KD11-A processor manual

pdp11

digital

DEC-11-HKDAA-A-D

KD11-A processor manual

digital equipment corporation · maynard. massachusetts

TABLE OF CONTENTS

1 INTRODUCTION

1.1 SCOPE

This manual describes the KD11-A Processor which is the basic component of the PDP-11/40 Computer System. The processor is connected to the Unibus as a subsystem and controls time allocation of the Unibus for peripherals, 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 in this manual (KY11-D, KJ11-A, KM11-A, and KW11-L).

This manual provides the reader with the information necessary to understand the normal operation of the KD11-A processor. Because the processor is a complex digital device, the user must understand normal processor operations in order to fully use its capabilities or to recognize and correct the cause of improper operations.

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

Table 1-1

Related Documents

Title	Number	Remarks
PDP-11/40 System Manual	DEC-11-H40SA-A-D	Describes overall PDP-11/40 system and includes sections on installation, operation, and programming.
KE11 Instruction Set Options Manual	DEC-11-HKEFA-A-D	Provides complete coverage on both the KE11-E Extended Instruction Set and KE11-F Floating Instruction Set processor options.
KT11-D Memory Management Option Manual	DEC-11-HKTDA-A-D	Provides complete coverage on the memory management option used with the processor.

1.2 ORGANIZATION

The description of the KD11-A processor itself is divided into four main sections: microprogramming, block diagram, flow diagrams, and logic diagrams.

Because microprogramming may be a new concept for the reader, the section on microprogramming (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.

The section describing the processor at a block diagram level (Chapter 3) 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 all 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 included on the flow diagram itself. Therefore, the section covering flow diagrams (Chapter 4) is 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.

The last section covering the KD11-A processor is Chapter 5 which 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 used with the processor, with the exception of operating procedures which are covered in the PDP-11/40 System Manual, DEC-11-H40SA-A-D.

Chapter 7 provides a complete description of three of the internal processor options that may be used with the KD11-A. These options are: KW11-L Line Frequency Clock, 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 other manuals listed in Table 1-1.

A complete drawing set is supplied with this manual and includes the basic block diagram, microword format, function tables, flow diagrams, and logic diagrams. The drawings are supplied in a companion volume entitled, PDP-11/40 System, Engineering Drawings. 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.

2 MICROPROGRAMMING

2.1 SCOPE

The purpose of this chapter is to provide a general introduction of 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. Therefore, 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 <u>instructions</u>. An instruction can include a number of operations so that data can be combined, changed, moved, or disposed of. The instructions can be further combined into <u>programs</u> 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 described in terms of the machine states by listing all of the states in which the processor can function. That is, all of the different operations or groups of operations that it can perform and all of the valid sequences in which these states occur. The sequence of machine states is determined by the current state of the computer system. For example, what instruction is being executed, the values of the data being operated on, and the results of the previous 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 section, which performs data handling functions; and the control section, which includes the logic that dtermines which operations are to be performed during a particular state and what the next machine state should be.

The interface section basically consists of logic necessary for transferring data between the processor, the Unibus, and the programmer's console.

The data and control 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 able to store 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. Whenever the processor must select some of this internally stored data, or store new data, the control 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 control section. The various logic elements that actually modify data are controlled by signals from the control 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 set up as required by the specific machine state. The control section generates signals that cause data routing logic elements to form appropriate interconnections within the processor and between the interface and data sections of the logic.

2.3 CONVENTIONAL IMPLEMENTATION

Before attempting to understand the microprogramming implementation of the control section, which is the key to the KD11-A processor, it is advantageous to review the conventional method of control section 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 control 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.

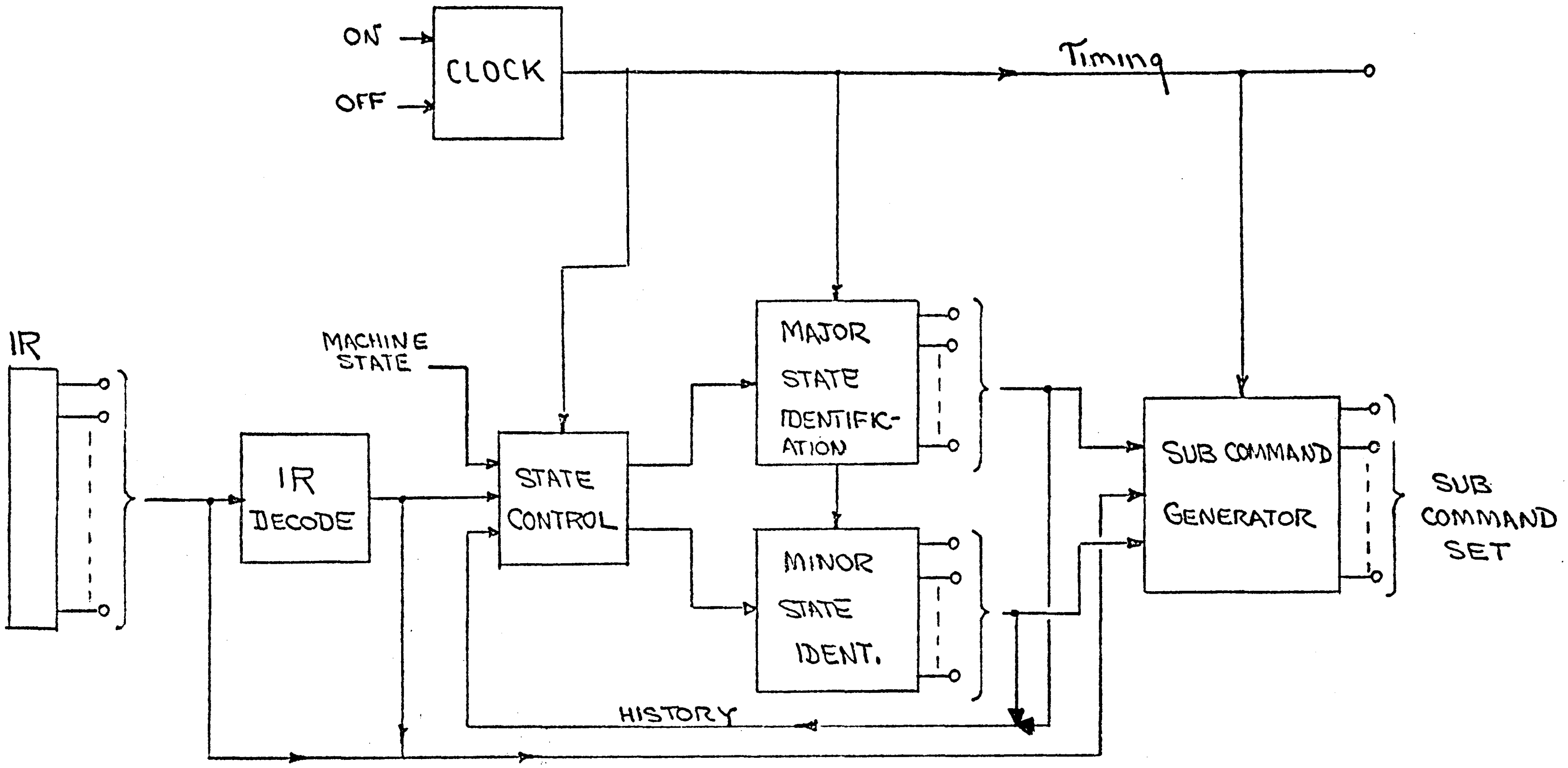

Figure 2-1 Conventional Control Section - Simplified Diagram

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 on the block diagram. 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 selects the next elementary operation as a function of the current operation and external conditions.

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

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) that 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 to generate 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.

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 <u>next</u> 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.

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 retrieves a specified control word (microword) from the ROM. This microword contains the control fields used by the processor to perform the selected function. The control word also contains a next address field and a branch test field which are fed back to the address generator to select the next microword in the sequence.

The combination of the next address field and the branch test field provides the means of controlling 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 counter.

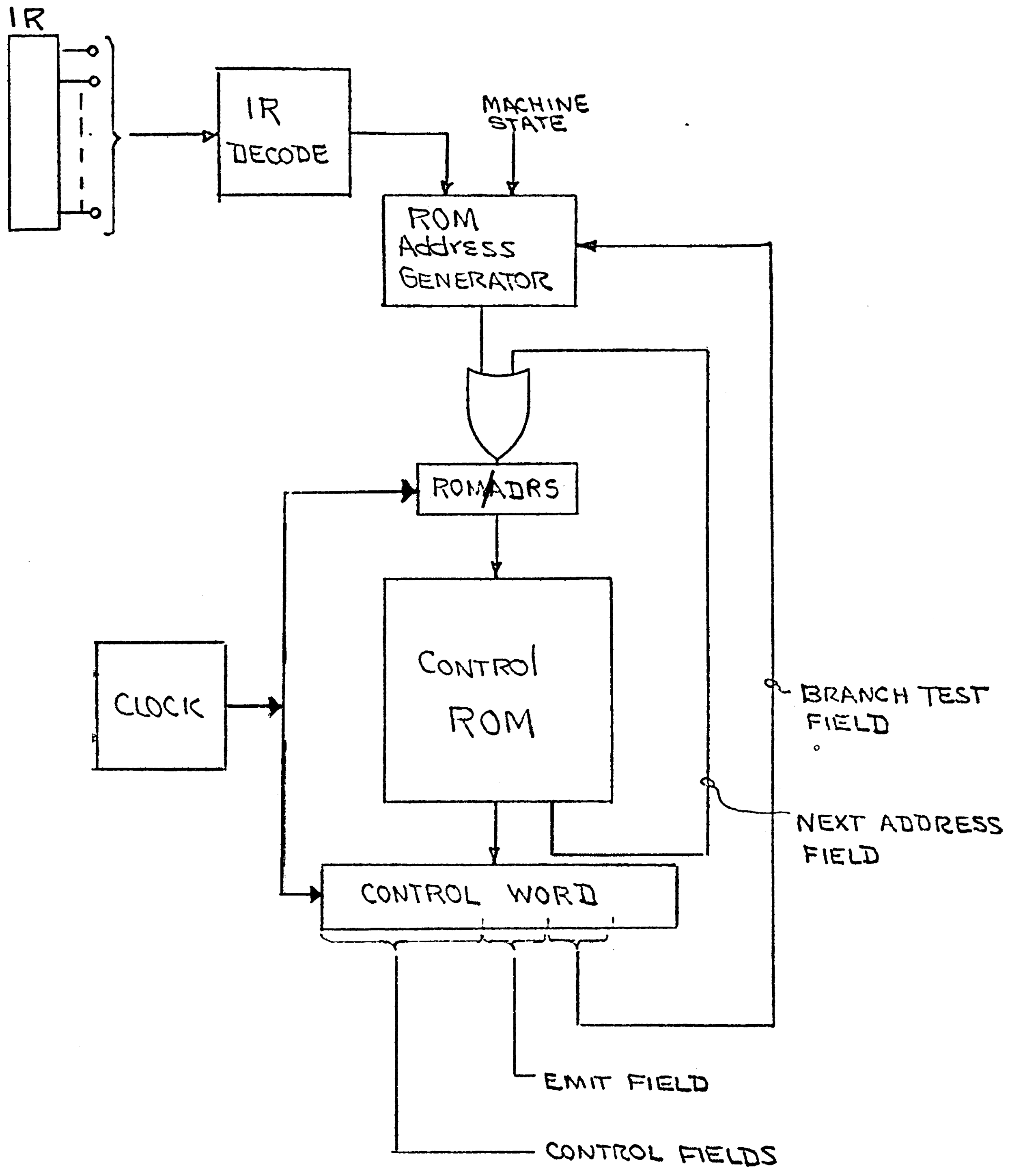

Figure 2-2 Microprogrammed Control Section - Simplified Block Diagram

Before discussing the address modification, it is important to understand that modification occurs prior to storage in the ROM ADRS block 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 in the ROM ADRS and, therefore, cannot be modified. However, when word 1 is being used and word 2 is in the pointer, the address for word 3 can be modified between the ROM output and the ROM ADRS.

The branch test field of the microword specifies conditions to be tested and controls when a branch is to occur and to what location the microprogram is to branch to. Other 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 block, the microprogram branches to the required location and retrieves the necessary microword 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 (bits 08:00) are used to hold the address of the next microword to be used. The remaining bits (56:09) are various control bits.

Figure 2-3 shows the basic structure of the microwords in the ROM. The detailed format of the microword is shown on print D-BD-KD11-A-BD. 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, the prefix micro (from the Greek Mu) is abbreviated as U (similar to). 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.

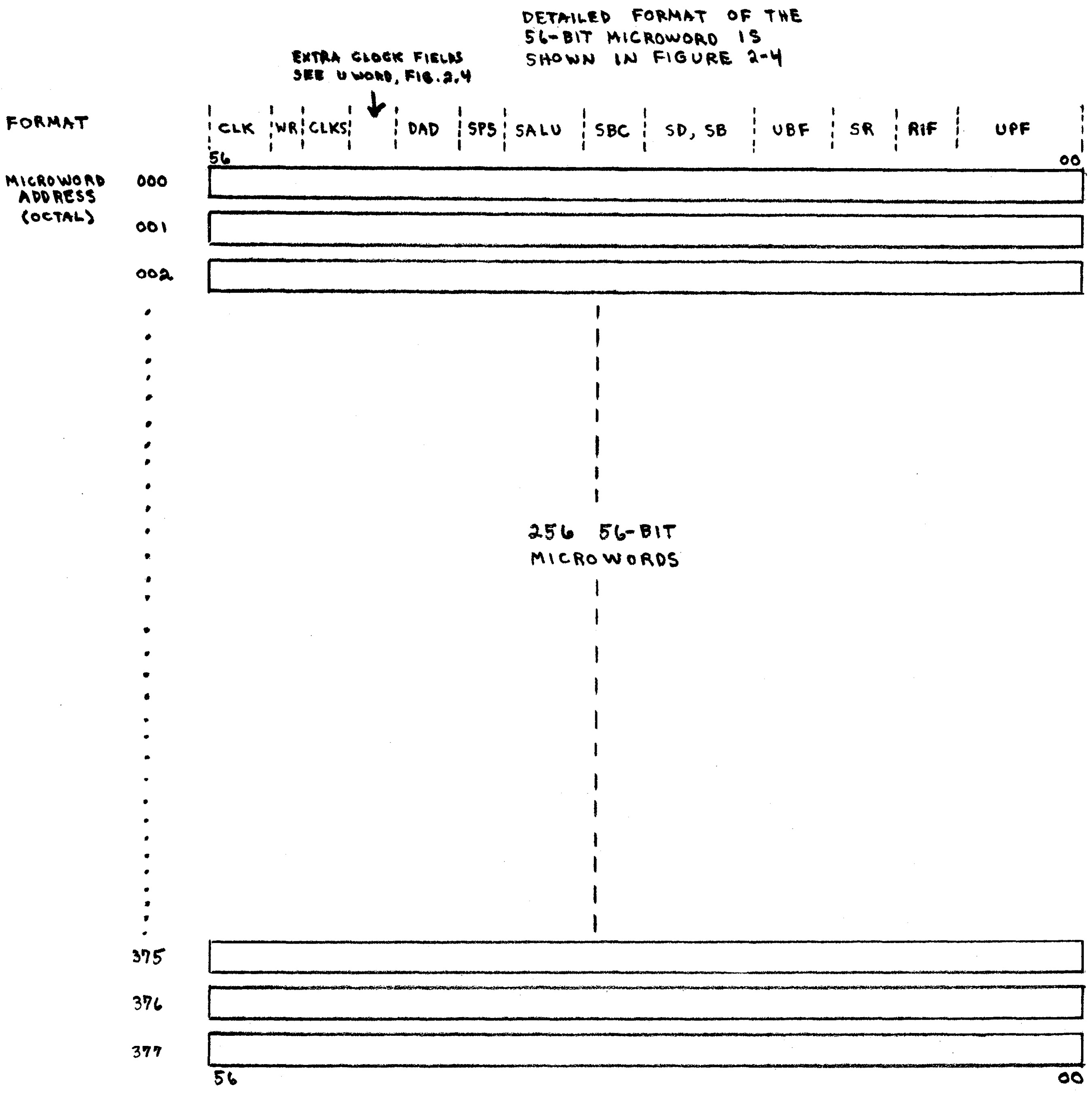
DETAILED FORMAT OF THE 56-BIT MICROWORD IS SHOWN IN FIGURE 2-4
EXTRA CLOCK FIELDS SEE U WORD, FIG. 2.4
FORMAT
CLK
WR
CLKS
DAD
SPS
SALU
SBC
SD, SB
UBF
SR
RIF
UPF
56
00
MICROWORD ADDRESS (OCTAL)
000
001
002
256 56-BIT MICROWORDS
375
376
377
56
00

Figure 2-3 Basic ROM Structure

Table 2-1

Function of Microword Bits (U WORD)

U Bit	Mnemonic	Meaning and Function
56 55	CLK1 CLK0	Clock length control. Permits the microprogram to select one of three clock lengths.
54	CLKOFF	Permits microprogram to turn off the processor clock.
53	CLKIR	Permits clocking Unibus data into the instruction register (IR).
52 51	WRH WRL	Permits 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	Specifies the type of data transfer bus transaction.
45	BGBUS	Initiates data transfer bus transaction.
44 43 42 41	DAD3 DAD2 DAD1 DAD0	Discrete alteration of data. Permits microprogram to alter operation of the data path. For example, modifying the ALU operation as a function of the instruction register.
40 39 38	SPS2 SPS1 SPS0	Controls loading and clocking of the processor status word.
37	SALUM	Selects the mode of ALU operation (mode can be either arithmetic or logical).
36 35 34 33	SALU3 SALU2 SALU1 SALU0	Selects the operation to be performed by the arithmetic logic unit (ALU) such as add, subtract, etc.

Bit	Mnemonic	Meaning and Function
32 31 30 29	SBC3 SBC2 SBC1 SBC0	Permits 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	Selects the input to the high-order byte of the B multiplexer.
26 25	SBML1 SBML0	Selects the input to the low-order byte of th e B multiplexer.
24 23	SDM1 SDM0	Selects the source of the input to the D multiplexer.
22	SBAM	Selects the source of the input to the bus address multiplexer.
21 20 19 18 17	UBF4 UBF3 UBF2 UBF1 UBF0	Represents microbranch field. Selects the microbranch condition to be tested. (This test is referred to as BUT, branch microprogram test.)
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 bits (12-09) for general register address.
12 11 10 09	RIF3 RIF2 RIF1 RIF0	Permits 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	Represents 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 branch test (BUT). The U08 bit is for UPF8 and is provided by the KE11-E option.

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 (print D-BD-KD11-A-BD) 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 paragraphs 3.2, 3.3, and 3.4, respectively. In addition, 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.

In the corner of each logic block on the block diagram is a K reference that indicates the module print upon which the logic occurs.

3.2 INTERFACE LOGIC

The first section of the processor shown on the block diagram is the interface logic which is used to interconnect the KD11-A processor with other components of the PDP-11/40 System such as the programmer's console, Unibus, etc. 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 systemand 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 logic that is considered to be a part of the processor interface section includes the switch register, the data display, the address display, and the console control.

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. The microprogram addresses the switch register during console operation and decoding the address enables the driver gates so that the value set on the switch register is loaded onto the Unibus.

The data display indicates the output of the processor data multiplexer which gates information from a variety of sources within the processor,and also gates data from the Unibus. The display consists of indicator lights (light emitting diodes) and associated current limiting resistors mounted in the programmer's console. These indicators are connected to the processor by cables. The output line of the data multiplexer (D MUX <15:00>) always controls the display. However, because the multiplexer can select multiple inputs onto the output line, information can be displayed from a variety of sources.

The address display indicates the contents of the processor bus address register (BA register). This display also consists of light emitting diodes 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 was the case with the data display. Although it is possible to load specific data into the bus address register for different situations arising in the logic flow, the contents of the bus address register is always displayed by the address display.

The 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 by means of both the microprogram and combinational logic flag flip-flops. The microprogram senses switch activation and branches to the specific routine required,depending on which switch has been used. The flags accomodate the special needs of the START and CONTINUEswitch sequences as well as the incrementation requirements of consecutive EXAMINE or DEPOSIT 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 bus with associated error checking (odd address, stack overflow) and correction (data time-out). An extra processor signal (MSYN A) is included for faster, parallel use with the Unibus MSYN signal. The logic also provides the gates and signals needed for the processor to respond once it has been addressed from the bus.

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 time-out 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.

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 acceptance of bus data and the deactivation of MSYN occur as a function of the next microword after a DATI or DATIP transfer operation; and the processor transmits address and data information to the bus under control of the microprogram. Note, however, that bus ownership, as well as the power fail logic, operates asynchronously and is independent from the microprogram.

The interface portion of the processor contains both bus transmitters (8881 gates) and bus receivers (7380 gates) 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 handled 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 provides the means of interconnecting 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 and connector module (M981) 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 arithmetic logic unit (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 labled REGISTER (REG).

The scratch pad register and the arithmetic logic unit interact in that the 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." Because of this input, the characteristics of the scratch pad register affect the data path structures. 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 into the B input of the ALU.

3.3.1 DATA PATHS, MULTIPLEXERS AND REGISTERS

Basically, there are two inputs to the ALU: A and B. The A input provides variable operands, the B input provides variable operands, constants, and sign-extended operands. The A input always comes from the scratch pad register although it can be wire ORed with basic processor inputs from the scratch pad register and the processor status register (as shown by the dotted OR gates on the block diagram).

The B input comes from the B multiplexer (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 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 multiplexer 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 permits 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 multiplexer output can be stored in the B register which in turn can be fed to the ALU by means of the B multiplexer. It should be noted that the buffered BUS RD signal can be fed through the D multiplexer into the B register. This data path is of special interest in the machine instruction for the register-to-register operations, where the B input of the ALU must come from the scratch pad register. For example, if both desired operands are stored in the scratch pad register, the first operand passes through the D multiplexer 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 multiplexer.

The B constants, which are applied through the B multiplexer 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 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 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 of the ALU equals B ⟨07:00⟩ and B ⟨15:08⟩ for the high byte and the low byte, respectively.

The arithmetic logic unit (ALU) provides an altered data output that is used for Unibus addresses and data and is also used 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 multiplexer 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 and 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.

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 zero (D<15:00>= 0 H). The BA (bus address) register is decoded to determine if a processor address has occurred, or if the address is less than specified balues. 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 is the address of the PS 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 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 (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 <07:00>), 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 status 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 non-loading 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 are referred to as 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 and 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 CONTROL LOGIC

The final section of the block diagram is the microprogram control 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 various microwords. The bits in each microword (U WORD), in turn, control machine operation as described in Chapter 2. Other elements within the control section include address and address 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 signal (basic microbranch code, BUBC) for several branch conditions and the discrete auxillary 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 path section. The operation of the basic microcontrol comes next.

3.4.1 CONDITIONAL 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. In addition, the information to be recorded can vary for different types of instructions. The condition codes logic is combinational logic that alters the condition codes during the latter part of an instruction cycle. During this time, condition codes are combinations of data register contents, overflow situations, etc. 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 MUS.

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 DAD code directly alters 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·

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. 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 microtest (BUT) multiplexer is to select the next proper machine state. The microbranch control provides some of the inputs to the branch microtest (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. On the other hand, they may be fairly simple, consisting of only three bits or, in some cases, three bits of another BUT encoded with another single condition. This is particularly true with the INSTR 2 BUBC and the (BYTE and INSTR 2) BUBC.

3.4.5 BUT MUX

The branch microtest 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, other characteristics determining the required branch are the inputs to the BUT MUX. 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 simply chooses 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 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 the ability to have 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 REG) 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 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 itself).

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 and 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 REG 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 constant, little attention would have to be given to it by 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 outputted 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 addresses 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 address of the above present U WORD are stored in the past microprogram pointer (PUPP) for reference.

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 with all 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 and occurs during word A while selecting word B which contains the address for word C. 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. In other words, 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 microprogramming address loop is also affected by the jam microprogram pointer (JAMUPP) logic which alters the sequential nature of the address loop. The JAMUPP logic provides a means of jamming an address into the microprogram pointer to modify the microprogram for certain conditions such as bus errors, stack overflow, auto restart, etc. This logic provides the next microword address directly as a function of previous start or error conditions in the machine. The output of the JAMUPP logic direct sets or direct clears the UPP register flip-flops to establish the required address. This method differs from the normal NOT/OR inputs 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 PUPP (past microprogram pointer) register at each system clock. The PUPP register maintains a history of the previous microprogram pointer and displays its contents on the maintenance console. Note that the previous pointer 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. Therefore, 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 should 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 as sync signal to trigger an oscilloscope or can be used 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 state of the address in the switch register.

3.4.11 CLOCK CONTROL

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 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, and the clock enable gates.

3.4.12 CLOCK PULSE GENERATOR

The clock pulse genrator provides the system clock pulses when triggered by the clock control logic. These clock pulses are used throughout the processor and are combined with the enable signals of the ROM to act upon the three major segments of the processor (interface, data path, and microprogram control). There are three pulses generated by the clock pulse logic: CL1 cycle which generates a P1 pulse; CL2 cycle which generates a P2 pulse; and 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 dtermined by microword clock control bits in that word. See print D-CS-M7234-0-1 for CLK waveforms.

3.4.13 CLOCK CONTROL

The clock control logic consists of a clock (CLK) and an idle (IDLE) flip-flop. The CLK flip-flop provides a pulse, and the IDLE flip-flop drives the RUN console light and indicates when the processor is sequentially processing microwords.

3.4.14 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.

The flag control logic recognizes a variety of asynchronous conditions and changes the sequence of processor operations in response to these conditions. The logic consists of discrete flip-flops and combinational logic that determines sequencing of trap elements, trap instructions, and error traps. When any of these conditions occur, the processor enters a trap service sequence of microprogram states and the logic generates a trap vector that is used to transfer system control to a specific trap service program.

3.5 MAJOR PROCESSOR COMPONENTS

Table 3-1 lists each of the major blocks shown on the processor block diagram in order to present a summary of processor components. The table lists the name of the component, provides a brief physical and functional description, and lists associated inputs and outputs. This table can be used as a quick reference or to provide a brief overview of the processor. The components are listed in alphabetical order.

Table 3-1

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 chip provide a 16-bit arithmetic logic unit with a look-ahead 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: ALUM,ALUS(3:0) 5-bit wide control that specifies ALU function	Data: Provides 16-bit output to either the D REG or to the BA register through the BAMUX. Status: COUT 7, COUT 15, ALU15 to input of COUT multiplexer.

Table 3-1
(continued)

Component	Description	Input	Output
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, ALUM,ALUS(03:00) that select the ALU function to be performed.
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 control section. STPM <04:02> from the trap sensing logic.	Selected constants applied to the B MUX.

Table 3-1
(continued)

Component	Description	Input	Output
B Multiplexer (B MUX)	Eight 74153 multiplexer chips. Provides the means of selecting the data input to the B input (BIN) of the ALU	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) d. B <07:00,07:00> (duplicate lower byte) e. B <07:00,15:08> (swap bytes) f. B <15:08=7,07:00> (sign extend)	Provides 16-bit wide input to the B input of the ALU
B Register	Four 74174 IC chips provide a 16-bit temporary storage register.	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 partioned into a high <15:08> and low <07:00> byte.

Table 3-1
(continued)

Component	Description	Input	Output
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 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 a bus driver to bus address lines BUS BA <17:00>. The address is also applied to the address display.
Bus Register Data (BUS RD)	Four 74H04 IC chips that provide 15 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. Register data (16 bits) b. Processor status (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.

Table 3-1
(continued)

Component	Description	Input	Output
Branch micro-test 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 U WORD buffer.	Control signals, especially to the flag control logic.
Branch micro-test multiplexer (BUT MUX)	Six multiplexer IC chips: three 16-line to 1-line type 74150 multiplexers two 8-line to 1-line type 74151 multiplexers one dual 4-line to 1-line type 74153 multiplexer	Any one of the following: a. IR register bits b. branch control signals c. IR decode signals d. machine status e. microword UBF <04:00> field for multiplexer selection.	Control signals that allow modification of the microprogram address field, UPF (07:00), prior to clocking the address into the UPP of the U WORD.

Table 3-1
(continued)

Component	Description	Input	Output
Buffered microprogram pointer and switch register match (BUPP & SR MATCH)	Nine exclusive OR gates connected to 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 as a sync scope signal or can be used to stop the clock during maintenance operation. 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

Table 3-1
(continued)

Component	Description	Input	Output
Clock Control	Network of combinational logic circuits that controls the clock and idle flip-flops.	CLK0, CLK1, and CLKOFF bits in the U WORD.	Control signals to the clock pulse generator.
Clock Pulse Generator	Three delay lines selected by combinational logic circuits to require the clock pulses specified by the current microword.	Same as clock control with pulse control signal from clock control.	Timing pulses P1, P2, or P3.
Clock Enable Gates	Combinational logic network that routes clock outputs to the interface, data path, and microword control portions of the processor.	Timing pulse P1, P2, or P3 from the clock pulse generator. CLKIR, CLKBA, CLKB, CLKD, WRH, WRL bits from the current U WORD.	Various clock signals. (CLK IR, CLK D, CLK BA, etc.)

Table 3-1
(continued)

Component	Description	Input	Output
D Multiplexer (D MUX)	Eight 74153 multiplexer IC chips.	A 2-bit control field selects one of the following four inputs: a. register (BUS RD) b. D register c. D register shifted right d. Unibus data	The DMUX distributes 16-bit data word to: a. Instruction Register b. General registers c. B register d. PS register e. Data display f. Internal data bus(DMUX)
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).
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.

Table 3-1
(continued)

Component	Description	Input	Output
Decoding (ADRS & DATA)	Combinational logic network that decodes the bus address and generates internal control signals for addressing processor registers. Sensing is provided for stack overflow situations.	18-bit input from bus address (BA) register	Processor status (PS) address Stack limit register (SLR) address General register (REG) address Switch register (SR) address BOVFL STOP and BOVFL signal
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.	Buffered UPP (BUPP) for application to PUPP register EUPP (expansion microprogram pointer) for an expansion ROM (KE11-E, KE11-F).

Table 3-1
(continued)

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, U branch control logic, and the BUT MUX.
Instruction Register (INSTR REG)	Four 74175 IC chips form a 16-bit storage register that holds the instruction	Output of D MUX during the instruction fetch sequence.	Output applied to IR decode logic where it is decoded and used to control the microprogram sequence
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 the UPP portion of the U WORD. Timing signals to load newly selected ROM word into the U WORD buffer.

Table 3-1
(continued)

Component	Description	Input	Output
Processor Status (PS) Register	Four 7474 IC chips provide 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>.
Past Micro-program Pointer (PUPP) Register	Two 74174 IC chips provide a 9-bit storage register for keeping a history of the previous UPP address	Loaded with the contents of the UPP register at each system clock.	Register contents displayed on KM11-A Maintenance Console option when used during maintenance operation.
Register (REG)	Four 3101 IC chips provide 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	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

Table 3-1
(continued)

Component	Description	Input	Output
Register Address (REG ADRS) Input	Combinational logic network used as an address multiplexer to select one of the 16 general-purpose processor registers for reading or writing.	There are four possible inputs. One of the four is selected by the control signals: a. IR⟨02:00⟩- 3-bit field from instruction register. b. IR ⟨08:06⟩- 3-bit field from instruction register. c. RIF⟨03:00⟩- 4-bit field from U WORD d. BA ⟨03:00⟩- 4-bit field from bus address register. Control signals are: SRD - selects IR ⟨02:00⟩ SRS - selects IR ⟨08:06⟩ SRI - selects RIF ⟨03:00⟩ SRBA- selects BA ⟨03:00⟩	Provides address selection to the register (REG).

Table 3-1
(continued)

Component	Description	Input	Output
Microbranch Control (U BRANCH CONTROL)	Large network of combinational logic circuits that provide control 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 main control ROM address as a function of BUT MUX selection.
Microword WORD Control ROM (U WORD CONTROL ROM)	A read-only memory storing the KD11-A microprogram. The ROM stores 256 56-bit words. Fourteen ROM chips provide 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>).

Table 3-1
(continued)

Component	Description	Input	Output
Microword WORD Register (U WORD REG)	A 56-bit storage register consisting of 74H74 and 74174 IC chips. This register is used to buffer the output of the U WORD CONTROL 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 the select the next U word. U WORD for U956:09) have a variety of mnemonics related to their control functions.
Microprogram Pointer (UPP) Register	Five 74H74 IC chips form 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. BUBC control (basic) c. EBUBC control (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.

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 chart corresponds to one processor time cycle which, in turn, corresponds to one word of the microprogram ROM.

This chapter is divided into two basic sections. The first section describes the format of the Flow Diagrams and explains the symbology and layout. The second section describes use of the flow charts.

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

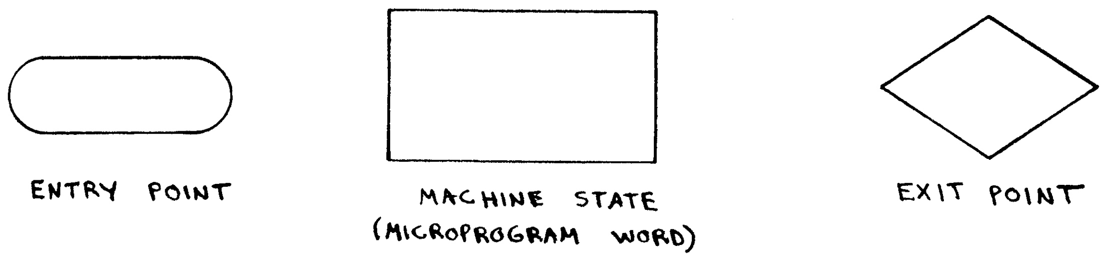

Figure 4-1 Basic Flow Diagram Symbols

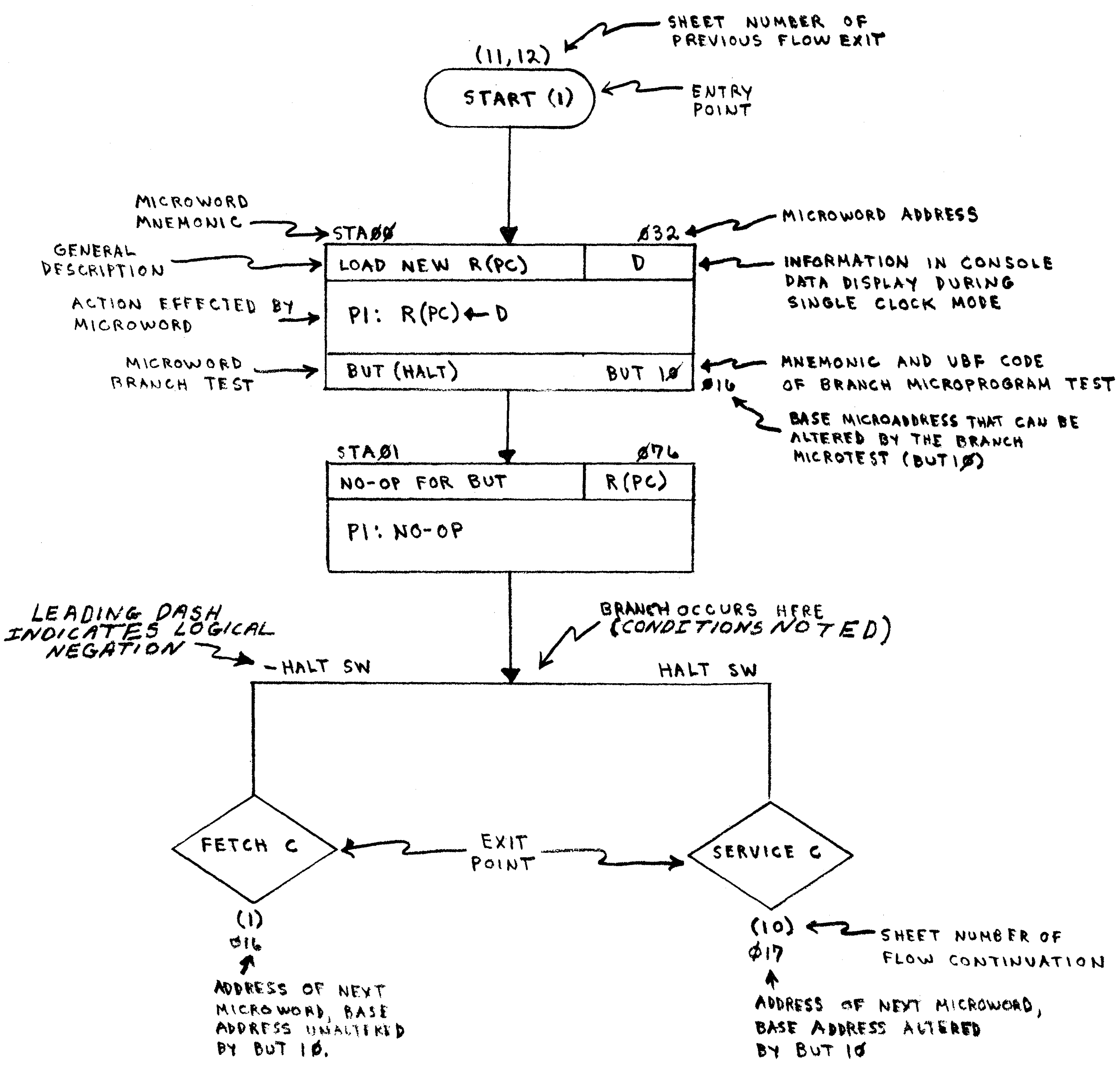
SHEET NUMBER OF PREVIOUS FLOW EXIT
(11,12)
START (1)
ENTRY POINT
MICROWORD MNEMONIC
STAØØ
Ø32
MICROWORD ADDRESS
GENERAL DESCRIPTION
LOAD NEW R(PC)
D
INFORMATION IN CONSOLE DATA DISPLAY DURING SINGLE CLOCK MODE
ACTION EFFECTED BY MICROWORD
P1: R(PC)←D
MICROWORD BRANCH TEST
BUT (HALT)
BUT 1Ø
MNEMONIC AND UBF CODE OF BRANCH MICROPROGRAM TEST
Ø16
BASE MICROADDRESS THAT CAN BE ALTERED BY THE BRANCH MICROTEST (BUT1Ø)
STAØ1
Ø76
NO-OP FOR BUT
R(PC)
P1: NO-OP
LEADING DASH INDICATES LOGICAL NEGATION
BRANCH OCCURS HERE (CONDITIONS NOTED)
-HALT SW
HALT SW
FETCH C
EXIT POINT
SERVICE C
(1)
Ø16
ADDRESS OF NEXT MICROWORD, BASE ADDRESS UNALTERED BY BUT 1Ø.
(10)
Ø17
SHEET NUMBER OF FLOW CONTINUATION
ADDRESS OF NEXT MICROWORD, BASE ADDRESS ALTERED BY BUT 1Ø

Figure 4-2 Flow Diagram Example

4.2.1 Entry Point

As shown on Figure 4-2, the entry point is labled START (1). This indicates that the section of the flow beginning at this point is activated when the console START flip-flop is sensed (1). The numbersin parenthesis 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 on 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 microword's function is to load a new value into the program counter register. This general description is provided in addition to the more detailed description of the microword operation, which is contained in the main body of the block.

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 shown on Figure 4-2, it states: P1: R(PC)$\leftarrow$D. This means that D register is being placed ($\leftarrow$) 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 microprogram branch 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 <u>not</u> the next word, but the word <u>after</u> the <u>next</u> word.

The purpose of the BUT(HALT) branch test is to determine if the HALT/ENABLE 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/ENABLE switch is set to HALT, then HALT is true and the flow exits at SERVICE C exit point. If the HALT/ENABLE switch is set to ENABLE, 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 or diamonds containing the name of the next entry point for the flow. The number in parenthesis beneath the diamond indicates the print of the Flow Diagrams containing the entry point. For example, on Figure 4-2, one of the possible exit points is FETCH C. The parenthesis (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 (entry) 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 upon the success of the branch, Microtest 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 Microtest (BUT) Instructions

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 a specific sequence is followed according to the requirements of the instruction that has been fetched.

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 ENABLE/HALT switch is sensed and the branch is effected 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 (labled 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 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 occurrs 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.

Whenever 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.

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 labled NOWR, WORDWR, and BYTEWR. However, the note on the flow diagram provides additional information that states 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 states 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 only occur after the <u>next</u> machine state, it is necessary to add a non-operational state after the BUT instruction. This is the purpose of a NO-OP FOR BUT.

Some of the notes on the flow diagrams refer to a "working BUT". A working BUT is a BUT that 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 sequentially 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 which is BUT (CBR1).

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 is devoted to explaining some of the general concepts.

In reading the 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. A statement describing the machine state operation follows the clock pulse. These statements are always read from <u>right</u> to <u>left</u>. For example:

$$P2:\ D \longleftarrow R0$$

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 contents of register R0 is loaded into the D register or D gets R00.

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

$$P1:\ B \leftarrow R(SF)$$

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 the source field. 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:

$$P2: \quad D \leftarrow f\,DAD\{R(SF)\ AND\ B\};\ DAD\ 14$$

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 read first. Thus, beginning at the semicolon and reading right to left, the statement is read: the contents of the register containing the source field and the contents of the B register are loaded into the D register as a function of DAD (discrete alteration of data); 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 fIR; 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:

$$\begin{aligned} \text{P2:}\quad & \text{BA} \longleftarrow \text{R(DF); DATI} \\ & \text{D} \longleftarrow \text{R(DF) PLUS 2} \end{aligned}$$

This indicates that three separate actions take place at clock time P2. First, the REGISTER defined by 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 destination field plus 2 is loaded into the D register.

Note that the usual use of parenthesis is to further define the preceeding 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 DF and SF) or the BA 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
```

The final item to be mentioned concerning the descriptions in the machine state boxes concerns statements that have an equal sign, such as SBC=7, DAD=10, BUS CODE=06, etc. These 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)$\leftarrow$D00; SPS=1

The above expression indicates that the value on bus data line D00 is to be loaded into the 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 4-1 and 4-2.

In the example in Table 4-1, 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 4-1 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 4-2 describes a subtract operation and is identical to Table 4-1 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 4-3 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 found.

Table 4-4 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 4-4, then Table 4-3 can be used to find the microword on the flows.

Table 4-1

Flow Diagram Example 1

Microword Mnemonic	ROM Address (PUPP)	Next 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 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	P1: 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.

Table 4-1
(continued)

Microword Mnemonic	ROM Address (PUPP)	Next 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 allow for a branch microtest (BUT).
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.

Table 4-1
(continued)

Microword Mnemonic	ROM Address (PUPP)	Next 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.
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.

Table 4-1
(continued)

Microword Mnemonic	ROM Address (PUPP)	Next Address (UPP)	Data Display	Operation	Description
DOP03	225	367	5	P2: D$\leftarrow f$ {DAD R(SOURCE) AND B} (DATOB); DAD=1; 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.
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$\leftarrow$R(PC); DATI; CLKOFF; SPS=0	Fetch of next instruction.

Table 4-2

Flow Diagram Example 2

CONDITIONS:

Address	Contents	Address	Contents
5000:	SUB #20, @#6000	6000:	30
5002:	20		
5004:	6000		
5006:	NEXT INSTRUCTION		

Microword Mnemonic	ROM Address (PUPP)	Next 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 2; 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
DST13	265	266	6000	P1: BA←R(DEST) DATIP; DAD=01; MM=01
DST14	266	277	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	5006	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 4-3
Microwords (Numbrical Order)

ROM Address	Microword Mnemonic	General Function	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, time-out 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 switch register	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
057	EXM03	Conditional plus 1	12

ROM Address	Microword Mnemonic	General Function	Print
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 switch register	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
136	FET06	Modify, store register (PC)	1
137	SRC16	Duplicate upper byte	2

ROM Address	Microword Mnemonic	General Function	Print
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
215	TRP02	Get new status	6
216	TRP04	Get MM vector (250)	6
217	FET08	New instruction from MM	1

ROM Address	Microword Mnemonic	General Function	Print
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
274	RSR05	Alter codes	9
275	RSR07	Store destination	9
276	RSR09	Duplicate byte, store	9
277	RSR10	Alter codes, finish store	9

ROM Address	Microword Mnemonic	General Function	Print
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
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

ROM Address	Microword Mnemonic	General Function	Print
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 4-4
Microwords (Alphabetical Order)

Mnemonic	Address
BRA00	111
BRA01	340
BRA02	341
CCC00	116
CCC01	350
CCC02	351
CON00	122
CON01	011
CON02	041
CON03	024
CON04	026
CON05	030
CON06	046
CON07	027
CON08	044
CON09	047
CON10	045
CON11	050
CON12	
CON13	315
CNT00	036
DEP00	034
DEP01	062
DEP02	063
DEP03	064
DEP04	065
DEP05	066
DEP06	067
DEP07	070
DEP08	071
DEP09	072
DEP10	073
DOP00	103
DOP01	364
DOP02	102
DOP03	225
DOP04	226
DOP05	227
DOP06	365
DOP07	224
DOP08	
DOP09	230
DOP10	231
DOP11	366
DOP12	367
DOP13	121
DOP14	370
DOP15	120
DOP16	371
DOP17	362
DOP18	361
DOP19	360
DOP20	375
DOP21	074
DOP22	254
DST00	161
DST01	162
DST02	164
DST03	260
DST04	163
DST05	165
DST06	167
DST07	166
DST08	
DST09	261
DST10	262
DST11	263
DST12	264
DST13	265
DST14	266
DST15	267
DST16	237
DST17	270
EXM00	035
EXM01	053
EXM02	054
EXM03	057
EXM04	056
EXM05	055
EXM06	031
EXM07	060
EXM08	061
FET00	013
FET01	000
FET02	016
FET03	001
FET04	004
FET05	005
FET06	136
FET07	100
FET08	217
FET09	252
JMP00	151
JMP01	152
JMP02	235
JMP03	154
JMP04	300
JMP05	153
JMP06	155
JMP07	157
JMP08	156
JMP09	301
JMP10	302
JMP11	303
JMP12	306
JMP13	313
JMP14	304
JMP15	305
LAD00	037
LAD01	051
LAD02	052
LAD03	033
MOV00	171
MOV01	172
MOV02	174
MOV03	173
MOV04	175
MOV05	177
MOV06	176
MOV07	257
MOV08	206
MOV09	212
MOV10	213
MOV11	207
MOV12	210
MOV13	203
MOV14	202
MOV15	205
MOV16	200
MOV17	201
MOV18	170
MOV19	160
MOV20	204
MOV21	003
MOV22	125
NBR00	110
MRK00	112
MRK01	353
MRK02	354
MRK03	355
MRK04	356
MRK05	357

Mnemonic	Address
RSR00	106
RSR01	271
RSR02	107
RSR03	272
RSR04	273
RSR05	274
RSR06	232
RSR07	275
RSR08	233
RSR09	276
RSR10	277
RSR11	376
RST00	127
RST01	025
RST02	040
RST03	043
RST04	042
RTI00	101
RTI01	320
RTI02	321
RTI03	322
RTS00	124
RTS01	323
RTS02	324
RTS03	325
SCC00	117
SCC01	352

Mnemonic	Address
SER00	114
SER01	002
SER02	017
SER03	123
SER04	006
SER05	015
SER06	012
SER07	020
SER08	021
SER09	014
SER10	022
SER11	023
SOB00	130
SOB01	342
SOB02	343
SOB03	344
SOB04	346
SOB05	345
SOB06	347
SRC00	141
SRC01	142
SRC02	144
SRC03	240
SRC04	143
SRC05	145
SRC06	146
SRC07	241
SRC08	242
SRC09	147
SRC10	243
SRC11	244
SRC12	245
SRC13	246
SRC14	247
SRC15	250
SRC16	137
SRC17	251

Mnemonic	Address
SSL00	105
SSL01	372
SSL02	104
SSL03	373
SSL04	221
SSL05	236
SSL06	220
SSL07	223
SSL08	222
SSL09	374
SSL10	211
SSL11	075
SSL12	253
STA00	032
STA01	076
SWB00	134
SWB01	135
SXT00	132

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

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 should 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 usedin 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 them named 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 debug 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	KF11-F Floating Instruction Set

Sheet information for each print prefix occurs as a dash (-) 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 or L. These level indicators are H for high and L for low and 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 labled 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 CIN00 L 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 upon 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 noted before, 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 <u>Parentheses and Colons</u>

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 of signals of like nature that appear singularly on the prints.

5.2.1.8 <u>Parentheses and Commas</u>

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 <u>Basic and Expansion Signals</u>

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 the 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 upon the etch boards. Special attention must be given to shield location as noted in the prints and upon 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 upon the logic prints.

5.2.2.2 Print References

Print references are noted in the "DRAW" column for all prints upon 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 Foward 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 path module includes the following logic:

a. The Arithmetic Logic Unit (ALU) with an A input and a B input with multiplexer (BMUX) 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 upon 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 lower order switch register (SR) settings and the buffered microprogram pointer (BUPP) to determine when the microprogram matches the console switch settings.

K1-2 Print: DATA PATHS (03:00)

This print contains the data path for the data bits 03 to 00. It has the arithmetic logic unit and its associated A and B input logic as well as the output D register and Bus Addresses Multiplexer (BA MUX).

K1-2 DMUX(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 arithmetic logic unit and to each of the other processor registers including the scratch pad (REG), and the processor status (PS). These signals also are available on the back panel and used in processor options (KE11-E, KE11-F, KT11-D).

The following inputs are combined or multiplexed into the DMUX signal: the D register at the output of the arithmetic logic unit; the buffered UNIBUS BUS D signals; a right shifted output of the D register, and the buffered BUS RD signals. The later signals (BUS RD) are from the outputs of the various processor registers located in the data path section of the machine and include Scratch Pad Register (REG), Instruction Register (IR), and Processor Status (PS), as well as other processor option registers. The DMUX signals are displayed in DATA display of the console.

K1-2 COUT03 L signal is the carry out of the third bit of the arithmetic logic unit. 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 7481 by looking ahead to determine if carries exist.

K1-2 D(03:00) (1) H output the D register as noted, inputs to driver 8881 gates to the UNIBUS, and feeds around to the DMUX 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 simultaneous read and write 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 driver (8881s) with a 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 up and is not used as a signal at all. 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 ALU00 H low order output signal of the arithmetic unit is used in the KE11-E option.

K1-2 BAMUX(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 upon bus operation, with the address coming from the Scratch Pad Register (REG). The ALU input accomodates those situations in which data is altered before use.

K1-3 Print: DATA PATHS (07:04)

K1-3 DMUX(07:04)H
K1-3 D(07:04) (1) H
BUS D(07:04) L
K1-3 BAMUX(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 BMUX07 H signal is the high order bit of the byte input to the B input of the arithmetic logic unit. 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 arithmetic logic unit prior to the B register. It is provided for test purposes.

K1-4 Print: DATA PATHS (11:08)

K1-4 DMUX(11:08) H K1-4 D(11:08) (1) H BUS D(11:08) L K1-4 BAMUX(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 Print: DATA PATHS (15:12)

K1-5 DMUX(15:12) H K1-5 D(15:12) (1) H BUS D(15:12) L K1-5 BAMUX(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 input 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 DMUX and BAMUX as all of the buffered BUS RD signals are. It is also used as an input to the condition code logic for determination of word overflow conditions (K5-2 Print).

K1-5 CMUX15 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 (K5-2 print).

K1-5 ALU15 H is the output of the bit 15 of the arithmetic logic unit and is used as an input to the D register, the BAMUX, 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 BMUX on this print and the SWAP BYTE input on K1-3 print. The KE11-E and KE11-F options also utilize this signal.

K1-6 Print: BA(15:00)

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 bit 13, 14, and 15 for the basic KD11-A. The Unibus drivers on the print can be disabled when the KT11-D option is installed; then the bus address is 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 (K1-9 print) and in 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.

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 (K1-9 print); for data display (K5-7 print); and for generation of Unibus addresses for the KT11-D option (if installed). Some of the signals are used in 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 for odd byte address sensing and allowance of odd byte on K4-4 print and for byte branching information for the BUBC codes (K3-7 print).

K1-7 Print: ADRS DECODE

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, 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 that 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 (K5-2 print) and sequencing of a BUS SSYN response by the processor (K4-6 print). 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 (K5-7 print). 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 installed, the signal provides selection of that register and the sequencing of a BUS SSYN through logic on K4-6 print. 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 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 (K3-2 print). 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 of K1-8 print under microprogram control.

K1-7 SR ADRS H provides Switch Register Address decoding and allows derivation 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 logic on K4-6 print.

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 in 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 specifically an address situation. This output is for test purposes.

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

K1-7 BOVFL L signal is used to sense a yellow zone stack violation and is used to set a trap servicing flag 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 (K5-2 print); it is also used as an input to the microbranch logic (K3-2 print). 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 (K5-2 prints).

K1-8 Print: REG(15:00)(15:00)

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 effected by jumpers (W5, W6, and W9) when the KT11-D option is installed. The Registers themselves take data in from the DMUX signals and outputs its data onto the BUS RD lines. The resistor terminator for this wired-OR BUS RD bus are resident on this print.

BUS RD(15:00) L common input bus in the data paths has several sources. Its input is on prints K1-2 through K1-5, respectively, 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 noted below.

K1-8 RADRS(3:0) L. These signals are the actual applied signals to the Scratch Pad register ICs. They are the complement of the address inputted to the selection AND/NOR gates. The signal K1-8 RADRS 0 L is used in conjuction with K1-8 R (X6+X7) H to specifically indicate the stack processor stack register (REG 06) to enable the check overflow logic in the bus timing circuitry (K4-4 print).

K1-9 Print: CONSOLE AND MATCH

This print provides a connector interface to the KY11-D console. The DATA display is provided to the console with 380 type gates buffering the DMUX (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 8881 gates to the Unibus BUS D(15:00). A matching circuit is provided with the lower order switch register settings, SR(08:00), compared with the basic microprogram pointer BUPP signals. Upon a match, a pulse, K1-9 P MATCH L is generated, K1-8. 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 upon match (see restrictions below).

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

BUS D(15:00) L Unibus signals are enabled by K4-4 BUS FR 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. There is a limitation that the matching address must have a CL2 or CL3 preceding it.

K1-9 P MATCH L - A timing pulse occurring at the end of a machine cycle is gated against the aforementioned match signal to provide a scope triggering pulse at the beginning of the selected word. This occurs independent of the maintenance module and is of value in situations where the machine is not going to 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 buffer register (various nuemonics per function), the Past MicroProgram Pointer register (PUPP), as well as certain driving buffering gates for signals BUPP(8:0) and EUPP(8:0). 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 buffer register which controls the machine. The first segment of the ROM(07:00) on prints K2-2,3 is concerned with the microaddress and has 74H10 gates between the ROM output and the U WORD buffer (UPP 08:00). This allows the next base address, BUS U 08:00, to be modified, if necessary, by basic microbranching logic (K3-2 BUBC 4:0) or expanded microbranching (KE-4 EUBC 4:1). Additional logic exists on the output of the UPP portion of the U WORD for driving expansion ROM's 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 expresses the physical wired-OR nature of these signals. No absolute correlation should be made between low active and the trueness of the ROM output or the U WORD buffer. In fact, for U (56:09) a high level indicates a true or active signal both at the ROM output and in the U WORD buffer. 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 buffer, UPP (08:00). In the U WORD buffer a high level represents an active or true signal. The microflow diagrams on sheets 9 through 12 show the complement of the ROM output for the UPF field; this is to allow 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 no microbranch inputs, then the UPF field represents the U WORD buffer , UPP (08:00).

The output signals noted in detail for this module are mostly those of the U WORD buffer. These signals, with the alterations possible to the UPF field, are directly compatable to the BASIC U WORD noted in the block diagram of the U WORD and 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.

Signals for prints K2-4,5,6,7,8 are presented in order from bottom to top. This is in order of assending BUS U allocation, and represents a logical presentation of the functions.

PRINT K2-2, U(03:00)

K2-2 BUPP(3:0) H signals are the Buffer Micro Programmed Pointer for the noted bits and it is used to address expansion ROMs in the KT and KE11-F options. It is also displayed on the KM11-A Maintenance Console, and it is matched against the switch register for a HALT or for scope timing pulses (K1-2 print). Note that the microprogram address present here is the address of the next microword. Alterations for microbranching have already occured on the input to the UPP register if they were to occur. Only a microjam (CLK JAM on print K4) can alter this address by setting or clearing the UPP register, this change is then reflected in these signals.

K2-2 PUPP(03:00) (1) H signals are the output of the Past Micro Program register. It provides the microprogram address of the microword presently in the U WORD buffer and acting upon the machine. 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 as 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 labled for reference on print K2-3 without the need of showing the pull-up resistors again.

PRINT K2-3, U(07:04)

Signals K2-3 BUPP(8:4) H and K2-3 PUPP(8:4) (1) H are similar to signals on the K2-2 print with the exception of bit references.

PRINT K2-4 U(16:09)

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

Signal K2-4 RIFO (0) H is used on the address input to the Scratch Pad Register, for the special situation where the KT11-D option is providing the user stack pointer instead of the usual REG 06 stack pointer. The RIF notation is discussed immediately below.

Signals K2-4 RIF (3:0) (1) H are for Register Immediate Field and provide direct microprogram selection of the 16 Scratch Pad Registers (REG) when the Select Register Immediate microcontrol is enabled (see below). Direct micro code selection of the Scratch Pad Registers 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 . REG 0 is selected during the HALT instruction and in console operations. In addition to the selection of the program scratch pad registers, REG(07:00), throughout the microprogram implementation of instructions and trap sequences, the registers REG (17:10) are also selected explicitly.Those upper Scratch Pad Registers provide intermediate storage which is of use in flow implementation and in maintenance. Other calculations interior to an instruction are stored and can be examined in Single Instruction operation. The existance of the RIF field makes use of these registers pratical.

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

Signal K2-4 SRI (1) H is Select Register Immediate and used on 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).

Signal K2-4 SRBA (1) H is Selects Register Bus Address and it 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) to indicate an internal register addressed by the Bus Address Register.

Signal K2-4 SRD (1) H is Select Register Destination and is used in the Scratch Pad address logic to enable the Destination field of the Instruction Register IR(02:00). This field is used throughout various instructions having destination addresses.

Signal K2-4 SRS (1) H is Select Register Source and it 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 emphasis on reducing access time for data.

PRINT K2-5 U(28:17)

Signal K2-5 CLR U(56:17) L is a nominally high signal provided by a pull-up resistor which is used for the clear line of the U WORD buffer on this print and on prints K2-6,7,8.

Signals K2-5 UBF(4:0) (1) H are the Micro Branch Field which enable various test conditions for microbranching the microprogram address. The actual switching of various conditions is done on the K3-2 print in multiplexers provided for that purpose. In all, 32 possible test microbranch tests are enabled throughout the microflows and are directly noted in the flow diagrams by the BUT notation (Branch Micro Test) and in the table on print D-BD-KD11-A-BD. The tests that are enabled can consist of a number of bits or a single bit. They can consist of tests relating to the instruction register or tests relating to a single flag flip-flop.

The UBF field is also decoded on the Status Board K5-3 to provide enabling signals during certain microwords; the microwords in which this field of a specific BUT is present. Those decoded signals are used to clear or set flags relating to the microtests upon which a Branch Micro Test is being performed.

Signal K2-5 SBAM (1) H is for Select Bus Address Multiplexer 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 arithmetic logic unit for input to the BA Address Register. A high level in this micro code control bit enables the ALU output to the Bus Address Register Data input.

Signals K2-5 SDM(1:0) (1) H are Select D Multiplexer and are used in the Data Path (K1-2,3,4,5) to select the DUMUX 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 PATH prints. Essentially a 00 SDM code selects the A input (BUS RD); a 01 SDM code selects C input (D register), a 10 code selects the buffer Unibus data (BUS D),and a 11 SDM code selects D input (right shifted D register).

Signals K2-5 SBML(1:0) (1) H are Select B Multiplexer Low microcontrol and provide selection signals to the lower eight bits of the BMUX in the Data Paths on print K1-2,3. The separation of the BMUX into an upper and lower portion for micro control allows additional flexibility in treating with byte, sign extend, or swap byte situations. The code enables the following to the B Input of the Arithmetic Logic Unit (ALU):

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

An SBML code of 01 selects the low bits of the B Register directly and is used for sign extension in the upper byte (BMUX 15:08)

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

An SBML code of 11 selects the B Constants, BC(07:00), as inputs to the ALU.

Signals K2-5 SBMH (1:0) (1) H are Select B Multiplexer High and provide selection signals to the upper eight bits of the BMUX in the Data Paths on print K1-4,5. The code enables the following to the B Input of the Arithmetic logic.

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

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

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

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

PRINT K2-6, U(40,29)

Signals K2-6 SBC(3:0) (1) H are Select B Constants and and they provide selection through combinational logic (print K5-5) of a potential 16 constants for use by the B multiplexer. The encoding of constants selection is utilized to conserve micro control storage (ROM) bits. A table of the constants is provided on print K5-5 or the Block Diagram Print D-BD-KD11-A-BD.

Signals K2-6 SALU(3:0) (1) H are the Select Arithmetic Logic Unit which provide direct micro code selection of the functions that the arithmetic logic unit will perform. These signals are selected by logic on print K3-8 for direct ALU control unless a DAD micro code 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.

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

Signals K2-6 SPS(2:0) (1) H are Select Processor Status and they provide an encoded combination for various functions on the Processor Status. These operations on the Processor Status 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 while there is also a total decoding of these bits to perform other functions. The code is used in logic on print K5-2 to directly select the inputs to the Processor Status Register. It is also used on print K3-9 to effect the condition code inputs.

PRINT K2-7,U(52:41)

Signals K2-7 DAD(3:0) (1) H are Discrete Alteration of Data and they provide an encoded portion of the microword for use throughout the machine in allowing exceptions. It is used with Unibus cycles to check for odd address or stack overflow; it is used in the Arithmetic Logic Unit logic to allow alteration of the code as a function of the Instruction Register. Discrete Alteration Data code is also used within the console loop for setting and clearing EXAMINE and DEPOSIT 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.

Signal K2-7 BGBUS (1) H is Begin Bus and it 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 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 on print K4-5 is used to clock the NPR signals.

Signals K2-7 C(1:0) BUS (1) H consist of the C1, C0 BUS signals usual to the Unibus. These control signals are clocked into holding flip-flops on print K4-4 for use throughout the bus cycle.

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

Signal K2-7 CLKD (1) H is Clock D and it is used on print K4-2 to enable pulses for clocking the D Register.

Signal CLKB (1) H is the Clock B and it is used on print K4-2 to enable pulses for clocking the B Register.

Signal K2-7 WRL (1) H is Write Low used on print K4-2 to enable a write signal to the Scratch Pad Register for the low order byte.

Signal K2-7 WRH (1) H is Write High used on print K4-2 to enable pulses to write into the high bit of the Scratch Pad Register.

PRINT K2-8, U(56:53) AND CONNECTORS

Signal K2-8 CLKIR (1) H is Clock IR and it is used on print K4-2 to enable pulses for clocking the Instruction Register. It is enabled only during the FETCH cycle.

Signal K2-8 CLKOFF (1) H is Clock OFF and is used 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.

Signals K2-8 CLKL(1:0) (1) H are the Clock Length code to provide selection of the cycle lengths used in the current microword. This signal (print K4-2) directly interacts with the pulse stream within the delay line chains. A CLKL code of 00 or 01 provides for a Clock Length 1 (CL1). If the CLKL code 00 is used, a special overlap situation may be in effect. A CLKL code of 10 effects a Cycle Length 2 (CL2), a CLKL code of 11 effects a Cycle Length 3 (CL3). The normal duration of these respective cycles are:

CL1	140 nanoseconds
CL2	200 nanoseconds
CL3	300 nanoseconds

Signals K2-8 CLKL(1:0) (0) H are the complement of the Clock Length code and are used to assure 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 is provided. The BUS signals noted as inputs (signals to the right of the connector) are wire ORed throughout the module to the basic ROM's output. EUPP(7:0) output signals (to the left of the connector) provide the address for the expansion ROM.

U WORD MICROPROGRAM LISTING

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

The U WORD Micprogram Listing presents the Read Only Memory (ROM) content of the M7232 module, and of the KD11-A basic 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 ADdRess column (ADR). 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 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)	Micro program Pointer Field represents the next microword address (base) 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 (UPP8:0) 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 that next ROM word enters the U WORD Buffer Register facilitates comparison of the U WORD and the Microprogram Listing. In single clock mode of the Maintenance Console option (KM11-A), the PUPP address can be used in the ADR column to find the presently controling microword. The Microprogram Listing can also be correlated with the Flow Diagram from its microword address.

NOTE

With the exception of the UPF field (noted above), 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)	Micro Branch 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(4:0). A correlation is made between the Branch Micro Test (BUT) number and its purpose on print D-BD-KD11-A-BD.
SBA U22	Select Bus Address directly controls the Bus Address Multiplex on the input to the Bus Address Register. When enabled, the U WORD Buffer signal, SBAM, selects the ALU output instead of the BUSRD signal output.
SDM U(24:23)	Select D Multiplexer directly controls the selection of inputs on the D Multiplexer (DMUX). Its octal code 0, 1, 2, and 3 correlates 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 Multiplex (BMUX).

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 a table on print D-BD-KD11-A-BD.
ALU U(37:33)	Arithmetic Logic Unit controls the mode of operation of the Arithmetic Logic Unit (ALU) of the Data Paths. The code is not used directly and does allow the Discrete Alteration of Data (DAD) microcode to provide ALU operation 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 micro control 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 an encoded microcode field that provides for the alteration of usual usages of data (including microcode data). A usual alteration is the checking for odd address or stack limit 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 micro control 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 (CLKL0, CLKL1) and on-off control (CLKOFF).

Three other columns occur in the Microprogram Listing, they 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 Maintenance Console option, KM11-A.
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 reencoded microaddress information necessary for the microbranches. The Instruction Register is present on this module, as is the Branch Micro Tests (BUT) multiplexer. In addition, combinational logic exists for instruction control of the Arithmetic Logic Unit and Condition Code inputs for the Carry (C) and oVerflow (V) bits of Processor Status.

K3-2 Print: BUT MUX

This print contains the Branch Micro Test multiplexer which combines diverse micro branch 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 then named in a standard way, such as K3-5, BUBC0(BUT37) H. This signal is essentially a Basic Micro Branch Code that will effect the 0 bit of the microaddress for the Branch Micro Test 37. There is a table of these branch microtests and their mnemonics on the Block Diagram Print (D-BD-KD11-A-BD). BUT 37 is the INSTR1 branch, occurring in Fetch and branches to all the various response micro flows for instruction implementation. It has an input to each of the five address bits that are effected. Other branch micro tests 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 the two possible inputs in the NOT-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 micro branch field (UBF) field for branch micro tests 37 through 34 inclusively. It is a single pin run and is provided for test purposes.

K3-2 BUT (3X) - signal is a decoding of the Micro Branch Field (UBF), 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 micro test for BUT 30 through 37 octally.

K3-2 BUBC(5:1) L - signals represent the Basic Micro Branch Code for the address bits 5 through 1, inclusively. They each represent a single input to the NOT-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 BUTs and higher order BUTs. Selection of these inputs is a function of the micro branch field from the U WORD applied against the appropriate multiplexers. In conjunction with the basic micro branch code, there are expansion micro branch code bits also inputted to the NOT-OR Gate.

K3-2 BUBC0(BUT37:20) L - provides Basic Micro Branch Code 0 for branch micro test 37 through 20 inclusively (the notation is octal). It is used in conjunction with the next signal to the exclusion of the expansion micro branch code for this bit. It is used on K2-2 print in the NOT-OR Gate.

K3-2 BUBC0(BUT17:00) L - provides the Basic Micro Branch Code for bit 0 for the branch micro tests (17:00), inclusively. The signal is selected as a function of the multiplexer and the UBF field in the U WORD, with the UBF field selecting the branch micro test being applied against the base address. This bit 0 has many test conditions applied against it, not only in the complex codes but the single bit codes.

K3-3 Print: IR AND DECODE

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

K3-3 IR(15:00) (1)H - is the (1) side of the IR Register 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 for 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 } are partial decoding of the IR (bit 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 and on the STATUS board (K5-3) for branch instruction decoding and upon the KT11-D option on 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 inclusively is 6; it is utilized 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 1 - is a partial decoding of the IR for a Destination Mode 0 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 Register.

K3-4 Print: IRD & OVLAP

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 inclusively, equal to 6 octally which is utilized by the KT11-D option.

K3-4 OVLAP CYCLE L - OVerLAP CYCLE includes the next signal OVerLAP INSTRuction as well as additional situations. An OVerLAP CYCLE is based upon the same premise as an OVerLAP INSTRuction; that is, 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 instruction occurs are usually Single Operands with Destination Mode zero, or Double Operands with both Source and Destination Modes zero. Exceptions to this are that the destination register cannot be REG 07; the program counter which is being used as the next address cannot 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 reentry into the Fetch micro flow.

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

K3-5 Print: BUBC(INSTR1)

This signal is Basic Micro Branch 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 these instruction signals from this sheet and instructions from oncoming sheets are reencoded into basic micro branch code (BUBC) for the first instruction branch. This instruction branch is known as INSTR1 for BUT 37 and appears on sheet 1 of the Flow Diagram (D-FD-KD11-A-FD).

K3-5 BUBC(5:0)(BUT 37) H - is the Basic Micro Branch Code for microaddress bits 5 through 0 inclusively and is activated upon the INSTRuction 1 branch test for BUT 37. It is decoded from the IR and available on the input to the multiplexer. The multiplexer itself on print K3-2 provides the selection for BUT 37 and this code is enabled over the base microaddress for this test. This branching code is especially critical and basic to the machine, as it is the first instruction branch in Fetch.

K3-5 DOP*-SM0 L - is a Double Operand instruction and Source Mode zero encoding together and provided for use within the IR board.

K3-6 Print: IR DISCRETE

Combinational logic upon 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.

Little information would be presented by listing these instructions and explaining that they occur when their certain IR code exists. Suffice to say, that 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 that the inhibit of the discrete HALT and WAIT signals by 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 reencoding from the decoded instruction Register.

K3-7 Print: BUBC (OTHER)

Located on this print are various Basic Micro Branch Code (BUBC) for tests OTHER than INSTR1 consisting of different numbers of address bit inputs for different Branch Micro Tests (BUTs). The ones that are shown on the extreme right have no greater importance than the ones shown on the left or midway. Essentially BUBC codes for BUTs 20,21,25,26,27, 31,33,34,35, and 36. There are also some additional instruction register type of signals such as, SERVICE, TRACE, and BYTE CODES. Signals within the print, as well as those on the extreme right, are of importance in this print. The majority of signals (BUBC codes) are used on print K3-2 as inputs to the multiplexers. A table of BUTs used exists on the BLOCK DIAGRAM, U WORD & TABLES print (D-BD-KD11-A-BD).

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 Trace Trap occurs after the next instruction if an RTT instruction is present. The signal is used on this print in the BUBC1(BUT26) signal and on STATUS (print K5-4,5) for flag control and trap vector generation.

K3-7 SERVICE H - is a definitive definition of the reasons to enter the Service section of the micro flows 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, reference to Processor Status ADdRess 5) situations requireing service. The signal is used on this print in the BUBC1(BUT20) signal for microbranching and 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.

K3-7 BUBC(5,3,0)(BUT36) H - is the Basic Micro Branch Code for microaddress bits 5, 3, and 0 for the Branch Micro Test 36. BUT 36 is the INSTRuction 3 branch associated with the next flow sequences after SOURCE calculations.

K3-7 BUBC(5,3:0)(BUT35) H - is the Basic Micro Branch Code for microaddress bits 5, bits 3 through 0 for the Branch Micro Test 35. BUT 35 is the Odd Byte and INSTRuction 3 branch associated with byte formatting of incoming data or the next flow sequences after SOURCE calculations.

K3-7 BUBC(3:0)(BUT34) H - is the Basic Micro Branch Code for microaddress bits 3 through 0 for the Branch Micro Test 34. BUT 34 is the INSTRuction 4 branch associated with the next flow sequences after DESTination calculations.

K3-7 BUBC(3:0)(BUT33) H - is the Basic Micro Branch Code for microaddress bits 3 through 0 for the Branch Micro Test 33. BUT 33 is the Odd Byte and INSTRuction 4 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 one 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(1:0)(BUT20) H - is the Basic Micro Branch Code for microaddress bits 1 and 0 for the Branch Micro Test 20. BUT20 is the Byte or Service or Fetch branch associated with the end of instruction execution.

K3-7 BUBC0(BUT31) H - is the Basic Micro Branch Code for microaddress bit 0 for the Branch Micro Test 31. BUT 31 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 Micro Branch Code for microaddress bit 0 for the Branch Micro Test 27. 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(1:0)(BUT26) H - is the Basic Micro Branch Code for microaddress bits 1 and 0 for the Branch Micro Test 26. BUT26 is the Request branch associated with the entry into the SERVICE flow and provides for the proper sequence and service of requests.

K3-7 BUBC(1:0)(BUT25) H - is the Basic Micro Branch Code for microaddress bits 1 and 0 for the Branch Micro Test 25. 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(1:0)(BUT21) H - is the Basic Micro Branch Code for microaddress bits 1 and 0 for the Branch Micro Test 21. BUT21 is the IR03 and Byte or Source branch associated with index address operations in the MOV address calculations.

K3-8 Print: ALU CONTROL

This print has two sets of combinational logic. One set is ordered toward the Arithmetic Logic Unit 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.

K3-8 COMUXS(1:0) 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 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 Arithmetic Logic Unit (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 IR decode. It is used with the IR DECODE module and upon TIMING (print K4-4) to alter DATIP bus cycles to DATI bus cycles for DESTination data references.

K3-8 ALUS(3:0) H - are the direct control for the Arithmetic Logic Unit 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 Arithmetic Logic Unit 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.

K3-9 Print: CODES C,V

This print contains 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 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 (DMUX01) 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 (DMUX00) 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.

Print K4-2: CLOCK

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 logics. Necessary peripheral logic provides on-off control (IDLE flip-flop), asynchronous restart inputs, and output enabling gates.

Assuming sequential, uninterrupted operation, the end of the last clock cycle is the beginning of the next clock cycle. The falling edge of the K4-2 RECLK H signal clocks a one to the CLK flip-flop (assuming continuous operation) which activates the pulse forming logic loop with Delay Line 1 (DL1). After delay, the DL1 loop will clear the CLK flip-flop. The CLK flip-flop, therefore, forms a pulse of approximately 40 nanoseconds (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 CLK1 (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 K42Z 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 (Delay Line 4). 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 segments of the KD11-A processor: the interface, the data path, and the microcontrol. The microcontrol clocking signals (CLK U signals, RECLK, PEND and PART P END) are ordered toward end of cycle pulses. For a CL1 this is P1 pulse; for a CL2 this is P2 pulse; and for a CL3 this is P3 pulse. Clocking to the U WORD and the clock logic is not conditioned by any enabling signal and is usual 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 one 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 one state; the CLK flip-flop is not clocked to the one state and no pulse forming occurs. Conditions to halt the clock are noted upon the inputs to the 74H53 gate at E77; the most usual 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 insure 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. Usual to each of these restart inputs is the enabling conditions for the restart condition and the restart signal.

An additional control flip-flop, MCLK, is provided for singular, manual operation. This flip-flop function in parallel with the CLK flip-flop to generate 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 effectively disabled with neither its data or set inputs enabled. Details of Maintenance Console interaction are available in Paragraph 7.3 of this manual.

Print K4-3: CLK JAM

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 the restart signals. An interruption (or pause) has occurred in the microflow, but sequential flow still occurs after restart.

The CLK JAM logic is ordered toward non-sequential interruptions of the microflow. 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 microflow to specific microaddresses. The microflow is irrevocably JAMmed to a specific operating flow. The JAMUPP ADDRESS table on this print correlates the reasons (USE) for the microjam and the new microaddresses (UPP).

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

The flag flip-flops which are clocked to the one state for activation are JBERR flag for odd address bus errors and red zone stock overflow, and JPUP flag for START switch activation in the HALT mode and PoWeR RESTART. Both of these flags, with addition 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 Micro Program 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 (= 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 (of print K4-6) if appropriate and restarts the CLK logic.

Print K4-4: BUS DATA CNTL

Logic on this print is associated with processor Unibus data transfers and the variety of required error checking and cycle alteration. 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 UWORD and IR decode inputs. 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 5-4). Tables are provided for the BUS and DAD fields of the microword.

The logic discussion is ordered toward the control flip-flops and their overall effect.

BUS Flip-Flop - The BUS flip-flop initiates all processor Unibus cycles. It is clocked to the one state by K4-2 CLK BUS H signal (derived from BG BUS of U WORD) except for DAD code (1X1X) in the Execution flow of the BIT or CMP or TST instruction and the non-existance of an OVer LAP 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 MSYN A flip-flop. (The MSYN A signal is used for internally mounted MM11-L memories that have fast 7380's Unibus receiver gates.)

Time exists during the deskewing delay, for error conditions to zero the data inputs of the MSYN and MSYN A flip-flops. Normally, however, the flip-flops are clocked to the one state and through appropriate gates drive the Unibus (or specially connected twisted pair wired to MM11-L memories). Disabling exists for the KT11-D option. The MSYN, MSYN A and BUS flip-flops are pulsed cleared from the BWAIT flip-flop.

CKOVF Flip-Flop - The CKOVF flip-flop controls the ChecK of OVerFlow upon the processor Stock Pointer. Only certain address modes in certain bus operations need to be checked. This is controlled by the DAD code (X11X) of the UWORD with REGister selection information a disabling flag (K5-4 STALL (↑) 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 is further conditioned by the KT11-D option and the Unibus cycle type. The check enabling signal enables error detection signals and provides for possible abortion of the Unibus cycle with corresponding raising of error flags. This occurs here only for red zone stock 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 one state by the K4-2 CLK BA H signal unless a Byte Instruction exists with a DAD code (XXX1) from the UWORD. 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 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 the Bus WAIT 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 zero state. Other clearing signals are combined in the pulse logic to accommodate situations where no peripheral response is made (NODAT error, microcontrol JAMUPP other bus errors) and processor INITializing.

BC1 and BC0 Flip-Flops - The Unibus Control signals are held in the BC1 and BC0 flip-flop. The flip-flops are loaded from C1BUS and C0BUS bits of UWORD by the K4-2 CLK BUS H signal (derived from BF BUS of UWORD). Modification of the data input for BC0 is made for Byte Instructions (to change DATO operation to DATOB) and BIT or CMP or TST Instructions (to change DATIP operation to DATIP). 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.

Print K4-5: BUS OWNERSHIP

Provided 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 one state, and is necessary upon 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 processor normally exerts bus ownership.

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 Non-Processor Requests; the BRPTR flag for Bus Request with Priority Request greater than Processor Status priority; and CBR flag for the 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 (7:4) H) on the Unibus (see 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 for the priority determination logic.

The K4-5 CLK NPR H signal also has clock inputs for Clock restart (K4-2 SET CLK L), data bus cycles beginnings (K2-7 BG BUS (1) H), BUT26 in Service flow, the deactivation of MSYN (K4-5 P MSYN H pulse) and CLK IR for OVerLAP situations. Independent of clocking the data input to the NPR flag provides zero data for power fail (K5-8 B AC LO L) and across DATIP operation. A DATIP flag flip-flop prevents the granting of bus control for Non-Processor Requests as the DATIP address location is still selected by the processor with the probability of a partial read/restore cycle in the peripheral.

Clocking for the K4-5 CLK PTRD H also occurs for BUT26 in Service flow and for 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 Requests 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.

Print K4-6: BUS RESPONSE

Three types of BUS RESPONSE are provided by 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 time-out flags for NO SACK and NODAT.

The Bus Grant signals (BUS BG (7:4) 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 OWERSTP logic of print K4-5 to determine if the BRPTR flag should be enabled. When enabled, processor service of the flag results in the K4-5 GRANT BR H enabling signal to activate one of the BUS BG (7:4) H signals on the Unibus.

Processor register response to absolute Bus Addresses is not completely specified by microprogram control. Bus Address decoding (K1-7 print) and Unibus Control decoding (K4-4 print) 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 for example). 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 DMUX signals. The Scratch Pad Register (REG) does not respond to processor and console Bus Address references; it responds to console references and then under microprogram control.

The time-out flags for NO SACK and NODATA provide a processor response when peripherals fail to respond. The NO SACK flag is set when peripherals granted bus ownership fail to respond; the NO DAT flag is set when data bus operation receive no SSYN response. In each case the time out duration is 15 microseconds. The service of the time-out flags differs. The NODAT flag results in microprogram interruption (JAMUPP) and a trap sequence. The NO SACK flag merely allows the processor to regain bus ownership and continue operation.
Each time out may be disabled for maintenance operation. See the note on the print or details of Maintenance Module operation (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:

Processor Status word with Priority bits for comparison to Bus Request, a Trace bit, and Condition codes N,Z,V, and C.

Branch Instruction implementation with comparison of the Condition Codes with IR decoding.

Branch Micro Test (BUT) Decoding with discrete outputs as a function of specific microwords.

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

B Constants decoding with Special Trap Markers (STPM) signals for Trap vectors.

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

Console Interface for the ADDRESS display and control inputs.

Power Fail and Bus one-shots for proper sequence of bus signals.

K5-2 Print: PS(07:00)

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 agains 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 DMUX 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 the Processor Handbook in detail. 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 the bits PS(N), PS(Z), PS(V), and PS(C). These bits are loaded from the DMUX upon a specific LOAD Processor Status clock from the processor, in addition to conditional inputs as a function of instruction operation 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 past A input (to the ALU) and past C (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(2:0)(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 DMUX(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 upon 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 microcontrol in instruction or trap operations. Input data for Condition Code operation comes from combinations of logic which selects 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 (K5-3 print).

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 (K5-3 print).

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 operations. Input data for Condition Code operation is provided by K3-9 V DATA L from the IR DECODE module. The 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 signal is used in the branch instruction logic (print K5-3), on the input multiplexer for D(C) (print K1-5), and in combinational logic for generation of signals K3-8 CIN00 L, K3-9 VDATA L, and K3-9 CDATA 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.

K5-2 LOAD PS L - is the enabling signal for the combinational logic on the data inputs of the Condition Codes to allow the DMUX data signals instead of Condition Codes 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. The 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 in the KT11-D option.

K5-3 Print: BUT & BRANCH

Two distinct sets of combinational logic exist on this print: Branch instruction logic for comparison of instruction decoding with Condition Codes; and Branch Micro Test (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 (K3-5 print) to 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 Micro Branch Field (UBF) of the U WORD. These decoded signals are used throughout the processor as auxilliary timing signals unique to the microword in which a specific Branch Micro Test (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 these BUTs do. 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.

K5-4 Print: FLAGS

Flag flip-flops for error conditions and machine sequencing are contained on this print. The logic discussion will treat 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 effected by the interaction of the flag flip-flops and basic to understanding their operation.

BUS ERROR Traps - Odd Address Fatal Stack Overflow (Red), Memory Management Violations to 250 (if KT11-D)

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

CONSOLE BUS REQUEST - Console operation after HALT switch

UNIBUS BUS REQUEST - Peripheral requests compared with Processor Priority, usually an Interrupt.

WAIT LOOP - Loop on a WAIT instruction in IR until an Interrupt allows exit. A CONSOLE BUS REQUEST returns to this loop after being honored.

BERR Flag - the Bus ERRor flip-flop provides a flag for trap service upon the occurrence of a NO DAta or ODd address ERRor in a processor Unibus transfer. The flip-flop is clocked to the one state by the activation of the data inputs from NODAT flip-flop (on TIMING) or the ODd Address ERRor signal with the clocking signal K4-3 JAM UPP H (which also jams the microflow to a trap routine). The BERR flag output generates appropriate STPM constants for trap vectors and accomodates 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 INITializing signal; the EXTernal Pulse CLeaR TRAP signal from the KE11-E option; BUT 03 in TRP16 microword at microaddress 140 in the trap sequence; and the establishment of a new stack at location 04 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 console operation. This allows the bus error of NO DAta 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 JAM UPP 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 a one state by the data input of a 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 JAM UPP signal directly clears the TRAP flag. TRAP flag service is aborted if a JAM UPP 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 flip-flop is clocked to the one state by the data and clock input of 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 Single Clock before completion.) After 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 TRAP flag is 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 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 INITializing signal and the operation of a new stack at location 04 upon 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 flip-flop is utilized directly on TIMING (print K4-2) to enable machine restart upon 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 one state if either error is present by the K4-4 CLK OVFLW H signal. Once 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 provide for proper trap sequencing.

A red zone stack error results in a JAM UPP 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 microflow 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 microflow 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 JAM UPP 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 no trap service condition. The flip-flop is clocked to the one state for DoUBle Bus ERRor, red zone stack overflow (K4-4 OVFLW ERR L) or PWRDN flag with the clocking signal K4-3 JAM UPP 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 processor INITialize signal and by BUT04 in TRP21 microword in the trap service. No inhibits exist on the BUT04 clearing of the STALL flag as the error condition requiring a suspension of overflow checking is serviced in this first trap service.

WAIT Flag - The WAIT flip-flop is clocked to the one 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.

The flag is cleared by the common clearing signal already noted under the BERR flag for the BERR, TRAP and INTR flags. Normal clearing of the flag occurs in the Bus Request service flow by BUT07 in SER10 microword at address 022.

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 BRPTR flag 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 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 one state by the data input of K3-4 OVLAP INSTR H signal 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. The flag also enables the IDLE flip-flop (print K4-2) upon 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 one or zero depending upon the K3-4 OVLAP INSTR H signal.

Print K5-5: CONSTANTS

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

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 throughout 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.

Print K5-6: CONSOLE

The logic associated with this print provides the necessary flags and Basic MicroBranch Constants (BUBC's) for console operation. The logic discussion is ordered toward console operation and not the 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-flop necessary to service the control switches with associated combinational logic to set and clear the flags. Some addition logic is necessary to generate the Basic MicroBranch Constants (BUBC's) utilized in the console flow service for microbranches to the individual switch service flows.

Activation of any console control switch (except ENABLE/HALT) results in the SWITCH flag flip-flop being clocked to the one 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 transition that clock the SWITCH flag also provided the signal levels necessary for the Basic Micro Branch Test (BUBC (2:0) (BUT30)) to access the individual switch flow responses. Reference to the Flow Diagram (D-FD-KD11-A-FD) for console operation and BUT30 show the exclusive nature of the switch BUBC code; only one switch can be serviced. The SWITCH flag is cleared by the processor INITializing 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.

Two switches, START and BEGIN require console flags. Each produces a non-filtered (contact bounce exists) INITializing signal upon console switch activation. Each clocks its flag flip-flop (and the SWITCH flag) to the one state as the switch is released. Both flag flip-flops then provide input 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 INITializing signal, by BUT37 in the Fetch flow, or by BUT10 in the Start flow.

The CONSL flag flip-flop is clocked to the one state upon 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 JAM UPP microaddress (print K4-3). Clocking for NPR's and BR's are also disabled (print K4-4). The flag is also used in the KT11-D option. The CONSL flag is clocked to the zero 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 essentially used for the same purpose. They provide automatic address incrementation for console operations which are consecutive EXAM's or DEP's. The flags are clocked to the one state during the latter part of their respective flow sequences: EXAM flag is clocked by BUT04 and DADO (1) H; DEP flag is clocked by BUT03 and DADO (1) H. The outputs of the 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 upon entry in the console flow (BUT24) and in Service (BUT26) and in the START flow (BUT05).

Print K5-7: CABLES

Two connectors are shown on this print. The KY11-D connector (J2) has associated logic to drive the ADDRESS display and accommodates the console control signals utilized on print K5-6. The other connector (J1) has limited capabilities which allow remote stop and start of the processor. This last connector is not used in the basic KD11-A processor.

Print K5-8: BUS DELAYS

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 initializing signal and a RESET RESTART signal. Start up delays for processor operation are provided by the PWRUP INIT and POWER RESTART.

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 insures that no power fail indications (the activation of BUS AC LO L) is missed and 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 delay interact (AC LO delay) but these are mostly ordered 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.

K5-8 PWR RESTART H - signal initiates a JAM UPP to begin microprogram sequences (print K4-3) approximately 70 milliseconds 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 exist. Variations in BUS AC LO L for power down are ignored.

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

K5-8 RESET RESTART - signal indicates the status of the 70 millisecond 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 INITializing 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 processor INITializing of 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 milliseconds and occurs upon the deactivation of BUS DC LO L. The signal initiates a JAM UPP in the micro control (print K4-3) to location 377 which contains all zeros.

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

K5-8 B 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 NPR's.

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.

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-06) 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.

6.2 KY11-D CONSOLE BOARD

The KY11-D Console board shown on print number D-CS-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 (LED's) 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; console notation for the displays is shown in parenthesis 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 parenthesis (SR09, for instance) 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 parenthesis, the connector signals are at the right.

An additional switch for Off, Power, Console Lock is also shown. Its connectors (J3,J4) consist of two quick disconnect tabs to allow direct interconnection to the Power Control Unit.

6.3 CABLES

The BC08R-06 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 as this connector provides only a switch closure, either interconnection of two wires is acceptable.

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. 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 options provides indicators and switches for manually operating the system and monitoring 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 option and KE11-F Floating Instruction Set option are both covered in the KE11 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, this 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 bus operations (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, then a red zone violation occurs and the operation causing the error is aborted. The stack is repositioned and a bus error trap occurs; that is, 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. Other fatal stack errors are odd stack or non-existent stack. 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.

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 logic 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 more 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.

Table 7-1

Comparison of Address and SLR

EXAMPLE 1 - VALID ADDRESS (GREATER THAN)

Bit Position (high byte)	15	14	13	12	11	10	09	08	Octal Value
Bus Address	0	1	1	0	1	1	0	0	0660
SLR Contents	0	1	1	0	1	0	1	0	0650

EXAMPLE 2 - 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

EXAMPLE 3 - 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

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 7-2

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	
⋮	More than SLR											⋮					VALID
400								1	0	0	0	0	0	0	0	0	
377								0	1	1	1	1	1	1	1	1	
⋮	Equal to SLR; bits 7,6,5 are all set											⋮					YELLOW
340								0	1	1	1	0	0	0	0	0	
337								0	1	1	0	1	1	1	1	1	
⋮	Equal to SLR; bits 7,6,5 are <u>not</u> all set											⋮					RED
000								0	0	0	0	0	0	0	0	0	

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

2. In <u>all</u> cases, highest yellow zone address must end in either 377 or 777.

3. In <u>all</u> cases, highest red zone address must end in either 337 or 737.

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 to the SLR Bus Address. The processor decodes this address and routes the data through the DMUX to the Stack Limit Register logic, providing a proper SSYN signal on the Unibus. These DMUX 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 (DOUT 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 reference to the SLR address with a DATO or DATOB 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. When the next stack pointer related to bus operation occurs, 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 <u>not</u> infringing on the stack because the bus address is higher than the stack limit value, no action occurs.

If A=B, indicating 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 <u>all</u> three of these bits are set, then K1-7 BA(07:05)=1 L is true and gates are enabled and KJ-2 EOVFLW L indicates a yellow zone violation. This signal also sets the V bit Condition Code in the Processor Status word. Note that one line on the gate that produces KJ2 EOVFLW L is tied to +5V. 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 infginging 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.

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 is lit, 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 .

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.

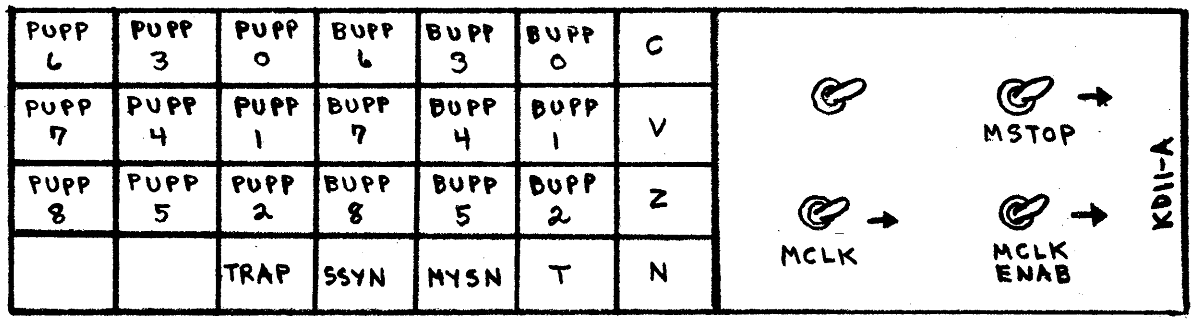

Figure 7-1 KD11-A Maintenance Console Overlay
(A-SS-5509081-0-12)

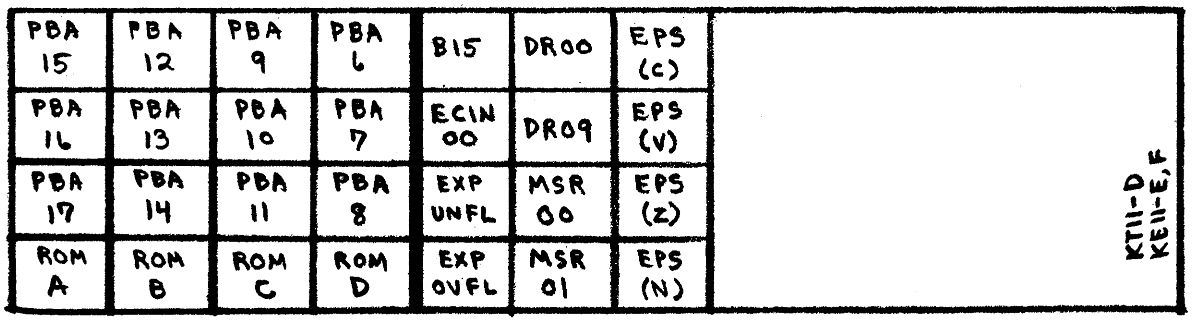

Figure 7-2 KT11-D, KE11-E,F Maintenance Console Overlay
(A-SS-5509081-0-13)

Table 7-3

KM11-A Controls and Indicators for KD11-A Overlay

Control or Indicator	Indication (when lit)	Print Showing Signal Origin
PUPP(8:0)	Indicates the Previous Microprogram Pointer (PUPP). These nine indicators represent a three digit octal word from 000 to 377. These indicators are the ROM address of the present U WORD.	K2-2, K2-3
BUPP(8:0)	Indicates the Buffered output of the 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 zero).	K5-2
N	Negative bit of the processor status word condition code is present (result of operation was negative).	K5-2

Table 7-3
(continued)

Control or Indicator	Indication (when lit)
MCLK ENAB	When set to on (in direction of arrow) this switch prevents the automatic reclocking of the CLK flip-flop on TIMING(K4-2) print. 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.
MCLK	This spring-loaded switch, (when moved toward the arrow) clocks the MCLK flip-flop on TIMING (K4-2) 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.
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 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 ONE 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 length. 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.

Table 7-4

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

Indicator	Indication (when lit)	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 than 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 lit.	KT-2

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

Indicator	Indication (when lit)	Print Showing Signal Origin
* 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 to the final bus cycle of the KD11 instructions for relocation in destination mode only.	KT-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.

Indicator	Indication (when lit)	Print Showing Signal Origin
B15	When lit, indicates that the first division step is an add function; if not lit, it indicates a subtract function.	K1-5
ECIN 00	Indicates an external carry into the arithmetic logic unit (ALU)	KE-5
EXP UNFL	Indicates that there is an underflow condition in the exponent as a result of the operation.	KF-4
EXP OVFL	Indicates that an exponent overflow condition exists.	KF-4
DR00	This indicator is used in conjunction with the B15 indicator and the EPS (C) indicator. When used with the B15 indicator, it provides one of the four indications listed below. For use with the EPS (C) indicator, refer to EPS (C).	KE-2

DR00	B15	
0	0	addition step in divide loop
0	1	subtraction step in divide loop
1	0	same as 01
1	1	same as 00

DR09	Used as a test for normalization (see flows)	KE-2
MSR00	When lit, indicates a subtraction operation in the addition routine of the floating divide loop. When off, indicates an addition operation.	KF-2
MSR01	When lit, indicates an addition in the floating multiply routine. When off, indicates a shift.	KF-2

Indicator	Indication (when lit)	Print Showing Signal Origin
EPS (C)	Carry bit of the processor status word associated with the EIS option. In addition, is used with DR00 bit to indicate the following:	KE-6

EPS(C)	DR00	
0	0	shift multipler
0	1	add function
1	0	subtract function
1	1	same as 00

Indicator	Indication (when lit)	Print Showing Signal Origin
EPS (V)	Overflow bit of the EIS processor status word.	KE-6
EPS (Z)	Zero bit of the EIS processor status word.	KE-6
EPS (N)	Negative bit of the EIS processor status word.	KE-6

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 outputs.

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 +5V 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 engineering drawing D-BS-KM11-0-MB. The three sheets of this drawing are labled KM-1, KM-2, and KM-3. The latter designations are used for the remainder of this discussion.

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 drawing 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 +5V power is applied at pin A2 of the W130 connector and is used to drive the W131 control switch logic. Nominal +8V power is applied at pin B1 of the W130 connector and provides power to the indicator lights. Each indicator driver circuit controls the voltage to its respective indicator light. The driver circuits are driven by the logic power of the signals being monitored.

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

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	uses 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 labled: 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 Engineerning Drawings.

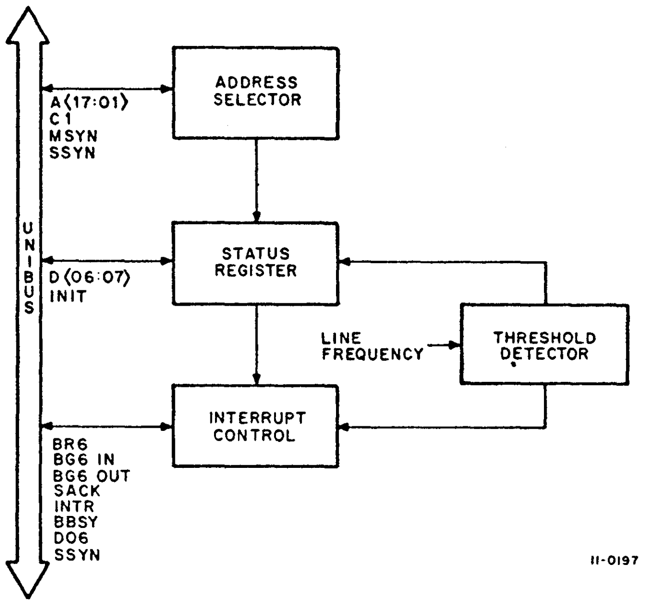

Figure 7-3   KW11-L Block Diagram

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 set or cleared by a processor DATO to the clock; it is also cleared by processor INIT. Bit 7 is set by a line clock pulse from the threshold detector or by a processor INIT; it is cleared by any processor DATO to the clock.

Bit 7 can be used by the processor to determine which device caused 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 signalling 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.

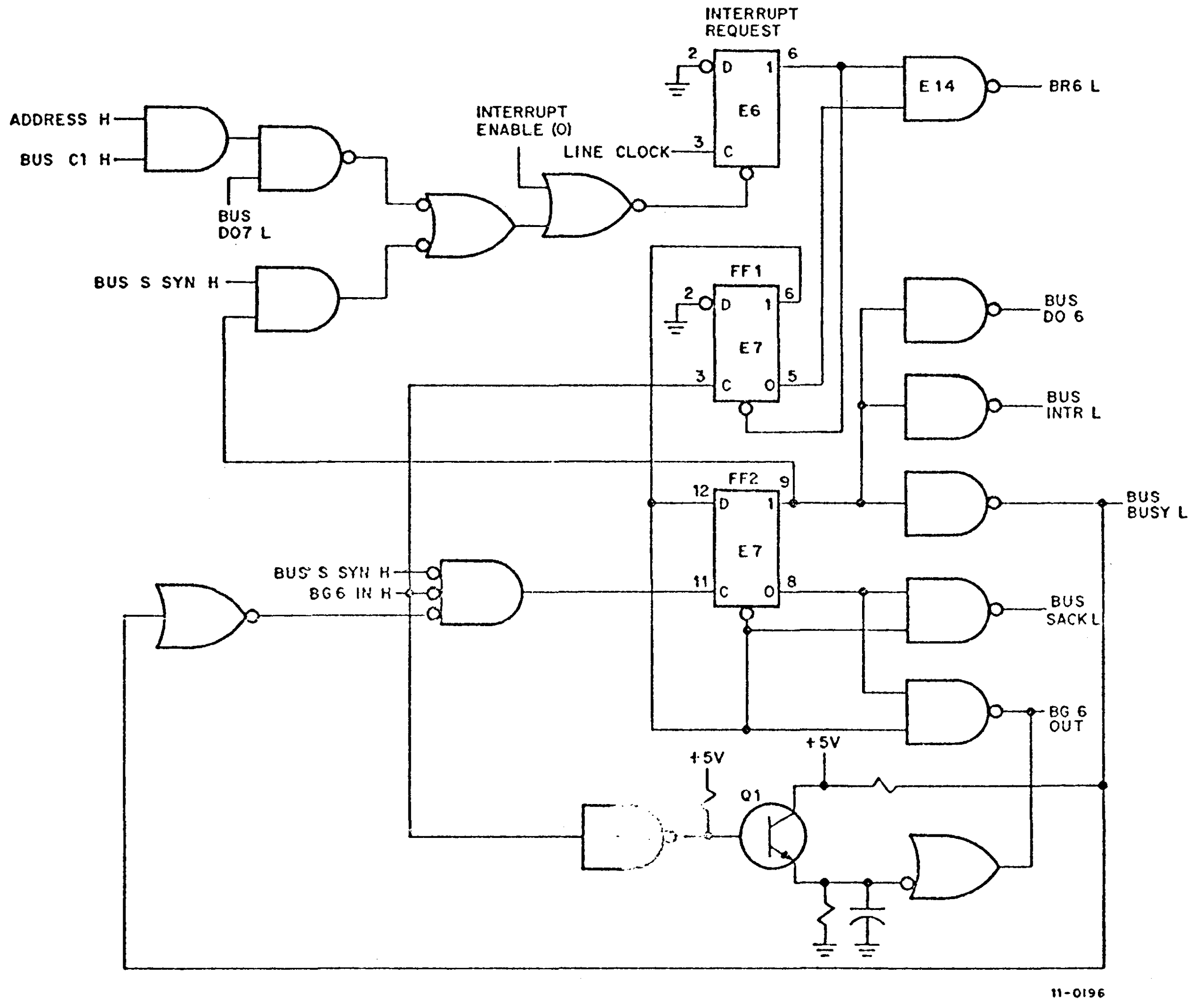

Figure 7-4 Interrupt Request Section - Simplified Diagram

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)

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.

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 depressing 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 on Figure 7-5 . The ADDRESS H signal is also used, after a dealy, to assert SSYN on the bus.

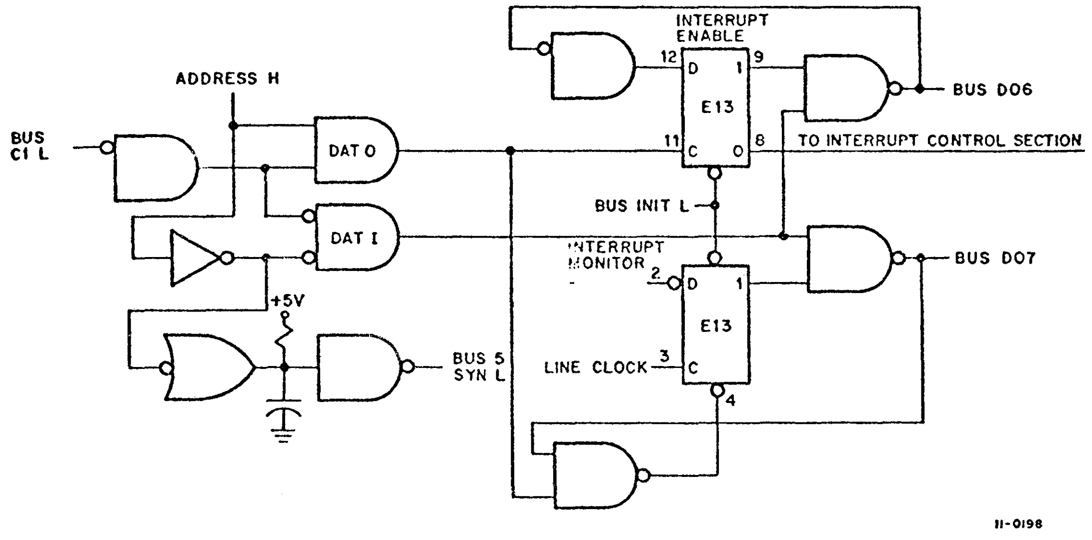

Figure 7-5 Status Register - Simplified Logic Diagram

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 read 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 onto BUS D06 and the contents of INTERRUPT MONITOR onto BUS D07.

digital

DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS 01754