Common clearing signals work for BERR, TRAP, and INTR flag flip-flops because their service is mutually exclusive. A BERR flag aborts the other two, TRAP service is due to instruction operation, and INTR service occurs only after instruction operations.

In addition to the common clearing signals, the BERR flag is cleared and held clear for KY11-D Console operation. This allows the bus error of NODAT and Odd Address Error to occur without a trap sequence that would alter Processor Status, the Program Counter, or the Stack Pointer. No trap response to the bus error in console operation is considered the safe response. The JAMUPP signal does occur but the microflow is jammed to the console switch loop microflow.

Normal sequential servicing of the BERR flag results in the BUT03 clearing the flag. The BERR is first priority and prohibits the clearing of lower order priority flag flip-flops during its trap service.

TRAP Flag – The TRAP flip-flop provides a flag for trap service in proper sequence for trap instructions (BPT, IOT, EMT, and TRAP). The flip-flop is clocked to the 1 state by the data input of an IR decode of a trap instruction with the clocking signal K5-6 P BUT37 H, which occurs in the FETCH cycle. The micrologic branches to the trap sequence for service with appropriate STPM constants generated by the TRAP flag and the IR decoding.

In addition to the common clearing signals noted under the BERR flag, the JAMUPP signal directly clears the TRAP flag. TRAP flag service is aborted if a JAMUPP signal occurs. Normal sequential servicing of the TRAP flag results in the BUT03 clearing the flag with lower priority flag flip-flops unaltered.

INTR Flag – The Interrupt (INTR) flip-flop is clocked to the 1 state by the data and clock input of the K4-4 B INTR H signal (decoded from the Unibus) with the clocking signal requiring the non-existence of the INTR flag, and the K4-2 SET CLK L signal for machine restart. (If the KM11-A Maintenance Module is present, single clock mode inhibits the K4-2 SET CLK L signal and the P3 signal is used to clock. Note that the INTR bus cycle waits for the next signal clock before completion.) After the INTR flag is set, the micrologic branches to the trap sequence with the trap vector provided by the interrupting peripheral. Exactly the same clearing signals used for the TRAP flag are used for the INTR flag.

The INTR flag is used both within this module for the sequential clearing of flags and on print K5-6 for Slave Sync (SSYN) response for the INTR bus cycle. Normal sequential clearing of this flag is done by BUT03 in the trap service.

AWBY Flag – The Await Bus Busy signal is utilized by the processor in its instruction flow and defines no trap service condition. It is set for specific U WORD BUS codes (C1BUS=0, C0BUS=1, BGBUS=0) with P1 or P3 timing pulses. These codes are generated in the SERVICE flow where the processor must have absolute control of the bus prior to granting the bus requests.

The AWBY flag is cleared by a P1 or P3 pulse and the absence of the U WORD bus codes previously used to set AWBY. This occurs directly after the machine restarts. Clearing also occurs for the processor INIT signal and the operation of a new stack at location 04 at power down. This last set of clearing signals is named K5-4 FLAG CLR H and is common to other flags.

The output of the AWBY flip-flop is utilized directly on TIMING (print K4-2) to enable machine restart on processor BBSY (1) H. It is also used on print K4-5 to disable the SET CLK signal from clocking the NPR flag.

BOVFLW Flag – The Basic Overflow flag senses stack overflow error for red zone violations (K4-4 OVFLW ERR L) and for yellow zone violations (K1-7 BOVFL, or KJ-2 EOVFL if the KJ11-A option is installed). The BOVFLW flip-flop is clocked to the 1 state by the K4-4 CLK OVFLW H signal if either error is present. Once the flag is set, a feedback signal to the data input allows further clocking without zeroing the flag. The output of the BOVFLW flag generates the STPM constants for the trap vector and provides for proper trap sequencing.

A red zone stack error results in a JAMUPP signal so that the BERR, TRAP, and INTR flags are zeroed. The jam entry into the trap sequence provides for the clearing of the BOVFLW flag by BUT01 in the TRP20 microword at address 332. (Note that the T bit of new Processor Status should not be set so that the K3-7 TRACE L signal is not active.)

A yellow zone stack error results in a normal microprogram flow with the BOVFLW flag being serviced in sequence; appropriate BUBC bits for a microbranch to SERVICE are enabled on IR DECODE (print K3-3). The BOVFLW flag is still cleared in sequence by BUT01 in TRP20 microword but only if the higher priority flags (BERR, TRAP, or INTR) have been serviced. If they are not serviced, the microprogram flow recycles through the trap sequence until service is complete.

PWRDN Flag – The Power Down flip-flop is clocked by the power fail synchronizing signal K5-8 CLK PWR DN H. The flag output alters microprogram flow by enabling appropriate BUBC bits for a microbranch to SERVICE on IR DECODE (print K3-3); STPM constants for the trap vector are also generated. Normal sequential service results in the flag being cleared by BUT04 of the TRP21 microword at address 333. The higher priority flags (BERR, TRAP, INTR, and BOVFLW) must have been serviced or recycling through the trap sequence.

If a JAMUPP signal occurs when the PWRDN flag is enabled, power fail takes precedence by clearing (K5-4 FLAG CLR H) the higher priority flags and using the new stack at location 04.

STALL Flag – The STALL flag inhibits the jam stack overflow checking and provides a no trap service condition. The flip-flop is clocked to the 1 state for Double Bus Error, red zone stack overflow (K4-4 OVFLW ERR L) or PWRDN flag with the clocking signal K4-3 JAMUPP L. Feedback from itself prevents the flag from being lost on reclocking. The STALL flag directly inhibits the overflow checking logic on TIMING (print K4-4). The flag is cleared by the processor INIT signal and by BUT04 in the TRP21 microword in the trap service. No inhibits exist on the BUT04 clearing of the STALL flag since the error condition requiring a suspension of overflow checking is serviced in the first trap service.

WAIT Flag – The WAIT flip-flop is clocked to the 1 state by the IR decode of the WAIT instruction with the K5-4 P BUT37 L clocking signal during the FETCH cycle. The flag enables BUBC signals for a WAIT loop in the SERVICE segment of the microflows.

BRSV Flag – The Bus Request Service flag is set in the SERVICE flows if a bus request requires service. The actual signal is BUT26 (in SER07 microword at address 020) and the BRPTR flag is active. The flag is used to enable asynchronous restarting signals to the CLK flip-flop (TIMING, print K4-2) after the bus request; the flag is also used to inhibit the clearing of BBSY and to generate the K4-5 PART GRANT BR H signal. The flag is cleared by the same signal used for WAIT clear, with the BUT07 clearing in the bus request SERVICE flow being the most usual.

OVLAP Flag – The Overlap flag is clocked to the 1 state by the data input of K3-4 OVLAP INSTR H with the K5-4 P BUT37 L clocking signal during the FETCH cycle. Once set, the flip-flop remains set (unless K5-4 FLAG CLR H occurs) for the instruction and provides proper microbranching information (BUBC signals of print K3-7) for a FETCH OVLAP entry to the FETCH flow sequence for the next instruction. The flag also enables the IDLE flip-flop (print K4-2) on FETCH OVLAP entry and provides an additional PTR clock (print K4-5). The flag is reclocked during the next FETCH cycle and is clocked to 1 or 0, depending upon the K3-4 OVLAP INSTR H signal.

Two sets of constants are generated on this print: STPM constants for trap vectors; and the B constants used throughout the microflows. Tables note the constants and their use.

K5-5 STPM (4,3,2) H are Special Trap Markers used for trap vectors. Input signals from IR decode and flag flip-flops provide the highest priority trap vector as the output STPM constant. The STPM signals input to the B constant logic where they are enabled by a BC code of 00. The STPM constants and use are noted in the STPM table.

K5-5 SBC=10 L is a decoded signal of the Select B Constant microcode used on the KT11-D option.

K5-5 BC (15:12, 10:09) H
K5-5 BC (11, 08) H
K5-5 BC (07:00) H signals are the B constants generated and selected by the Select B Constant (SBC) microcode of the U WORD. Correlation between the SBC code, the B constants, and their use can be found in the SBC table. Of special interest are the jumpers (W2 through W7) which allow a power-up vector different from 24 to be used; the initial jumper selection, however, is for location 24.

Jumper W8 allows the signal PERR (1) L in conjunction with BERR (1) L to generate the trap vector 114 for memory parity errors.

K5-5 BCON (1+2) H is a conditional B constant output which allows a B constant of 1 to become 2 by providing a K3-8 CIN00 L signal. The signal results from the SBC=3 code and is used through the flows in address calculations where the last address incrementation may be byte or word ordered. A REG (X6+X7) input forces the incrementation to 2 for byte incrementation on PC or SP Registers.

K5-5 SBC=16 L is a decoded signal of the Select B Constant microcode used on the KT11-D option.

The logic associated with this print provides the necessary flags and BUBCs for console operation. The following logic discussion is ordered toward console operation rather than output signals. A functional description of console switch operation is presented in Chapter 3 of the *PDP-11/40 System Manual.*

Console logic consists of the flag flip-flops necessary to service the control switches with associated combinational logic to set and clear the flags. Some additional logic is necessary to generate the BUBCs utilized in the console SERVICE flow for microbranches to the individual switch SERVICE flows.

Activation of any console control switch (except HALT/ENAB) results in the SWITCH flag flip-flop being clocked to the 1 state. This flag is sensed directly through the BUT MUX of print K3-2 in the console loop by BUT06 in CON04 microword at location 026. The transitions that clock the SWITCH flag also provide the signal levels necessary for the Basic Microbranch Test [BUBC (02:00) (BUT30)] to access the individual switch flow responses. Reference to the flow diagram (D-FD-KD11-A-FD) for console operation and BUT30 shows the exclusive nature of the switch BUBC code; only one switch can be serviced. The SWITCH flag is cleared by the processor INIT signal, by BUT37 in the FETCH cycle (for START), and by BUT3X (at BUT30 when switch type is being sensed). The PART P END signal indicates a cycle end pulse for a CL2 or CL3 only.

The START and BEGIN switches require console flags. Each produces a non-filtered (contact bounce exists) INIT signal when the console switch is activated. Each clocks its flag flip-flop (and the SWITCH flag) to the 1 state as the switch is released. Both flag flip-flops provide inputs to the BUBC logic for switch sensing; the BEGIN flag is also used for microbranching to sequence a START flow sequence after a LOAD ADRS flow sequence. The flags are each cleared by the processor INIT signal by BUT37 in the FETCH flow or by BUT10 in the START flow.

The CONSL flag flip-flop is clocked to the 1 state on entry into the console loop by BUT24 in CON12 microword at address 255 and by BUT06 in CON04 microword at address 026; both are in the console flow. The CONSL flag allows single instruction operation by inhibiting the HALT signal in the BUBC (BUT26) signal (print K3-7) in the SERVICE flow. This allows the CONT switch one instruction FETCH before the HALT switch is serviced as a console bus request. The CONSL flag also inhibits usual bus error responses by disabling logic for ODA ERR (print K4-4) and altering the JAMUPP microaddress (print K4-3). Clocking for NPRs and BRs is also disabled (print K4-4). The flag is also used in the KT11-D option. The CONSL flag is clocked to the 0 state by a BUT10 in the START flow, by a BUT04 if BEGIN, and by a BUT26 in SERVICE flow.

The EXAM and DEP flags are used for essentially the same purpose. They provide automatic address incrementation for console operations which are consecutive EXAMs or DEPs. The flags are clocked to the 1 state during the latter part of their respective flow sequences: the EXAM flag is clocked by BUT04 and DAD0 (1) H; the DEP flag is clocked by BUT03 and DAD0 (1) H. The outputs are ORed together (K5-6 CONSL INC H) and used in the B constant for SBC=7. To prevent the incrementation when EXAM and DEP are directly intermixed, the EXAM flag is zeroed at input to the DEP flow and the DEP flag is zeroed at the input to the EXAM flow (BUT03 and BUT04, respectively). Both flags are cleared on entry in the CONSOLE flow (BUT24), in the SERVICE flow (BUT26), and in the START flow (BUT05).

Two connectors are shown on this print. The KY11-D connector (J1) has associated logic to drive the ADDRESS display and accommodate the console control signals utilized on print K5-6.

The other connector (J2) provides an interface for a remote Unibus console to allow remote control of the stop, initialize, and start functions at a console-provided Unibus address. A manual or automatic remote console interface may be used, however, care must be taken to avoid simultaneous use of the remote console and the programmer's console, as both consoles are logically active. Signal interconnections should be twisted pair cables, with the ground returns provided on the connector. Length is limited to twenty feet. The remote console's Unibus interconnection for the starting address is governed by appropriate Unibus rules.

The characteristics of the connector signals are noted in their order of activation.

REMOTE STOP L input signal is logically ORed with the KY11-D HALT switch to halt the machine after instruction completion. This signal should be asserted for a minimum of 100 ms.

K5-7 CONSOLE H output signal indicates that the processor is in console mode, awaiting switch activation. This signal should be present before either REMOTE INIT L or REMOTE START L are activated. The K5-7 CONSOLE H signal should become active sometime during the REMOTE STOP L signal activation. It is cleared during REMOTE INIT L activation and is reasserted under control of the microprogram approximately 450 ns after the deactivation of REMOTE INIT L. It is deactivated again under control of the microprogram approximately 40 ms after activation of REMOTE LOAD L.

REMOTE INIT L signal input clears both the Unibus and the processor. This signal should be asserted for a minimum of 100 ms and only if the processor has responded to the REMOTE STOP L signal with K5-7 CONSOLE H. Note that direct enabling of REMOTE INIT L by K5-7 CONSOLE H is not possible due to variations in the K5-7 CONSOLE H signal.

REMOTE LOAD L input signal clocks the BEGIN flip-flop to the 1 state and activates the microprogram BEGIN sequence. This sequence consists of a LOAD ADRS and a START. The REMOTE LOAD L signal should be a minimum 2 μs pulse with a Unibus address provided at activation and maintained as noted below; actual microprogram action begins at the trailing edge of the pulse. The signal should be asserted when the K5-7 CONSOLE H is reasserted after the REMOTE INIT.

BUS D (15:00) L Unibus signals are used to provide the starting address for the BEGIN sequence. Electrical characteristics of these signals are the same as for any other device on the Unibus data lines. The address should be gated upon the Unibus at the leading edge of the REMOTE LOAD L signal and can be removed with the next deactivation of K5-7 CONSOLE H.

Delay circuits associated with Unibus and processor operation are shown on this print. Several delays sequence the BUS AC LO L and BUS DC LO L signals of the Unibus for power fail operation. Another two delays provide a RESET instruction Unibus INIT signal and a RESET RESTART signal. Start-up delays for processor operation are provided by the PWRUP INIT and POWER RESTART one-shots.

K5-8 CLK PWRDN H is the Clock Power Down clock signal to the PWRDN flag flip-flop on print K5-4. Necessary to this signal is the synchronizing LOWAC flip-flop; this flip-flop, with its associated gating, ensures that no power fail indication (the activation of BUS AC LO L) is missed and that none provides more than one clocking signal. Sensing of power failure occurs immediately unless the DELAY POWER DOWN delay is still active after the power-up situation. Some of the other power fail delays interact (AC LO delay) but these are ordered primarily toward the proper sequencing of BUS AC LO L and BUS DC LO L signals on the Unibus. Typical waveforms are shown in the table USUAL POWER FAIL WAVEFORMS. Examples of the power-down and power-up sequences are presented in detail in Chapter 3.

K5-8 PWR RESTART H signal initiates a JAMUPP to begin microprogram sequences (print K4-3) approximately 70 ms after the deactivation of BUS AC LO L. The PWR flip-flop associated with the POWER RESTART delay prevents the one-shot from firing unless a power-up situation exists; variations in BUS AC LO L for power-down are ignored.

K5-8 P ENDRESET L signal is an asynchronous pulse restart signal to the CLK flip-flop (print K4-2) for the RESET instruction. This restart signal occurs approximately 70 ms after the halt in the RESET flow at RST01 microword at address 025 containing a BUT02.

K5-8 RESET RESTART L signal indicates the status of the 70 ms RESET RESTART one-shot.

K5-8 INIT + RESET H signal provides a test point for the signal producing the BUS INIT signal.

BUS INIT L is the Unibus INIT signal consisting of a RESET initialize and the processor initialize (INIT 1). The signal is used by Unibus peripherals.

K5-8 INIT 1 L
K5-8 INIT 2 L
K5-8 INIT H are signals for the processor to initialize itself and the system. The signal consists of START and BEGIN switch initialize, direct BUS DC LO L initialize, and a PWRUP INIT one-shot initialize which becomes active at the deactivation of BUS DC LO L. The signal is used by the processor control flip-flops and all Unibus peripherals.

K5-8 PWRUP INIT L signal is approximately 20 ms and occurs on the deactivation of BUS DC LO L. The signal initiates a JAMUPP in the microcontrol (print K4-3) to location 377, which contains all 0s.

K5-8 B DC LO H is an identification signal for the buffered BUS DC LO L signal.

K5-8 D DC LO L is the buffered BUS DC LO L signal and is used to directly set the IDLE flip-flop on print K4-2.

K5-8 B AC LO L is the buffered BUS AC LO L signal used as a data input to the JPUP flip-flop and as an inhibit to the clocking of NPRs.

BUS DC LO L is the Unibus signal indicating low dc voltages. See table of USUAL POWER FAIL WAVEFORMS.

BUS AC LO L is the Unibus signal indicating low ac voltages. See table of USUAL POWER FAIL WAVEFORMS.

CHAPTER 6
KY11-D PROGRAMMER'S CONSOLE

6.1 KY11-D CONSOLE

The KY11-D Programmer's Console consists of the KY11-D Console Board (5409701) and two cables (BC08R-03) which are used to interconnect the console to the KD11-A Processor. Both power and logic signals are provided by these cables that connect to the DATA PATHS (M7231) board and the STATUS (M7235) board. Operating instructions for the console are included in the *PDP-11/40 System Manual.*

A remote Unibus console interface also exists in the KD11-A Processor. This interface is described in Chapter 5 in relation to print K5-7.

6.2 KY11-D CONSOLE BOARD

The KY11-D Console Board shown on print D-CD-5409701-0-1 consists of displays with data and control switch inputs.

6.2.1 Print KYD-2, Display

The display on the console consists simply of light-emitting diodes (LEDs) with current limiting resistors; the drivers for these displays are located on the DATA PATHS and STATUS boards of the KD11-A Processor. Input signals from the processor are shown at the left of the displays; actual console panel notation for the displays is shown in parentheses near the diode symbol.

Connectors (J1, J2) for processor interconnection are also shown on this print. These connectors provide for the display signals from the processor, as well as the Switch Register data and control signals to the processor.

6.2.2 Print KYD-3, Switches

The data switch inputs from the Switch Register are shown at the right. Simple resistor inputs are used. The console functions are shown in parentheses [(SR09) for example] with the connector signals at the right.

The control switches have Set-Reset flip-flops to eliminate contact bounce, in addition to a driving gate. The console functions are noted in parentheses; the connector signals are at the right.

An additional switch for OFF/POWER/PANEL LOCK is also shown. Its connectors (J3, J4) consist of two quick disconnect tabs to allow direct interconnection to the cabinet power control unit.

6.3 CABLES

The BC08R-03 cables are interconnected to the KY11-D console (J1, J2) and the M7231 and M7235 modules according to the instructions on the printed circuit boards and the circuit schematics. Orientation of the shield is specified and required for proper interconnection. Connection for power control to J3 and J4 is simple: this connector provides only a switch closure and, therefore, either interconnection of the two wires is acceptable.

CHAPTER 7
PROCESSOR OPTIONS

7.1 SCOPE

This chapter provides a complete description of three of the internal processor options that may be used with the KD11-A Processor. These options are:

a.	KJ11-A Stack Limit Register	In the basic machine, a fixed boundary is provided to prevent stacks from expanding into locations containing other information. The Stack Limit Register provides a programmable boundary with both warning (yellow) and fatal (red) stack error indications.
b.	KM11-A Maintenance Console	This option provides indicators and switches for manually operating the system and for monitoring the status of key signals during maintenance procedures.
c.	KW11-L Line Frequency Clock	This option references real intervals and generates a repetitive interrupt request to the processor. The rate of interrupt is derived from the ac line frequency.

Processor options differ from bus options in two respects: they are physically mounted within the processor, and they interact with the processor without necessarily using the Unibus. For example, for many processor options, jumpers are often added or removed from the processor modules so that the option is logically connected directly to the processor.

Other processor options are available for use with the KD11-A. Because of their size and relative complexity, they are covered in other manuals. The KE11-E Extended Instruction Set and KE11-F Floating Instruction Set are both covered in the *KE11-E and KE11-F Instruction Set Options Manual.* The KT11-D Memory Management option is covered in the *KT11-D Memory Management Option Manual.*

7.2 KJ11-A STACK LIMIT REGISTER

The KD11-A Processor is capable of performing hardware stack operations. Because the number of locations occupied by a stack is unpredictable, some form of protection must be provided to prevent the stack from expanding into locations containing other information. In the basic machine, the protection is provided by a fixed boundary; the KJ11-A Stack Limit Register provides a programmable boundary.

The KJ11-A consists of a single addressable register, accessible to both the console and the processor, that is used to change the stack limit and to provide warning (yellow zone violation) and error (red zone violation) indications for the stack. The Stack Limit Register is an 8-bit register (high-order byte) that can be addressed either as a high-order byte (777775) or as a full word (777774).

During operation, the register is loaded with an address signifying the lower limit of the stack (stack violations occur at or below this limit). During subsequent stack pointer related operations of certain bus cycle types (DATO, DATOB, and DATIP), if the address of the bus operation is less than the contents of the Stack Limit Register, an error condition exists.

If the difference is less than or equal to 16 words, a yellow zone violation occurs. The operations that caused the yellow zone violation are completed and then a bus error trap occurs. This error trap, which itself uses the stack, executes without causing an additional violation.

If the space between the bus address and the Stack Limit Register is greater than 16 words, a red zone violation occurs and the operation causing the error is aborted. The stack is repositioned and a bus error trap occurs; i.e., the old PS and PC are pushed into locations 2 and 0 and the new PC and PS are taken from locations 4 and 6. A red zone violation is a fatal stack error. Note that these two stack error conditions exist in the basic KD11-A Processor; however, in this case the stack limit is fixed at memory location 400_8. Other fatal stack errors are odd stack or non-existent stack.

The KJ11-A Stack Limit Register Option is a single-height module that plugs into slot E03 of the processor. It requires the movement or removal of the following jumpers on KD11-A Processor modules.

Module	Print	Jumper	New Position
M7231	K1-7	W2	Connect W2 between module pin E04H2 and pin 06 of E63.
M7234	K4-4	W1	Connect W1 between module pin B07F2 and pin 10 of E16.
M7235	K5-4	W1	Connect W1 between module pin D06R2 and pin 01 of E51.

7.2.1 Functional Description

The Stack Limit Register logically determines if a particular address is within valid limits or if it is in the yellow (warning) or red (error) zone of the stack. The logic first compares the high-order byte of the address with the value in the Stack Limit Register. If the high-order byte is greater than the Stack Limit Register value, then the address is valid and not infringing on the stack. If, however, the high-order byte of the address and the contents of the Stack Limit Register are equal, then the address is not valid and the logic must determine which type of violation (yellow or red) has occurred. The logic then examines bits (07:05) of the low-order byte of the address to determine if the violation is a yellow zone or red zone violation. If the high-order byte of the address is less than the Stack Limit Register value, a red zone violation has occurred.

The comparison of the high-order byte of the address and the contents of the Stack Limit Register is shown in Table 7-1.

For the situation where the upper bytes of the Stack Limit Register and the bus address are equal, it is necessary only to monitor the value of bits (07:05) to determine if a red or yellow zone violation has occurred. If all three of these bits are set, then the value of the low-order byte must be somewhere in the range of 340 to 377 (20 octal or 16 decimal word locations) which is a yellow zone violation. If any one of the bits is not set, then the highest possible address would be 337, which is the upper limit of the red zone.

Table 7-2 summarizes the method of monitoring the low-order byte to determine whether a red or yellow zone violation is present.

Table 7-1
Comparison of Address and SLR

Bit Position (High Byte)	15	14	13	12	11	10	09	08	Octal Value
	VALID ADDRESS (GREATER THAN)								
Bus Address	0	1	1	0	1	1	0	0	0660
SLR Contents	0	1	1	0	1	0	1	0	0650
	INVALID ADDRESS (EQUAL)								
Bus Address	0	1	1	0	1	0	1	0	0650
SLR Contents	0	1	1	0	1	0	1	0	0650
	INVALID ADDRESS (LESS THAN)								
Bus Address	0	1	1	0	1	0	1	0	0650
SLR Contents	0	1	1	0	1	1	0	0	0660

7.2.2 Detailed Description

The Stack Limit Register logic is shown on print D-CS-M7237-0-1. The prime elements of this logic are two 74175 IC circuits (D type registers) and two 7485 IC circuits (4-bit comparators).

The high byte of the Stack Limit Register is loaded by a Unibus reference by the processor or console to the SLR bus address. The processor decodes this address and routes the data through the D MUX to the Stack Limit Register logic, providing a proper SSYN signal on the Unibus. These D MUX signals are loaded through the 5384 gates to the 74175 registers with the processor providing the clocking signals. The clock input is true when the Stack Limit Register has been selected for use (ADRS 777774 H is true) and is being loaded (DATI HIGH H is true). Under these conditions, the register is clocked, storing the desired value, and the value of the Stack Limit Register is applied to the input lines of the comparators. The 8881 gates provide a Unibus output so that the Stack Limit Register can be read. A processor or console reference to the SLR address with a DATI bus cycle enables the 8881 gates. Again, the basic KD11-A Processor provides all Unibus signals in addition to the gating signals.

The two comparator ICs function as a single 8-bit comparator circuit. The 8-bit byte that indicates the value of the Stack Limit Register is the A input to the comparator. The high byte of the Bus Address Register (which indicates the address of the bus operation being performed) is applied as the B input to the comparator circuit.

If A<B, indicating that the bus operation is *not* infringing on the stack because the bus address is higher than the stack limit value, no action occurs.

If A=B, it indicates that either a yellow (warning) or red (fatal) stack error exists because the stack limit value and the high byte of the bus address are identical. In this case (A=B), bits 07 through 05 are examined by the processor address decoding logic. If *all* three of these bits are set, then K1-7 BA (07:05)=1 L is true, gates are enabled, and KJ-2 EOVFLW L indicates a yellow zone violation. Note that one line on the gate that produces KJ-2 EOVFLW L is tied to +5 V. When the KT11-D Memory Management option is installed, that input is used to inhibit all overflow conditions in user mode.

If any one of the bus address bits 07 through 05 is *not* set, then the signal K1-7 BA (07:05)=1 L is high and qualifies an AND gate for KJ-2 BOVFLSTOP 11, thereby indicating a red zone violation.

If A>B, indicating that the bus operation *is* infringing on the stack because the bus address is lower than the stack limit value, then a red zone violation occurs and the logic produces KJ-2 EOVFL STOP H, which is used by the processor to provide appropriate service of the error.

Table 7-2
Detecting Type of Violation

	High-Order Byte								Low-Order Byte								
Bus Address	**15**	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**	
421								1	0	0	0	1	0	0	0	1	VALID
.												.					
.			More than SLR									.					
.												.					
400								1	0	0	0	0	0	0	0	0	
377								0	1	1	1	1	1	1	1	1	YELLOW
.												.					
.			Equal to SLR;									.					
.			bits 7,6,5 are									.					
.			all set									.					
.												.					
340								0	1	1	1	0	0	0	0	0	
337								0	1	1	0	1	1	1	1	1	RED
.												.					
.			Equal to SLR;									.					
.			bits 7,6,5 are									.					
.			*not* all set									.					
.												.					
000								0	0	0	0	0	0	0	0	0	

NOTES: 1. In above example, SLR is loaded with 000.

2. In *all* cases, highest yellow zone address must end in either 377 or 777.

3. In *all* cases, highest red zone address must end in either 337 or 737.

7.3 KM11-A MAINTENANCE CONSOLE

The KM11-A Maintenance Console (also referred to as the maintenance module) provides the user with a means of manually operating the system and monitoring machine states during maintenance operations.

The maintenance console itself contains four switches and 28 indicators that monitor various signals within the processor. When an indicator lights, it means that the associated logic level is high. An overlay can be attached to the module to indicate what signals are being monitored. This overlay is necessary because the console is designed as a general-purpose device and can be used, with different overlays, in many PDP-11 devices. The specific functions monitored by the console depend on the logic signals wired to the device.

If the maintenance console is to be used for monitoring KD11-A Processor operation, then the KD11-A overlay (Figure 7-1) is used and the module is inserted into processor slot F01. The functions controlled by the switches and monitored by the indicators are listed in Table 7-3.

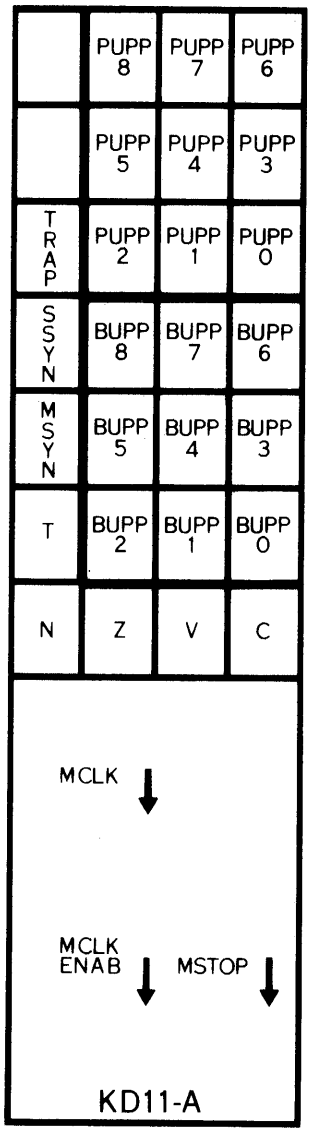


Figure 7-1 KD11-A Maintenance Console Overlay
(A-SS-5509081-0-12)

Table 7-3
KM11-A Controls and Indicators for KD11-A Overlay

Control Indicator	Indication	Print Showing Signal Origin
PUPP (08:00)	Indicates the Previous Microprogram Pointer (PUPP). These nine indicators represent a 3-digit octal word from 000 to 777. These indicators are the ROM address of the present U WORD.	K2-2, K2-3
BUPP (08:00)	Indicates the output of the Basic Microprogram Pointer (UPP) Register. In effect, displays the address of the next U WORD (includes branching).	K2-2, K2-3
TRAP	Indicates that the TRAP signal is present.	K3
SSYN	Unibus Slave Sync (SSYN) is present.	K4-6
MSYN	Unibus Master Sync (MSYN) is present.	K4-4
T	T bit of the processor status word is present. This bit is used in program debugging and results in a trap sequence.	K5-2
C	Carry bit of the processor status word condition code is present (previous operation resulted in a carry from the most significant bit).	K5-2
V	Overflow bit of the processor status word condition code (operation resulted in arithmetic overflow).	K5-2
Z	Zero bit of the processor status word condition code is present (result of operation was 0).	K5-2
N	Negative bit of the processor status word condition code is present (result of operation was negative).	K5-2
MCLK ENAB	When active (in direction of arrow) this switch prevents the automatic reclocking of the CLK flip-flop on TIMING (print K4-2). The asynchronous restart of the CLK after bus cycles is also inhibited. The machine halts after each microword and during bus cycles (including INTR). The IDLE flip-flop is not affected, and NODAT timeout flag (print K4-6) is disabled.	
MCLK	This switch (when moved toward the arrow) clocks the MCLK flip-flop on TIMING (K4-6) print and provides the timing pulses for the present microword. The user can follow the flow diagrams one microword at a time (Chapter 4 of this manual) to determine the proper indications on the maintenance module and the programmer's console. Use of this maintenance clock is considered to be single clock operation.	

Table 7-3 (Cont)
KM11-A Controls and Indicators for KD11-A Overlay

Control Indicator	Indication	Print Showing Signal Origin
MSTOP	This switch is used to examine a specific microword in a program. The address of the microword to be examined is set into the programmer's console Switch Register bits (08:00) and MSTOP is set to ON (toward arrow). The program is then started in a normal manner and continues running until it reaches the microword address that has been set into the Switch Register. At that time, the K1-9 UPP MATCH H signal loads the IDLE flip-flop of TIMING (print K4-2) to a 1, causing a machine halt. MCLK can continue operation. Note that MSTOP can only be used at the machine speed if the previous microword is of a CL2 or CL3. A CL1 word does not allow the UPP MATCH logic sufficient time for comparison. If single clock operation is being used, all cycle lengths may be used.	

If the console is to be used for monitoring operation of the KT11-D Memory Management Option and/or the KE11-E Extended Instruction Set and KE11-F Floating Instruction Set Options, then the KT11-D, KE11-E,F overlay (Figure 7-2) is used and the module is inserted into processor slot E01. In this case, the 16 indicators at the end of the overlay are used for the KT11-D functions and the 12 indicators near the switches are used for the KE11-E,F functions. Note that none of the switches are operational when the console is used for this purpose. The functions monitored by the indicators are listed in Table 7-4 and must be correlated with the information in specific microwords of the flow diagram.

7.3.1 Functional Description

The KM11-A Maintenance Console consists of 28 indicator lights, four control switches, control switch logic, and 28 indicator driver circuits mounted on a 2-module set.

The 28 indicator driver circuits provide a low output level (activating the lamps) when a high logic level is the input. The driving circuits have a high input impedance and can be used on fully loaded TTL logic output.

The four control switches and associated control switch logic initiate logic sequences and conditions in the unit tested by generating three key logic signals (switches S2, S3, and S4) with a grounding control signal (S1). Switches S2 and S4 are normally used for clock enable and clock signals, respectively.

7.3.2 Physical Description

The KM11-A Maintenance Console is contained on two modules: Maintenance Board 1 (W130 module) and Maintenance Board 2 (W131 module). The W130 module contains the 28 indicator driver circuits and connects the control switch signals and +5 V between the unit under test and the W131 module. The W131 module contains the indicator lights, the control switches, and the control switch logic. The maintenance console is shown on print D-BS-KM11-0-MB.

The W131 module plugs into the W130 module, which, in turn, plugs into the unit under test. Pin and signal designations for the W131 connector are shown on print KM-3.

7.3.3 Configurations

Because of the number of functions to be monitored, some PDP-11 units have two slots for use with the KM11-A. In these instances, the KM11-A can be used in one slot or the other, depending on what is being monitored; or, two KM11-A consoles can be used so that all functions can be monitored simultaneously. Table 7-5 lists PDP-11 units tested and includes the number of available slots.

7.3.4 Power

The KM11-A receives two voltages from the unit under test. The +5 V power is applied at pin A2 of the W130 connector and is used to drive the W131 control switch logic. Nominal +8 V power is applied at pin B1 of the W130 connector and provides power to the indicator lights. Each indicator driver circuit controls the voltage across its respective indicator light. The driver circuits are driven by the logic power of the signals being monitored.

Note that no +8 V power is available in the KD11-A Processor backplane; +5 V power is used for the indicator lights.

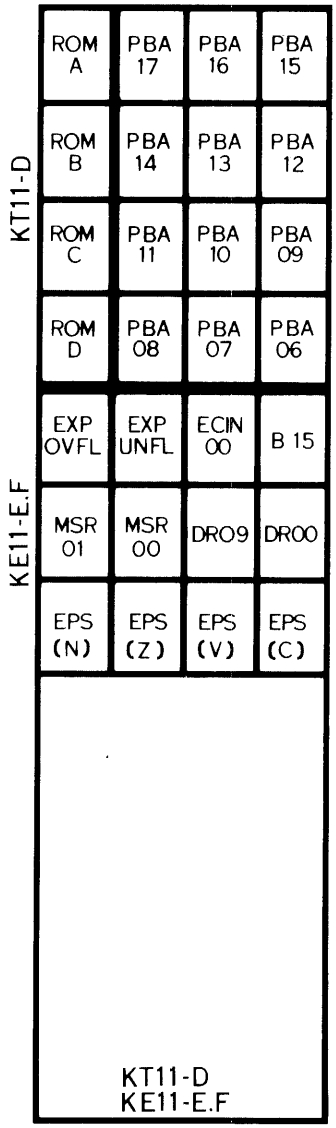


Figure 7-2 KT11-D, KE11-E,F Maintenance Console Overlay
(A-SS-5509081-0-13)

Table 7-4
KM11-A Indicators for KT11-D and KE11-E, F Overlay

Indicator	Indication	Print Showing Signal Origin
* PBA (15:06)	Indicates a logic 1 in the associated bit of the physical bus address. Note that the physical bus address is the address from the KT11-D and may be different from the address in the Bus Address Register of the processor.	KT-4
* ROM A, ROM B	These two lights form a pattern to indicate the appropriate mode and the space to be used on a memory access. The pattern is listed below. A 0 indicates the light is off; a 1 indicates it is on. **ROM A** **ROM B** 0 0 Current Mode 0 1 Temporary Mode 1 0 MTPI/D, Previous Mode or not MTPI/D, Current Mode 1 1 MFPI/D, Previous Mode or not MFPI/D, Current Mode	KT-2
* ROM C	Indicates presence of ROM bit C which is used to enable clocking of PS (15:14) current mode into PS (13:12) previous mode for future controlled access and clocking of T (15:14).	KT-2
* ROM D	Indicates presence of ROM bit D which is used in conjunction with the final bus cycle of the KD11 instructions for relocation in destination mode only.	KT-2
B15	Bit 15 of CPU B Register. In divide, used with DR00 to determine the ALU function to be performed in division loop.	K1-5
ECIN 00	An external carry-in to the ALU.	KE-5
EXP UNFL	Indicates exponential underflow during EXI1 of floating point flows.	KF-4
EXP OVFL	Indicates exponential overflow during EXI1 of floating point flows.	KF-4
DR00	Used in conjunction with other bits to indicate various conditions; e.g., with B15 in divide to determine ALU functions and to determine need for divisor correction. See EPS(C) for other use.	KE-2
DR09	Used as test for normalization.	KE-2

*These indicators are used only with the KT11-D Memory Management Option; the remaining indicators are used with the KE11-E EIS and the KE11-F FIS Options.

Table 7-4 (Cont)
KM11-A Indicators for KT11-D and KE11-E, F Overlay

Indicator	Indication	Print Showing Signal Origin
MSR00	Bit 00 of MSR Register. Indicates ALU function in FDIV.	KF-2
MSR01	Bit 01 of MSR Register. Indicates ALU function in FMUL.	KF-2
EPS(C)	C bit of extended processor status. In MUL, used with DR00 to determine ALU function in multiply loop.	
EPS(V)	Overflow bit of the extended processor status.	KE-6
EPS(Z)	Zero bit of the extended processor status.	KE-6
EPS(N)	Negative bit of the extended processor status.	KE-6

NOTE: The functions described in Table 7-4 describe the general purpose of the indicator. At times, a single indicator may show a number of functions, depending on the current state of the processor and option. This is why in order to use the maintenance module properly, the flow diagrams should be followed to determine the significance of an indication at any one time.

Table 7-5
KM11-A Configurations

Unit Tested	Available Slots	Remarks
KD11-A Processor	2	one slot used for KD11-A; one slot used for KT11, KE11-E, F
KT11-D Memory Management	0	uses KD11-A Processor slot shares overlay with KE11
KE11-E, F Extended Instruction Sets	0	use KD11-A Processor slot shares overlay with KT11
TM11 DECmagtape Control	1	peripheral controller
DT11 Bus Switch	1	
RK11-C Moving Head Disk Drive Control	2	peripheral controller overlays labeled: RK11-1 RK11-2

7.4 KW11-L LINE FREQUENCY CLOCK

The KW11-L Line Frequency Clock is a PDP-11/40 processor option that provides a method of referencing real intervals. This option generates a repetitive interrupt request to the processor. The rate of interrupt is derived from the ac line frequency, either 50 Hz or 60 Hz. The accuracy of the clock period, therefore, is dependent on the accuracy of this frequency source.

The KW11-L Line Frequency Clock can be operated in either an interrupt or non-interrupt mode. When the interrupt mode is used, the clock option interrupts the processor each time it receives a pulse from the line frequency source. In the non-interrupt mode, the clock option functions as a program switch that the processor can either examine or ignore. Mode selection is made by the program.

The KW11-L Line Frequency Clock is installed in slot F03 of the KD11-A Processor backpanel. Installation requires that a backpanel wire between pins F03R2 and F03V2 be removed. This places the KW11-L option in the BG6H signal line.

7.4.1 General Description

The KW11-L Line Frequency Clock is a single-height module containing an address selector, threshold detector, interrupt control, and a 2-bit Status Register. A block diagram of the clock is shown in Figure 7-3 with details on prints D-BS-KW11-L-0-1 and D-CS-M787-0-1 of the *PDP-11/40 System Engineering Drawings.*

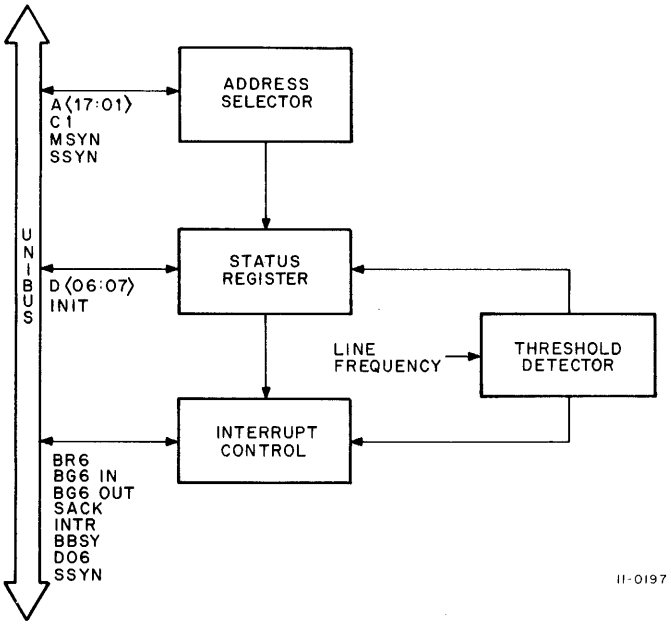


Figure 7-3 KW11-L Block Diagram

When the KW11-L is in interrupt mode, the interrupt control section of the option provides the circuits and logic required to make bus requests, gain bus control, and generate interrupts. Whenever the threshold detector provides a pulse from the line frequency source, the interrupt control section of the clock initiates a bus request on priority level 6 (BR6), which is the priority level of the clock.

The priority logic in the processor recognizes the request and issues a Bus Grant signal if the clock is the highest priority device requesting an interrupt. The KW11-L responds with a Selection Acknowledge (SACK) signal. When the requirements for becoming bus master have been fulfilled, the clock asserts Bus Busy (BBSY), an Interrupt (INTR) signal, and an interrupt vector address of 100. The processor generates a Slave Sync (SSYN) signal, then responds to the interrupt with an interrupt service routine. The interrupt control section of the clock then enters a rest state until the next initialization.

The 2-bit Status Register in the clock consists of bits 6 and 7 on the data bus line. When bit 6 is set, the clock is in the interrupt mode; when it is clear, the clock is in the non-interrupt mode. Bit 6 is loaded by a Unibus DATO to the clock; it is also cleared by Unibus INIT. Bit 7 is loaded to a 1 by a line clock pulse from the threshold detector or cleared by a Unibus INIT; it is cleared by any Unibus DATO to the clock.

Bit 7 can be used by the processor to determine which device causes the interrupt. The interrupt service routine should include a DATI which reads the interrupt monitor bit (bit 7) to serve as a partial check on the origin of the interrupt vector. Thus, if bit 7 is clear there is an indication to the processor that the clock did not request the interrupt.

In the non-interrupt mode, the clock performs a more passive function by serving as a program switch that the processor can examine or ignore. The interrupt control section is disabled so that the clock cannot assert a bus request (BR6) and, therefore, cannot go into an interrupt sequence. A programmed DATO must be used to return the clock to the interrupt mode; programmed DATIs must be used to examine the status of the clock. In the non-interrupt mode, the clock is controlled by programmed instructions from the processor.

7.4.2 Address Selector

The address selector logic of the KW11-L clock is permanently wired to respond to incoming address 777546. Input signals consist of address, BUS A (17:00); Bus Control, BUS C1; and BUS MSYN (drawing D-BS-KW11-L-0). BUS A00, which is used for word or byte control, is not brought into the clock because the KW11-L deals only with full 16-bit words. When the address is decoded by the address selector and BUS MSYN is active, gate E3 output goes high (drawing D-BS-KW11-L-01), thereby signaling that the clock has been addressed.

7.4.3 Interrupt Control

The interrupt control section of the KW11-L Clock provides the necessary logic for issuing bus requests, gaining bus control, and generating interrupts. The interrupt logic uses three flip-flops: INTERRUPT REQUEST, FF1, and FF2 (Figure 7-4). Table 7-6 lists the settings of these flip-flops in relation to the bus states and the signals asserted.

When the clock is not issuing an interrupt request, all three flip-flops are in the 0 state and no signals are asserted on the bus. The request state is entered when the INTERRUPT REQUEST flip-flop is set by a line clock pulse. This setting of the flip-flop can occur only when the status bit 6 flip-flop (interrupt enable) is in the 1 state. Setting the INTERRUPT REQUEST flip-flop generates a BR6 request.

The priority arbitration logic of the processor determines whether priority level 6 is the highest requesting level. If BR6 is the highest level, then the processor asserts a Bus Grant signal (BG6 IN H) that sets the FF1 flip-flop. Signal BG6 is blocked from being passed on to the next device and the assertion of BR6 is dropped. With flip-flop FF1 set and flip-flop FF2 clear, the Selection Acknowledge (SACK) signal is asserted on the bus.

On receiving the SACK signal, the processor drops BG6 IN and flip-flop FF2 is set, provided SSYN and BBSY are both unasserted. The BBSY and INTR signals are then asserted on the bus, as well as interrupt vector address 100 (BUS D06).

The processor responds to these signals by asserting a Slave Sync (SSYN) signal that clears the INTERRUPT REQUEST flip-flop. Flip-flops FF1 and FF2 are subsequently cleared, causing the interrupt control section of the KW11-L Clock to return to the non-requesting state. At the same time SSYN is asserted, the processor enters the interrupt service routine at vector address 100.

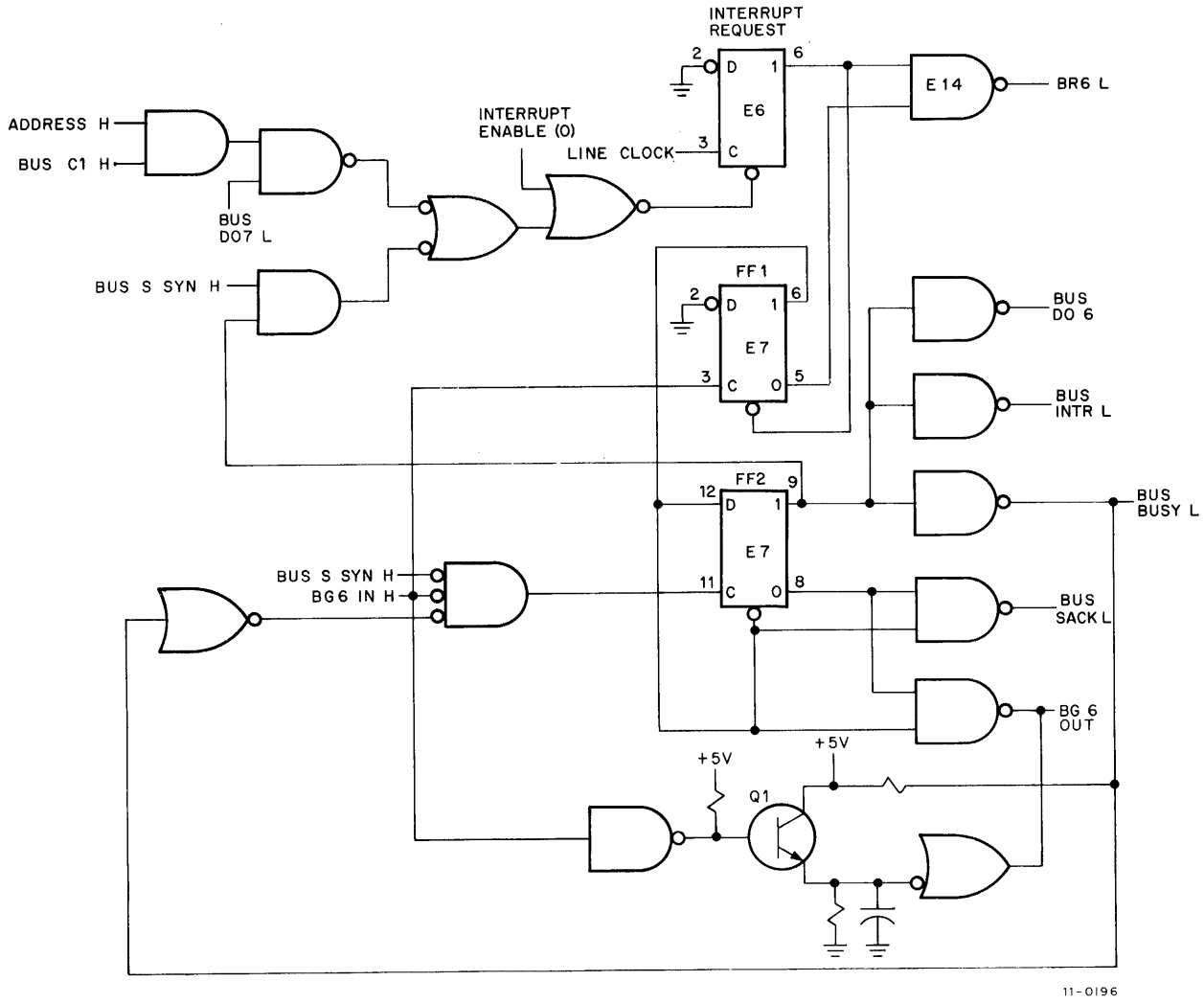


Figure 7-4 Interrupt Request Section, Simplified Diagram

Table 7-6
Interrupt Control Flip-Flops

Interrupt Request	FF1	FF2	State	Signals
0	0	0	Not requesting	None
1	0	0	Requesting	BR6
1	1	0	Granted	SACK, BG6 OUT inhibited
1	1	1	Master	BBSY, INTR, BUS D06 (vector address)

7.4.4 Status Register

The Status Register of the KW11-L contains the INTERRUPT ENABLE and the INTERRUPT MONITOR flip-flops (Figure 7-5). Operation of the Status Register logic is controlled by INIT, the line clock pulse, and DATO and DATI transfers.

The INIT signal is generated by either pressing the START switch on the programmer's console or by issuing a programmed RESET instruction. The INIT signal clears the flip-flops to initialize the Status Register for a new operation.

The line clock pulse supplied by the threshold detector is used to set the INTERRUPT MONITOR flip-flop (bit 07). A DATO and ADDRESS H clear the INTERRUPT MONITOR flip-flop, provided BUS D07 is high, by applying a signal to the direct clear input of the flip-flop.

In order for DATO and DATI transfers to affect the logic of the Status Register, the address of the KW11-L and MSYN must be asserted on the bus to provide the ADDRESS H input as shown in Figure 7-5. The ADDRESS H signal is also used, after a delay, to assert SSYN on the bus.

The combination of DATO and ADDRESS provides a signal to the clock input of the INTERRUPT ENABLE flip-flop. Depending on BUS D06, the flip-flop is either set or cleared. Thus, the processor can write a bit into this flip-flop by issuing a DATO and BUS D06=1 for a 1 and a DATO and BUS D06=0 for a 0. The 0 side output of the INTERRUPT ENABLE flip-flop controls the interrupt function of the clock by holding the INTERRUPT REQUEST flip-flop in the interrupt control section in a cleared state when INTERRUPT ENABLE is in the 0 state.

A DATI and ADDRESS H provide gating that reads the contents of INTERRUPT ENABLE on BUS D06 and the contents of INTERRUPT MONITOR onto BUS D07.

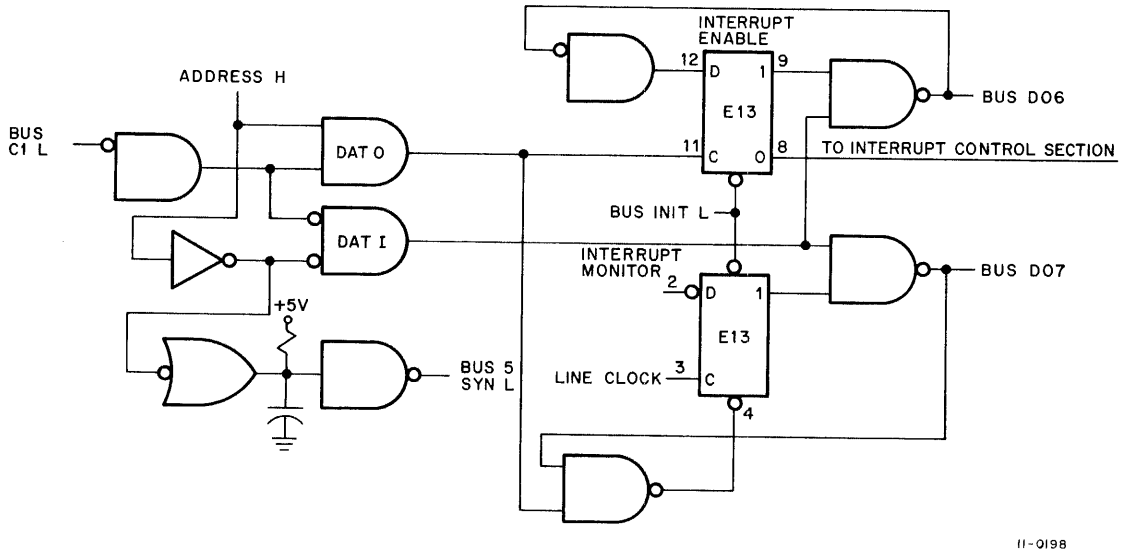


Figure 7-5 Status Register, Simplified Logic Diagram

KD11-A PROCESSOR
MAINTENANCE MANUAL
EK-KD11A-MM-001

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? _____

What features are most useful? _____

What faults do you find with the manual? _____

Does this manual satisfy the need you think it was intended to satisfy? _____

Does it satisfy *your* needs? _____ Why? _____

Would you please indicate any factual errors you have found. _____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

CUT OUT ON DOTTED LINE

— — — — — — — — — — — — Fold Here — — — — — — — — — — —

— — — — — — — — — Do Not Tear - Fold Here and Staple — — — — — — — — —

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

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

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
146 Main Street
Maynard, Massachusetts 01754