PREFACE

This manual provides the user with all the information required to write, assemble, debug and execute microprograms on the LSI-11. This facility is provided by the Writeable Control Store (WCS) hardware (KUV11 option) in conjunction with microprogramming support software.

Chapter 1 provides an introduction to microprogramming the LSI-11. It discusses machine-micromachine relationships and supplies an introductory glossary of important definitions.

Chapter 2 is divided into two parts: LSI-11 machine architecture and operation. The first part is an LSI-11 family hardware overview and is included for completion. The new LSI-11 user can gain a basic system understanding from this, but should refer to the Microcomputer Handbook for a more complete description. The second part is recommended reading for all LSI-11 microprogrammers because it introduces control flow diagrams which explain the transfer of control between the LSI-11 machine and micromachine.

Chapter 3 follows the same organization as Chater 2, but concentrates on the LSI-11 micromachine. The first part covers the LSI-11 micro processor as implemented in the control and data chips and details their internal organization. The second part describes the micromachine operation and its relationship to higher level machine operation.

Chapter 4 presents the LSI-11 microinstruction set, organized on a functional basis. The chapter also contains a detailed explanation of each microinstruction along with an example containing assembly mneumonics and assembled octal equivalents.

Chapter 5 concentrates on the DATA ACCESS group of microinstructions and provides details sufficient to accurately determine the execution times for microprogrammed I/O transactions.

Chapter 6 presents the LSI-11 Writeable Control Store hardware and its relationship to the LSI-11 microprocessor. The discussion includes the data buffer and control/status registers.

Chapter 7 details the microprogramming support software, principally the microassembler. Also discussed are the WCS loader and dump routine. The chapter concludes with a discussion of how to form assembly mnemonics for user defined machine instructions.

Chapter 8 discusses techniques which may be used to write microprograms and further explains the examples presented in Chapter 4. It also contains longer examples as well as information on how to use selected portions of the LSI-11 console ODT microcode.

Chapter 9 contains information on the WCS module installation and checkout.

Chapter 1 INTRODUCTION TO LSI-11, PDP-11/03 USER MICROPROGRAMMING

CHAPTER 1

INTRODUCTION TO LSI-11, PDP-11/03 USER MICROPROGRAMMING

1.1 GENERAL

This chapter provides an introduction to LSI-11, PDP-11/03 microprogramming. Section 1.8 lists other documents referenced by this manual. With the microprogramming facility, the user has access to nearly all the resources and techniques used in implementing the LSI-11 architecture in four LSI (Large Scale Integration) circuit chips. In accepting this degree of programming freedom, the user also assumes new responsibilities. The benefits and demands of user-microprogramming are discussed in this chapter. All required supplementary detail is contained in this manual.

1.1.1 LSI-11, PDP-11/03 Microprogramming Features

1. LSI-11 microprogramming is implemented by a Writeable Control Store module in conjunction with microprogramming support software.

2. The addition of the WCS module places fully one half (1024) of the total (2048) control store locations addressable by the microprocessor at the disposal of the microprogrammer.

3. The microprogramming support software includes a microassembler, control store loader and dumper, and a Micro Octal Debugging Tool which allows simultaneous debugging at the machine and micromachine levels.

4. The microprogramming support software also includes the micro code which implements the EIS/FIS machine instructions. The user may selectively include portions of this microcode in the final control store load module, thus combining the existing advantages of the familiar extended and floating point machine instructions with user designed instructions.

5. LSI-11 microprogramming combines the simplicity of vertical microinstructions with the individual bit-control of horizontal microinstructions. The existing bit-control field supported by the CPU micromachine is expanded by two additional bits with the addition of the WCS module. This new bit-control facility is also timed with respect to the micromachine cycle, thus enabling higher control rates and more accurate control timing than can be achieved with normal LSI-11 system bus I/O control.

6. The WCS module is equipped with a 16-word recirculating microlocation trace buffer which aids in microprogram debugging. It contains the last 16 microlocations placed on the microinstruction bus. This hardware feature is utilized by MODT to display the last 16 microlocations accessed prior to a user-specified dump point.

1.1.2 Basic LSI-11 Machine/Micromachine Structure

Figure 1.1 contains a simplified illustration of the LSI-11 machine. The LSI-11 machine encompasses the processor, main store or memory, and the input/output devices. These three main components are interconnected by the LSI-11 system bus. The LSI-11 processor is realized by a micromachine which contains a (micro)processor element, a storage element (control store), and external I/O capability. The microprocessor is implemented with two chips (called the Data chip and the Control chip). The Data chip contains the Arithmetic Logic Unit (ALU), the register file, and provides connection to the 16 time multiplexed data/address lines (DAL). The Control chip arbitrates all system bus transactions and determines the microinstruction execution sequence. The structure of the microprocessor chip set is presented in detail in Chapter 3.

LSI-11 MACHINE-MICROMACHINE STRUCTURE

FIGURE 1.1

The microinstructions are stored in a control store consisting of one or more MICROMs which provide non-alterable, non-volatile memory. The term MICROM is an acronym for MICrocode Read Only memory. In the basic LSI-11 processor, the MICROM control store contains microcode which performs 2 functions: (1) PDP-11 emulation and (2) console ODT. When included on the processor, The KEV11 EIS/FIS option expands the control store to support the extended and floating point machine instructions.

The Writeable Control Store is connected to the micromachine via a cable/plug assembly (which replaces the KEV11 option in the third MICROM socket). This gives the WCS module access to the microinstruction bus which is the internal bus interconnecting all components of the micromachine. The WCS then may be accessed by the microprocessor in the same fashion as a MICROM. The total control store locations addressable by the microprocessor is 2048. The PDP-11 emulation and console ODT microcode requires only 1024 locations, or two MICROMs. The WCS module supplies the remaining 1024 control store locations and these are totally under control of the LSI-11 microprogrammer. The WCS memory is loaded and dumped via its LSI-11 system bus interface, which consists of normal control/status and data buffer registers.

1.2 THE BENEFITS OF USER MICROPROGRAMMING

The facility of user microprogramming affords a heightened degree of control over machine operations. The user can create new machine instructions which may be used in the same manner as members of the LSI-11, PDP-11/03 machine instruction set. The following paragraphs identify specific areas in which user microprogramming is beneficial.

1.2.1 Arithmetic Calculations

Many arithmetic calculations are characterized by a concisely-defined algorithm which is often repetitive in nature. The execution of the routine for such an algorithm normally requires many machine instruction fetches and operand address calculations. If the algorithm is suitable for microprogramming it can be implemented in microcode which is then executed in response to a single, user-defined machine instruction. This approach eliminates the multiple machine instruction fetch operations because control remains entirely within the micromachine until the routine is completly executed. A good example of the improvement available in this area is the KEV11 EIS/FIS option. The EIS/FIS option contains microcoded routines for the extended and floating point machine instructions. Upon execution of a single machine instruction, FDIV for example, control is transfered to the appropriate starting point in the FIS microcode. When the routine has terminated, control is returned to the LSI-11 emulation microcode contained in the 2 standard microms. The speed advantage realized by the EIS/FIS microcode ranges from a 3 to 10 times improvement over comparable PDP-11 macroinstruction routines, depending upon the instruction executed and the operand values.

1.2.2 Critically-Timed Input/Output and Control Operations

The rate at which real-time input/output (I/O) operations can be performed depends on two factors: (1) the speed of machine instruction execution and (2) the time delays associated with the LSI-11 system bus. The microprogramming facility allows the user to write specialized I/O routines for execution in microcode. In this context, the user employs the "data access" group of microinstructions which are discussed generally in Chapter 4 with detailed explanations in Chapter 5. sufficient information is provided to allow identification of where the bus delays occur. Although it is not possible to eliminate the time delays caused by the system bus, identification coupled with the use of microcode can make the delays manageable.

User-written microcode can make use of a special field of 4 TTL control bits within the microinstruction. Of the 16 possible states encoded in this field, 8 are presently used to control the LSI-11 system bus logic which interfaces between the system bus and the microprocessor chip set. The other 8 states are available for user-defined functions. The 4 bits from this microinstruction field appear on the LSI-11 module fingers as does the third phase of the microcycle clock (PH3 H). These TTL control bits enable the microprogrammer to produce high rate control signals which are timed with respect to the micromachine cycle.

In addition to the 4 control bits (MI<21:00>), the WCS module stores 2 extra bits (MI<23:22>) in each control store location. These two bits are available as signals directly on the LSI-11 system backplane and may be used for any user-defined purpose. The high-order bit (MI<23>) is also used to control the microlocation trace buffer.

1.2.3 Data Manipulation and Relocation

The two microprograms discussed in Section 1.6 provide additional examples of potential applications. The first example presents the microprogram for a routine which pushes all processor registers (R0-R5) onto the stack in response to one machine instruction. Following which the related function of popping all 6 registers is presented.

The second example is a block move operation in which the arguments required are (1) the starting address of the block to be moved, (2) the starting address of the new block location, and (3) the block length. The execution time saved is proportional to the block length, because machine instruction fetches are eliminated for each word moved.

1.3 SYSTEM IMPLICATIONS OF USER MICROPROGRAMMING

With the microprogramming facility, all the resources used to emulate the LSI-11 architecture and operation are at the user's disposal. The only resource that cannot be accessed is the Control chip translation array which is specifically configured to implement the opcodes of the standard and extended machine instruction set. However, the microprocessor instruction set does provide a means for accomplishing translations in only a few additional microcycles.

1.3.1 Control Flow Integrity

The first responsibility of the microprogrammer is to maintain the integrity of the control flow which implements the machine operating cycle. Control is transferred to user microcode when the microprocessor control chip, via the translation array, determines that a the fetched instruction is a user-defined machine opcode. User microcode then maintains control until the microroutine has executed, whereupon control must be returned to the trap interrupt service routine, thus maintaining the normal FETCH-EXECUTE-INTERRUPT machine cycle.

An additional requirement arises when the user-microprogrammed routine performs I/O operations. In executing an I/O transfer the LSI-11 system bus transaction requires that the addressed I/O device return a reply signal to the processor, acknowledging its role in the transfer. If no reply is received by the CPU within 10 microseconds, the processor executes a bus error trap through LSI-11 memory location 10. Because a bus error can occur in a number of contexts, the microprogrammer must prepare for the proper response by setting an internal flag (see Section 8.6.4.1).

1.3.2 Interrupt Response Latency

Interrupts are recognized by the processor only during the final phase of the normal machine operating cycle (FETCH-EXECUTE-INTERRUPT). The delay in acknowledging a pending interrupt is directly related to the length of time of the EXECUTE phase. The microprogrammer is provided with means of testing for external interrupts and the Event Line interrupt during user-microprogram execution. If such an interrupt is pending, control can be transferred to a user microcode routine. This routine could save the interrupted machine state, decrement the PC and then return control to the last part of the machine cycle.

Once the normal LSI-11 interrupt service has been completed, the user-defined machine instruction is fetched again and the user microprogram can then execute to completion. The external interrupt test facility allows potentially lengthy microcoded routines to operate in a time critical environment. Interrupt testing is unnecessary when the microcode to be executed is of known short duration. See Section

8.6.7.

1.3.3 Register Content Security

The microprogrammer has access to the LSI-11 processor registers as well as to the internal registers. Several of the internal registers have predefined uses (e.g., PSW, bus error flags) and should only be modified in accordance with those uses. Manipulation of the standard processor registers (R0-R7) should only occur as part of the intended function of the user-defined machine instruction (See Section 8.6.5).

1.3.4 Processor Status Word Updating

The processor status word (PSW) in the LSI-11 is a composite of (1) the 4 PDP-11 Status Flags (N,Z,V,C) and (2) the Trace Bit and (3) the Interrupt Enable bit. Internally, the PDP-11 Status Flags are explicitly accessed by 2 microinstructions (LFR, CFR) and implicity altered by executing a microinstruction capable of affecting these flags (e.g., ADDF). The Trace Bit and Interrupt Enable Bit are altered by executing a Set Interrupt (SI) or Reset Interrupt (RI) for the microprocessor control flags I^4 and I^5 respectively. Since these flags cannot be read, a copy of these flags is kept in an internal register (RPSWL H) In the bit positions that correspond to the Trace bit (bit <4>) and the Interrupt Enable bit (bit <7>) of the LSI-11 PSW. See Section 8.8.

1.3.5 Dedicated Control Store Locations

The 1024 Writeable Control Store locations are normally configured at microaddresses 2000 through 3777 (octal). The WCS module therefore replaces the EIS/FIS MICROM which answers to addresses 2000-2777. The LSI-11 emulation microcode, in conjunction with the translation array, performs a partial decode of the extended and floating point machine instructions and transfers control to control store locations in the 2000-2777 range. The microprogrammer has the responsibility of handling such a transfer as a reserved instruction trap. (See Section 8.12.1).

1.3.6 Machine Instruction Support

A newly defined machine instruction must be explicitly documented as to function, execution characteristics, and operand requirements. It should also be assigned an assembly mnemonic which is supported by a macro definition to enable the instruction to be programmed and assembled along with the standard instruction set (See Section 8.2).

1.5.1 Creating the Source File

The RT-11 Operating System environment in which the LSI-11 microprogrammer works is familiar to the experienced PDP-11 assembly language programmer. A fundamental reference is the microinstruction set description, which is contained in Chapter 4 and Appendices A and B of this manual. Chapter 7 describes the WCS Software Tools available.

The source file for the microprogram is created with either text editor, EDIT or TECO. The recommended extension in the file specification is .MIC signifiying a microprogram source file.

1.5.2 The Microassembler

Microassembler facilities are provided by MACRO-11, augmented by a preprocessor routine which generates the necessary calls to a special library file containing ~~of~~ a set of macro definitions that enable the assembler to recognize and process the microinstruction mnemonics. The Microassembler is described in detail in Chapter 7. The capability of MACRO-11 to accomodate large symbol tables allows a microassembly to be performed in a minimum configuration (16K) development system. The microassembler output (.OBJ extension) is then loaded into the WCS module as described in the next section.

1.5.3 Writeable Control Store Loaders

The microprogramming support software provides a WCS loader program which can initially clear the WCS RAM memory and then load from one to six specified .OBJ files produced by the Assembler. The WCS loader is described in Chapter 7.

1.5.4 Micro Octal Debugging Tool (MODT)

The requirements of microprogram debugging are satisfied by the Microprogram Octal Debugging Tool program (MODT). This debugging facility expands the familiar PDP-11 ODT so that user-written microprograms can be examined, modified, and executed from the system console. Since all microprograms are executed in response to machine instructions, MODT allows the user to write, modify and execute short programs. Breakpoints may be set in the machine instruction flow to examine program progress at the machine level. At the micromachine level, the user may establish "dump points" which display the contents of all internal registers at the selected point. Once the dump results are analyzed, the dump points can be moved and subsequent executions allow the user to examine additional portions of microcode. This dump-examine process continues until the microprogram is completely debugged and verified. MODT is described in detail in Chapter 7.

1.5.5 Microprogram Trace Facility

The microaddress trace hardware on the WCS module is normally controlled and examined by the Micro Octal Debugging Tool (MODT) program. It consists of a 16-word buffer which stores a sequence of 16 microinstruction locations placed on the microinstruction bus (MIB) during the execution of user-written microcode. The last location to be stored is designated by the microprogrammer (using MODT). Once the 16 addresses have been stored, MODT may display the contents of the addressed locations in both octal and symbolic forms. See Chapter 7 for a complete description.

1.6 SAMPLE MICROPROGRAMS

The microprograms presented in this section are tested, working routines which may be microcoded and executed by the user. They serve to introduce a few of the microinstruction assembly mneumonics and some special characteristics of the data access group of microinstructions. These programs are supplied with generous comments in order to make them more understandable. The full microinstruction set is presented in Chapter 4, and additional details on the operation of the data access group in Chapter 5.

1.6.1 Pushing and Popping the Processor Registers

Figure 1 shows a microprogram example that implements 2 new LSI-11 instructions as described below:

Opcode	Description
076700	PUSHR - causes the LSI-11 General Purpose Registers (R0-R5) to be saved on the system stack (pointed to by R6).
076701	POPR - causes the LSI-11 General Purpose Registers (R0-R5) to be restored from the system stack.

These instructions facilitate context switching for subroutines or interrupt service routines where many registers may be needed, but the previous register contents must be saved.

The actual microprogram source file for the new instructions is listed below, and can be assembled by the microassembler, loaded into the WCS, and tested using MODT (see Chapter 7). Note that the assembly listing for this example is in Chapter 7.

1.6.2 Moving A Block of Data

PUSHR and Popr Example

Figure 1,2

This example is the implementation of a new instruction, BLKMV, that moves a string of bytes from one location in memory to another non-overlapping location in memory. The opcode is 076700, R0 contains the starting address, R1 contains the destination address and R2 contains the byte count to be moved. Note that this exmpale is microinterruptable.

BLKMV Example

Figure 1.3

INTRODUCTORY MICROPROGRAMMING GLOSSARY

This section contains a glossary of selected terms which provide an introduction to the vocabulary which supports microprogramming concepts. The glossary is not arranged alphabetically, but in a way which will emphasize conceptual distinctions.

Machine — The LSI-11 machine encompasses the LSI-11 processor, main or program memory and input/output devices.

Micromachine — The micromachine is equivalent to the LSI-11 processor. It encompases a two-chip microprocessor, two or three MICROMs (Control Store chips), and the LSI-11 system bus interface logic. The first two MICROMS contain the LSI-11, PDP-11/03 emulation microprogram as well as the console ODT microprogram. The optional KEV11 MICROM contains microprograms which execute the extended and floating point machine instructions.

Processor — The LSI-11 processor consists only of the Large Scale Integrated microprocessor chip set and the LSI-11 system bus interface and control logic. A system based on the LSI-11 processor is bus-oriented in that a single machine instruction may manipulate data within a bus-installed device. The management of data in and out of the processor is often transparent or irrelevant to machine-level programming functions.

Microprocessor — The microprocessor is implemented as a two-chip set of large scale integrated (LSI) circuits called the Data and Control chips. The Data chip contains the Arithmetic Logic Unit (ALU), the Register File and interconnection to the LSI-11 system bus address/address lines (DAL). The Control chip provides the system bus control lines and contains the translation array. Microprocessor programs are contained in Microcode ROM chips called MICROMs.

Data Chip — All Data processing occurs within the Data chip. It contains the ALU and the internal registers. It also provides bi-directional connection to the LSI-11 system bus data/address lines (DAL).

Control Chip — The Control chip functions as a controller/sequencer. It provides all control signals for the system bus and also determines which microinstructions are executed. An important feature of the Control-chip is the translation array. This is a large combinatorial logic network which accepts the machine instruction as input and outputs the addresses of the microprogrammed routine appropriate to that instruction.

MICROM Chip

The MICROM (MICrocode Read Only Memory) chip impelements the unalterable, non-volatile read only control store memory. One or more MICROMs are connected to the Data and Control chips via the microinstruction bus (MIB).

System Bus

The LSI-11 system bus is the one common, bidirectional path which passes address, data and control information between the LSI-11 processor and all other modules which make up a given machine configuration.

Microinstruction Bus

The Microinstruction Bus (MIB) is the common bidirectional and control path between all elements of the micromachine. It is through connection to this bus that the WCS module makes its control store accessible to the microprocessor.

Instruction Register

When an LSI-11 machine instruction is fetched from main memory, it is placed in the Instruction Register (RIR) in the Data chip as well as in the translation register in the Control chip. The instruction register provides a convenient method for decoding an instruction in addition to utilizing the translation register.

Microinstruction Register

A microinstruction which has been fetched from either MICROM or user control store is placed in the microinstruction register. This register is implemented on both the Data and Control chips. The microinstruction register on each chip interprets only the portion of the microinstruction relevant to the chip's function.

Program Counter

The Program Counter (PC) stores the main memory address of the next machine instruction to be feched during a machine cycle. It is automatically incremented by 2 as the last part of the FETCH Machine Instruction phase.

Microprogram Counter

The Microprogram Counter (MPC) stores the address of the next microinstruction to be fetched during a microcycle. The microprogram counter may be loaded from various sources, depending upon the type of microinstruction being executed.

Program Store

Program Store refers to the LSI-11 main memory which contains both program and data information. Program store is accessed via the LSI-11 system bus and its size is determined by the bus addressing capability (32K 16-bit words maximum). Program store may alternately be referred to as Main Store, Memory or sometimes as core.

Control Store

Control store refers to the storage area for microcode and consists, potentially, of 2048 locations. Apart from user microprogramming via the WCS module, control store is implemented in one or more MICrocode Read Only Memories, or MICROMs. Microcode is accessed by the microprocessor much faster than machine instructions are accessed by the LSI-11 processor.

Writeable Control Store

Writeable Control Store (WCS) refers to a control store which may be altered by the user. Alterable or writeable control store is implemented by fast semiconductor read/write random access memory (RAM) devices which are interfaced to the microinstruction bus. The LSI-11 Writeable Control Store uses volatile RAM devices which means appropriate power-up loading procedures must be utilized.

Machine Cycle

Machine cycle refers to the smallest complete cycle of operations performed by the LSI-11 machine. The three fundamental operations of the machine cycle are: (1) FETCH MACHINE INSTRUCTION, (2) EXECUTE MACHINE INSTRUCTION (3) TRAP AND INTERRUPT SERVICE. The number and type of micromachine operations which must be performed during a given machine cycle are a function of the fetched machine instruction and whether a machine interrupt or system fault is present. Consequently, the time required to complete a machine cycle is variable.

Microcycle (Micromachine Cycle)

Microcycle refers to the smallest complete cycle of operation performed by the LSI-11 micromachine. The operations which make up the microcycle are of a more fundamental nature than those of which a machine cycle is composed. A microcycle is further decomposed into four equal phases (PH1-PH4).

Machine Instruction

A machine instruction is the 16-bit word input from main memory to the processor during a machine instruction fetch. The machine instruction is the smallest indivisable machine operation which can be controlled by the programmer.

Machine instructions are commonly referred to as instructions.

Microinstruction (Micromachine instruction)

The micromachine instruction or microinstruction is a 22-bit word which is executed by the microprocessor. The operations controlled by microinstructions are of a more fundamental nature than those controlled by machine instructions. Consequently, several microinstructions are executed to implement each machine instruction executed.

Machine Code

Machine code refers to the binary object code as stored in main memory ready for direct execution by the processor. The term is often shortened to "code".

Microcode (Micromachine code)

Micromachine code refers to the assembled 22-bit microinstructions which are stored in the MICROMs (control store) or in Writeable Control Store. This term is often expressed by the contracted form microcode.

Assembly Language

Assembly language refers to the collection of machine opcode and operand mnemonics, syntactical rules, and macroinstructions which assist the programmer in the creation of machine language programs.

Microassembly Language

Microassembly Language refers to the collection of micromachine opcode and operand mneumonics, syntactical rules, and macroinstructions which assist the programmer in the creation of micromachine language programs.

Assembler

The assembler is a program development tool which accepts as input a source file containing machine assembly language with optional comments. The assembler output is an object file (.OBJ extension) which can be linked with other object files to form an executable load file or module (.SAV extension). The assembler supplied with RT-11 V03 is MACRO-11.

Microassembler

The microassembler is a microprogram development tool which accepts as input a source file containing micromachine assembly language with optional comments. The microassembler output is an object file which may be loaded into the Writeable Control Store. The loaded microcode is then executed by the micromachine in response to a user-defined machine instruction.

Interrupt	The action of diverting control from the normal (uninterrupted) machine operating cycle of repeating FETCH-EXECUTE events. The simplest complete operating cycle is expressed as FETCH-EXECUTE-INTERRUPT.

1.8 REFERENCES

The following documents provide sufficient background information for this manual:

1. RT-11, System User's Guide, DEC-11-ORGDA-A-D
2. PDP-11 TECO User's Guide, DEC-11-UTECA-A-D
3. The Microcomputer Handbook, EB-07948-53
4. KD11-H processor schematic diagram, D-CS-M7264-0-1 (revision Y or greater)
5. WCS schematic diagram, D-CS-M8018-0-1

Additional copies of all items above can be ordered from:

DIGITAL Equipment Corporation
444 Whitney Street
Northboro, MA 01532

Attn: Communications Services (NR2/M15)
Customer Services Sections

CHAPTER 2

THE LSI-11 MACHINE STRUCTURE

2.1 GENERAL

This discussion of the LSI-11 machine structure covers both the architecture and operation of the LSI-11. Architecture relates to the system resources and their configuration. Operation indicates how data and address information is moved between and manipulated within the system resources.

This chapter is a subset of the Mirocomputer Handbook, which should be consulted for additional detail, specifically in the areas of LSI-11 options and hardware. This chapter emphasizes selected topics which are essential to the understanding of the microprocessor contained within the LSI-11 processor module. Chapter 3 enhances the information sufficient to perform conventional machine language programming to include the additional information required to microprogram the LSI-11 micromachine.

2.2 MACHINE ARCHITECTURE

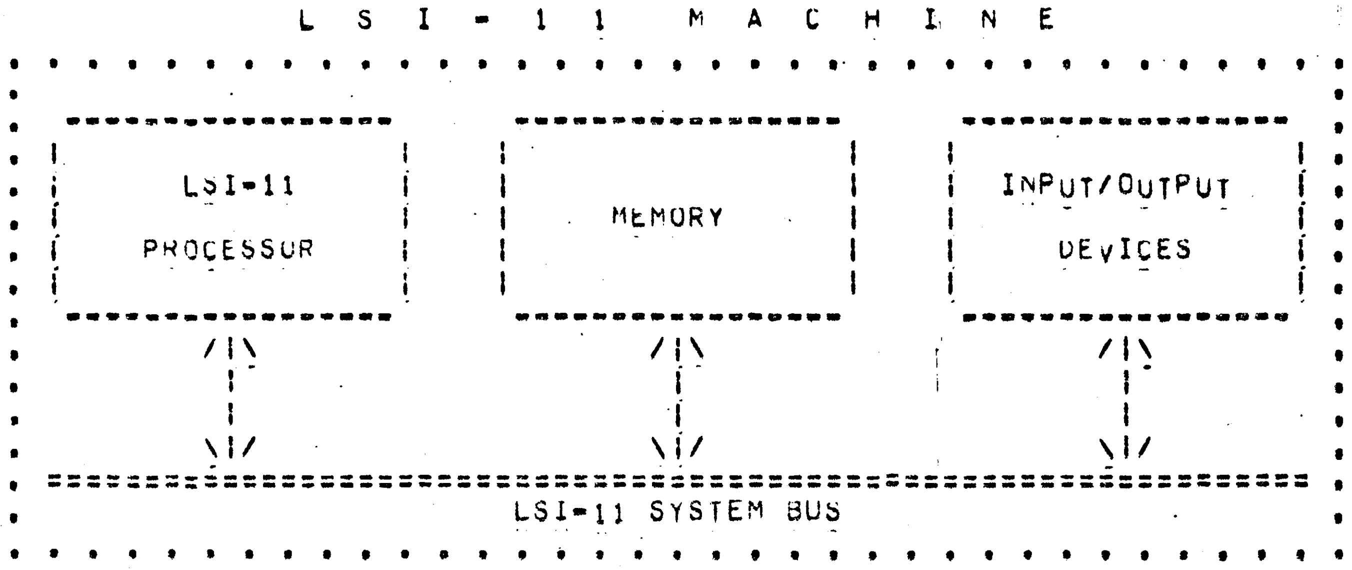
The LSI-11 machine consists of three general component areas connected by a common system bus. Any specific machine configuration may be accurately represented by using specialized components in these three areas:

1) The LSI-11 Processor

2) The Memory

3) The Input/Output Devices

These components and their common, bidirectional access to the system bus are illustrated in Figure 2.1.



LSI-11 Machine Structure

Figure 2.1

2.2.1 System Bus

The system bus is characterized by the number of memory or device locations addressable, the type of information transfers supported, and by the auxillary system control signals it contains.

2.2.1.1 System Bus Address Space - The virtual (and physical) addressing capability of the system bus is determined by the 16-bit width of the binary addressing word. The system bus supports both 8 bit byte and 16 bit word addressing. Figure 2.2 illustrates the address space with which the LSI-11 machine operates. The bottom 28K (28,672) word addresses constitute the memory space. The top 4K (4096) word addresses are normally dedicated to the Input/output devices. The LSI-11 processor does not occupy any address locations.

2.2.1.2 System Bus Data Transfer - All data transfer operations supported by the system bus are under the control of the LSI-11 processor. A portion of the processor is dedicated to the control of the system bus, and administrates the transfer of data and address information between the machine components. There are three types of data transfers possible within the LSI-11 machine:

1) Programmed Input/Output Transactions (Programmed I/O)

2) Interrupt-Driven Input/Output Transactions (Interrupt I/O)

3) Direct Memory Access Input/Output Transactions (DMA I/O)

Programmed I/O occurs in response to programmed machine instructions. An example of Programmed I/O is the execution of a MOV instruction where at least one of the source or destination addresses is in memory or in a device register. If both source and destination operands are in the processor registers, no bus I/O transfer is required.

The simplest of a Programmed I/O bus transaction is the DATI (Data-In) bus operation. This operation illustrates how the 16 Data/Address Lines (BDAL L <15:00>) are time-multiplexed between address and data information. The first event of the DATI cycle is placing the memory or device address on BDAL L <15:00>. After the address information settles (or becomes valid), the BSYNC L signal is asserted. This signal causes each memory and I/O DEVICE module connected to the bus to check whether the address corresponds to its own address(es). Recognition by the addressed device is represented by a signal internal to that device which is latched (or stored) by the assertion of BSYNC L for the duration of the bus operation. Storing this signal constitutes the second event. The third event of the DATI cycle removes the address information from BDAL L<15:00> and asserts BDIN L.

This signal informs the memory or selected device that a DATA-IN cycle is to be performed. The fourth event is in response to BDIN L; the selected device places its data on BDAL L<15:00> and returns BRPLY L to the processor. The asserting of BRPLY L is the first indication to the processor that the addressed component exists and is putting data on the bus. Upon receipt of BRPLY L, the processor accepts the information on BDAL L<15:00> as valid and stores it internally. If BRPLY L is not received within 10 microseconds after BDIN L was asserted, a processor trap occurs to memory lcoation 10. By this means, the processor avoids waiting for a reply from a memory location or device which does not exist or is malfunctioning. The bus cycle's fifth event is the accepting and storing of the data and the subsequent negation of BDIN L. In the sixth event, the selected device responds to the negation of BDIN L by terminating BRPLY L. In the seventh and final event, the processor terminates the bus cycle by negating BSYNC L.

The DATI bus cycle is described in detail in the Microcomputer Handbook along with the DATO (DATA-OUT), DATOB, DATIO (Data In-Output) and DATIOB bus cycles. The "IO" (Input-Output) bus cycles enable the processor to execute a Read-Modify-Write operation. This allows data to be retrieved from an addressed location, manipulated within the processor, and re-deposited in the same location while asserting only one bus address. The "B" suffix on the DATOB and DATIOB bus cycles indicates that the output portion of the bus cycle is a byte rather than to a word transfer.

Interrupt I/O is initiated by an Input/Output device. The interrupting device requests service from the LSI-11 processor by asserting the BIRQ L signal on the system bus. When the processor acknowledges a device request by asserting the BIACK H signal , it causes the interrupting device that is electronically closest to the processor on the LSI-11 bus to return a vector address. The vector address tells the processor where the address of the device service program is located in memory. Once execution of the service program has been completed, the LSI-11 processor resumes execution of the interrupted program.

Priority among multiple external devices having Interrupt I/O capability is established via electrical position relative to the processor. The device closer to the processor posesses the higher priority. The interrupt Grant chain is broken by the device being serviced, thus making further interrupts from lower priority devices impossible. Further interrupts from higher priority (electrically closer) devices may still be acknowledged and serviced. Additional details regarding Interrupt I/O transactions are available in the Microcomputer Handbook.

Although the LSI-11 processor remains master of the system bus during both Programmed I/O and Interrupt I/O. DMA I/O requires that bus mastership be granted to an Input/Output device for the purpose of high speed data transfer to or from memory. During a DMA I/O transaction, the Programmed I/O and Interrupt I/O operations of the LSI-11 processor are suspended. The device functioning as bus master can now perform any of the possible bus cycles (DATI, DATO, DATOB, DATIO, DATIOB). When the DMA transfer is completed, bus mastership is relinquished to the processor. Note that during DMA I/O, the CPU module waits to perform Programmed I/O or Interrupt I/O only. Any other processor activity (such as operations between registers) can continue.

2.2.1.3 System Bus Control Signals - In addition to the bus signals required to support Programmed I/O, Interrupt I/O, and DMA I/O several auxilliary control signals are contained within the bus system.

LSI-11 Machine Address Space

Figure 2.2

(Use figure on page 483 of the LSI-11 handbook)

System Initialization Signal

The common system initialization signal is BINIT L. It is asserted by the processor whenever BDCOK H is passive and whenever the BINIT command is issued by the processor. Examples of the latter are during the Power-up routine and in execution of the RESET machine instruction. All peripheral devices should use the BINIT L signal to initializae and clear internal flip-flops and registers.

Power Supply Signals

Two signals which indicate the status of the system power supply are part of the bus system. These signals are generated by the power supply itself and the processor monitors them to take appropriate action during Power-Up and Power-Fail sequences. At the beginning of the Power-Up sequence, BDCOK H and BPOK H are both passive. BDCOK H is asserted first, whereupon the processor initializes itself and all system components and waits for BPOK H to be asserted. When BPOK H is asserted, the processor executes the selected Power-Up routine. During a Power-Down or Power-Fail sequence, BPOK H is negated first, causing a Power-Fail trap. The processor then executes the program located at the trap vector address (24). When BDCOK H goes passive, the processor asserts BINIT L. It should be noted that a proper power-Fail sequence will negate BDCOK H before restoring BPOK H. BDCOK H must go passive to re-initiate the Power-Up sequence.

Processor Control Signal

The only signal in the bus system directly controlled by the operator of the processor is BHALT L. This signal is asserted by a front panel (PDP-11/03) switch or alternately by the BREAK key on the console terminal. Pressing the BREAK key causes a framing error which asserts BHALT L via the console serial line interface. When the front panel Run/Halt switch is used to halt the processor, it must be reset to RUN before the processor can proceed. However, pressing the console BREAK key asserts BHALT L only as long as the key is depressed.

Processor Monitor Signal

The ability to monitor the Run/Halt state of the processor is provided by the SRUN signal. The SRUN signal is asserted once each time the processor performs a machine instruction fetch. This signal is the input to a circuit on the PDP-11/03 console that drives a RUN indicator light.

Memory Refresh Control

All dynamic MOS memory modules which do not have self-contained refresh capability must be refreshed via the system bus. A signal called BREF L is asserted during the addressing portion of the

BSYNC/BDIN transaction to differentiate between memory refresh and the standard DATI bus cycle.

The details of memory refresh generation are discussed further in Section 2.2.2.3.

2.2.2 Memory

The memory component of Figure 2.1 may be implemented with either semiconductor or magnetic devices. Each memory device has operating characteristics which determine how it is to be used in a particular system.

Most memory devices function as Read/write memory may be accessed by either the processor or a DMA I/O device.

2.2.2.1 Semiconductor Memory - Semiconductor memory may be classified as either dynamic or static memory. The most familiar example of dynamic semiconducor memory is the 4K MOS Read/Write memory located on the KD11-F LSI-11 processor module. Dynamic memory must be periodically refreshed in order for the memory to retain its contents.

Static semiconductor memory is also available as Read/Write memory and does not require refreshing. An example of this type is the 256 16-bit words of RAM found on the MRV11-BA UVPROM module. An additional static semiconductor memory type is Read Only Memory. READ ONLY memory types are found on the MRV11-AA PROM module and on the MRV11-BA UVPROM module. Both the PROM and UVPROM are non-volatile types: they retain their contents even after power has been removed. The MRV11-AA PROM module contains fusible link semiconductor memory devices whose contents are established by special programming equipment. Once programmed, the fusible link PROM contents can not be altered. The MRV11-BA UVPROM is also programmed by special equipment, but its contents can subsequently be erased by exposure to UV light, effectively clearing the contents in preparation for re-programming with new data.

2.2.2.2 Dynamic Memory Refreshing - Three techniques are available to satisfy the requirement for dynamic memory refreshing. These three techniques are:

1. Processor Controlled Refresh

2. Direct Memory Access Refresh

3. Distributed Refresh

Processor Controlled Refresh is a feature available on the M726-

LSI-11 processor module. This refresh mode is normally disabled, but can be enabled by removing the appropriate jumper on the module. When enabled, a 600 Hz oscillator causes an internal interrupt. This interrupt initiates the execution of a microprogrammed routine which refreshes all dynamic memory devices used in the system. The refresh routine performs 64 BSYNC/BDIN cycles with BREF L asserted, occupying the system bus for about 130 microseconds. When processor-controlled refresh is employed, all dynamic memory modules should have their Reply During Refresh options disabled, except for the memory module located farthest from the processor. This assures that the longest bus delays will be compensated for during execution of the refresh microprogram. Note that this form of refresh is not recommended for systems utilizing lengthy customer microprograms.

Direct Memory Access Refresh is performed by a DMA device and relieves the processor of refresh responsibility. Examples of DMA refresh are the REV11-A and REV11-C modules. The DMA refresh technique differs from the processor-controlled method in that the memory modules are refreshed one row at a time instead of all rows at once. Therefore the DMA device controls the system bus for purposes of refresh for only 1.5 microseconds at a time, thus eliminating the 130 microsecond dead time inherent in the former method. Any user-designed DMA device may also perform refreshing as long as proper sequencing and timing assure that each row of memory is refreshed at least once each 2 milliseconds or less.

Distributed Refresh is the technique used by the MSV11-CD memory module. This module is equipped with timing and sequencing circuitry which performs refreshing for its own memory devices. It refreshes one row at a time every 25 microseconds and isolates itself from the system during each row refresh. If the LSI-11 processor, or DMA device, requires a memory access when the module is refreshing a row, the memory module merely delays its response by returning the BRPLY L signal after the row refresh is completed. The relatively short, occasional delays that occur with this technique are compatible with the essentially asynchronous system bus data transactions.

The memory component of a specific LSI-11 machine may be composed of more than one memory type to satisfy user requirements.

2.2.2.3 Magnetic Memory - Magnetic Memory is non-volatile, so it does not require refreshing. Therefore, memory contents stored in magnetic core are not lost during a power failure. Magnetic memory can maintain the information of a partially-executed program or routine when power is removed so that the routine may continue to completion when power is restored. A Power-Fail routine (which is initiated by the processor Power-Fail trap) must save all volatile machine state (e.g., the General Purpose Registers) in the magnetic portion of memory before power goes down (BDCOK H goes passive).

2.2.3 Input/Output Devices

The Input/Output devices connected to the LSI-11 system bus provide a means for interfacing control and data information between the LSI-11 machine and the outside world. An example of an I/O device which bi-directionally passes both data and control is the DLV11 serial line unit which connects to the console terminal.

2.2.3.1 Device Address Format - All Input/Output device locations on the LSI-11 system bus are accessed in the same manner as memory. Normally, each I/O device has four sequentially-numbered word locations associated with it. These four locations provide for both control and data transfer between the processor and I/O device according to the following convention:

XXXXX0	Receive Control Status Register	(RCSR)
XXXXX2	Receive Buffer	(RBUF)
XXXXX4	Transmit Control Status Register	(XCSR)
XXXXX6	Transmit Buffer	(XBUF)

The Receive Buffer (RBUF) holds data which has been received from the I/O device and which can be transferred to the processor or to memory. The Receive Control Status Register (RCSR) contains control flags related to the device's receive function. The Transmit Buffer (XBUF) holds data which has been transferred to the I/O device for presentation to the outside world. The Transmit Control Status Register (XCSR) contains control flags related to the device's transmit function.

2.2.3.2 Enabling Device Interrupts - Input/Output devices which support interrupt-driven I/O transactions are equipped with an interrupt-enabling mechanism which must be explicitly set by the processor before the device can initiate an interrupt. The interrupt-enabling mechanism is reset or disabled at Power-Up time by the system initialization signal BINIT L.

2.2.3.3 DMA Transfer Restrictions - A Direct Memory Access (DMA) data transfer is accomplished by the DMA device becoming master of the system bus (which suspends LSI-11 processor I/O operations). If the processor is responsible for dynamic refresh, only single byte or single word transfers are allowed to give the processor opportunity to execute a memory refresh routine. A long burst of DMA transfer could cause the refresh period to delay beyond the 2 millisecond maximum allowable time. If Processor-controlled Refresh is not utilized, restrictions on DMA transfer are eliminated.

2.2.4 The LSI-11 Processor

The LSI-11 Processor module which appears in Figure 2.1 is presented with greater internal detail in Figure 2.5. The three functional areas are:

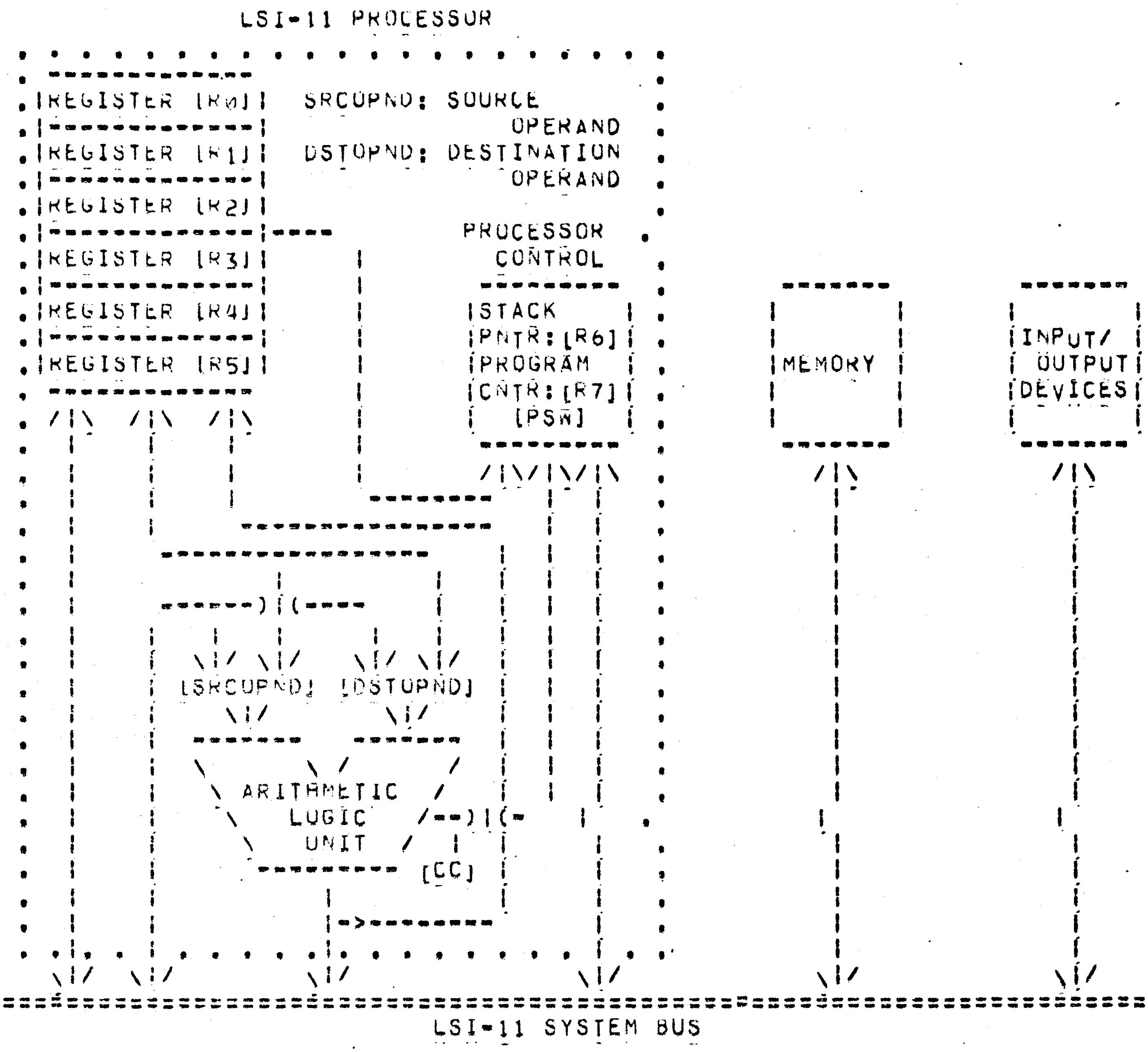
1) The Arithmetic Logic Unit

2) The General Purpose Registers

3) The Processor Control

2.2.4.1 Arithmetic Logic Unit - The Arithmetic Logic Unit (ALU) performs the operations required by the machine instruction set. Arithmeic operations employ two's complement number representation in fixed point format and with the addition of the KEV11 EIS/FIS MICROM (Extended/Floating Instruction Set), floating point arithmetic operations are also performed. Arithmetic and logical operations are executed on both byte and word data. The results of ALU operations are monitored by four Condition Code Flags (N,Z,V,C) which are part of the Processor Status Word (PSW) register. The source and destination operands which constitute the inputs to the ALU may be located in the General Purpose Registers, in Memory, or in Input/Output device registers.

2.2.4.2 General Purpose Registers - The General Purpose Registers are located within the Processor and thus their contents are accessed without the use of a system bus operation. The registers may contain data or address information. Registers R6 and R7 are dedicated to Stack Pointer (SP) and Program Counter (PC) use, respectively, and are therefore associated with Processor Control. Both byte and word addressing is supported for registers R0 through R5. Because of their dedicated application, registers R6 and R7 are restricted to word addressing only.



LSI-11 Processor Detail

Figure 2.3

2.2.4.3 Processor Control -
There are four main areas into which all functions petormed by the Processc Control may be divided. These four areas are:

1) Processor Control (Overall Control)

2) Machine Instruction Execution

3) Address Generation

4) System Bus Control

These four areas and their associated processor resources are illustrated in Figure 2.4.

```
 ------------------------------------------------------------
|                                                            |
|                    PROCESSOR CONTROL                       |
|                    =================                       |
|                                                            |
|          POWER SUPPLY STATUS     POWER-UP OPTIONS          |
|          -------------------     POWER-FAIL TRAP           |
|                                                            |
|          OPERATOR CONTROL        RUN/HALT SWITCH           |
|          ----------------        CONSOLE ODT               |
|                                                            |
|------------------------------------------------------------|
|                            |                               |
|   ADDRESS GENERATION       |      INSTRUCTION EXECUTION    |
|   ==================       |      =====================    |
|                            |                               |
|   INSTRUCTION ADDRESS      |      INSTRUCTION TYPES        |
|   -------------------      |      -----------------        |
|                            |      DATA MANIPULATION        |
|                            |      PSW  CONTROL             |
|                            |      PROGRAM CONTROL          |
|                      -------------                         |
|   PROGRAM            | FETCH       |                        |
|   COUNTER [R7]  ---  | MACHINE     | --->                   |
|                      | INSTRUCTION |                        |
|                      -------------                         |
|   OPERAND ADDRESS          |                               |
|   ---------------          |     ARITHMETIC LOGIC UNIT     |
|                            |       CONDITION CODES         |
|                            |        PSW BITS 3-0           |
|                            |                               |
|   GENERAL PURPOSE          |   SOURCE OPERAND              |
|   REGISTERS [R0-R5]        |                               |
|                            |   DESTINATION OPERAND         |
|                            |                               |
|                      -------------                         |
|   STACK              |   TRAP  &   |                        |
|   POINTER [R6]       |  INTERRUPT  |                        |
|                      |   SERVICE   |                        |
|                      |  PSW BIT 7  |                        |
|                      |  PSW BIT 4  |                        |
|                      -------------                         |
|                            |                               |
|                            |                               |
|------------------------------------------------------------|
|                    SYSTEM BUS CONTROL                      |
|                    ==================                      |
 ------------------------------------------------------------
```

Processor Control Functions
Figure 2.4

Overall Processor Control is controlled by (1) the status of the system power supply and (2) the operator.

In the PDP-11/03, the processor is informed of power supply status by means of the BDCOK H and BPOK H signals (see Section 2.2.1.3). The processor will cause either a Power-Up or a Power-Fail operation to be performed in response to the status of these signals. BDCOK H and BPOK H originate in the H780 power supply and their exact sequence and timing details are contained in The Microcomputer Handbook. In general, BPOK H is the last signal to be asserted in a Power-Up sequence and the first signal to go passive in a Power-Fail sequence. When BDCOK H is passive, indicating the lowest state of the system power supply, it asserts BINIT L and forces the microprocessor Control chip to the Reset state. When the processor is in the Run state, the change of BPOK H from active to passive will cause a Power-Fail trap to be performed. The Power-Up mode is determined via jumper configuration on the processor module. The means by which the Processor interprets the jumper configuration and the power supply status signals are further detailed in the control flow diagram of Figure 2.16-2 in the Machine Operation section (Section 2.3).

Operator Control over the LSI-11 processor is achieved through two means: (1) the state of the front panel Run/Halt switch and (2) Console ODT. Placing the Run/Halt switch in the Halt position causes a Halt interrupt which passes control to the microprogrammed ODT routine. Once the processor has entered the Console ODT/Halt state, the Run state may be reentered by operator execution of the "P" or "G" commands. When the Run/halt switch is in the halt position and the "P" command is repeatedly issued, single-step program execution is achieved. A complete description of Console ODT is in The Microcomputer Handbook.

As shown in the Figure 2.4, the contents of the LSI-11 Processor Status Word (PSW) register have been divided up and allotted to two areas. The four ALU condition code flags appear in the Machine Instruction execute sub-area and the Trace Trap and External Inerrupt Enable flags appear in the Trap & Interrupt Service sub-area. The complete PSW, which the operator may access by either the "RS" ("$S") Console ODT command or under program control via the MFPS or MTPS machine instruction, is illustrated in Figure 2.5. The four lower flags are conditionally set as a result of any processor operation which manipulates data in the ALU or moves data within the LSI-11 Machine. Data moved between a memory location and a device register will affect the condition codes as will the execution of an arithmetic or logical operation. The specific condition code functions for each machine instruction is found in The Microcomputer Handbook. The Machine Instruction Execute area performs the operations dictated by the fetched machine instruction. All members of the LSI-11 Machine Instruction Set may be classified in the three following groups:

1. Data Manipulation Instructions

2. Program Control Instructions

3. Processor Status word Control Instructions

This instruction group includes all single and double operand instructions with the exception of the PSW operators MFPS and MIPS. All instructions in this group set or reset the ALU condition codes as a result of the operation performed. None of the instructions in this group can change the processor priority, PSW BIT 7, or the trace trap enable, PSW BIT 4.

```
---------------------------------
!           !   !   !   !   !   !
!(P)RIORITY ! T ! N ! Z ! V ! C !
!       #   !   !   #   !   !   !
---------------------------------
   7   6   5   4   3   2   1   0
```

P: EXTERNAL INTERRUPT ENABLE (BIT <7> ONLY)

T: TRACE TRAP ENABLE

N: NEGATIVE

Z: ZERO

V: OVERFLOW

C: CARRY

Processor Status Word (PSW)

Figure 2,5

These instructions are listed below.

Single Operand

General: CLR(B), COM(B), INC(B), DEC(B), NEG(B), TST(B) Shift & Rotate: ASR(B), ASL(B), ROR(B), ROL(B), SWAB Multiple Precision: ADC(B), SBC(B), SXT

Double Operand General: MOV(B), CMP(B), ADD, SUB Logical: BIT(B), BIC(B), BIS(B), XOR

The KEV11 EIS/FIS (Extended/Floating Instruction Set) adds four fixed point and four floating point instructions to the group.

Extended Fixed Point
MUL, DIV, ASH, ASCH

Floating Point
FADD, FSUB, FMUL, FDIV

THE PROGRAM CONTROL INSTRUCTIONS ARE DIVIDED INTO TWO SUB-GROUPS, DEPENDING ON WHETHER THE PSW CONTENTS ARE AFFECTED. THE EXECUTION BY THE PROCESSOR OF ANY INSTRUCTION IN THE FIRST SUB-GROUP HAS NO EFFECT ON THE PSW contents. This sub-group includes all Branch, Jump & Subroutine, and Miscellaneous instructions.

Branch: BR, BNE, BEQ, BPL, BMI, BVC, BVS, BCC, BCS

Signed Conditional Branch: BGE, BLT, BGT, BLE

Unsigned Conditional Branch: BHI, BLUS, BHIS, BLO

Jump & Subroutine: JMP, JSR, RTS

Miscellaneous: HALT, WAIT, RESET, SOB

The second sub-group of Program Control instructions is executed in the Trap & Interrupt Service area shown in Figure 2.4. These instructions can control every working bit in the PSW by moving a byte to the PSW register from a vector location or from the stack.

Trap & Interrupt: EMT, TRAP, BPT, IOT, RTI, RTT

Processor Status Word Control Instructions direct control over the PSW register contents. The condition code operators may be used to set or clear any combination of the condition code flags. The NOP instruction is also included here.

Condition Code Operators
Clear: CLC, CLV, CLZ, CLN, CCC
Set: SET, SEV, SEZ, SEN, SEC

NOP

Two single operand instructions belong to this group because ~~of~~ their execution affects the PSW register contents.

Processor Status Word Operators
MFPS
MTPS

The MFPS (Move byte From Processor Status word) instruction transfers the PSW register contents to the destination contained in the instruction. If the destination is mode 0, PSW BIT 7 is sign extended through the upper byte of the register. However, the movement of the PSW contents through the processor to the destination will modify the information in the PSW register according to the following rules.

N	=	Set if PSW Bit <7> = 1; cleared otherwise
Z	=	Set if PSW<7:0> = 0; cleared otherwise
V	=	cleared
C	=	not affected

The MTPS (Move byte To Processor Status word) instruction transfers the 8 bits of the source operand to the PSW register. All working bits may be set or cleared, except the Trace Trap Enable (PSW Bit <4>) which may only be cleared.

The LSI-11 machine instruction set contains four additional reserved instruction groups which have no assigned mnemonic:

The 21R opcode (where R is a 0 - 7) causes the contents of the internal temporary registers to be transferred to consecutive locations pointed to by the contents of register R. Register R is not restored at the end of execution. The internal registers accessed by this instruction are illustrated in Figure 2.6, which also shows their relationship to the Processor control functions.

This instruction is used for diagnostic purposes only and belongs to the Data Manipluation group listed above.

Machin-level execution of instructions in the range of 220-227 causees control to be transferred to the microinstruction located at microaddress 3000 (in user control store). If control store is not present (or if it is disabled) a reserved instruction trap through memory location 10 will occur. (See the Microcomputer Handbook for a description of illecal instruction traps).

Instructions in the range 075040 through 075777 cause control to be transferred to the microinstruction located at microaddress 3003 (in user control store). If control store is not present (or if it is disabled), a reserved instruction trap through memory location 10 will occur.

```
  -------------------------------------------------------------
  |                    PROCESSOR CONTROL                       |
  |                    =================                       |
  |                                                            |
  |           POWER SUPPLY STATUS    POWER-UP OPTIONS          |
  |           -------------------    POWER-FAIL TRAP           |
  |                                                            |
  |           OPERATOR CONTROL       RUN/HALT SWITCH           |
  |           ----------------       CONSOLE ODT               |
  |                                                            |
  |------------------------------------------------------------|
  |                              |                             |
  |  ADDRESS GENERATION          |    INSTRUCTION EXECUTION    |
  |  ==================          |    =====================    |
  |                              |                             |
  |  INSTRUCTION ADDRESS         |    INSTRUCTION TYPES        |
  |  -------------------         |    -----------------        |
  |                              |    DATA MANIPULATION        |
  |                              |    PSW CONTROL              |
  |                              |    PROGRAM CONTROL          |
  |                        -------------                       |
  |  PROGRAM              | FETCH       |                      |
  |  COUNTER (R7)    ---  | MACHINE     | ---> (INSTRUCTION    |
  |                       | INSTRUCTION |            REGISTER) |
  |                        -------------                       |
  |  OPERAND ADDRESS             |                             |
  |  ---------------             |   ARITHMETIC LOGIC UNIT     |
  |                              |      CONDITION CODES        |
  |                              |       PSW BITS 3-0          |
  |                              |      (PSW REGISTER)         |
  |                              |                             |
  |  GENERAL PURPOSE             |  SOURCE OPERAND             |
  |  REGISTERS (R0-R5)           | (SOURCE OPERAND REGISTER)   |
  |                              |                             |
  |                              |  DESTINATION OPERAND        |
  |                              | (DESTINATION OPERAND        |
  |                              |                  REGISTER)  |
  |                        -------------                       |
  |  STACK                 |   TRAP  &  |                      |
  |  POINTER (R6)          | INTERRUPT  |                      |
  |                        |  SERVICE   |                      |
  |                        | PSW BIT 7  |                      |
  |                        | PSW BIT 4  |                      |
  |                        -------------                       |
  |  (BUS ADDRESS REGISTER)      |                             |
  |------------------------------------------------------------|
  |                   SYSTEM BUS CONTROL                       |
  |                   ==================                       |
  -------------------------------------------------------------
```

Processor Control Functions (With Internal Registers)
Figure 2.6

The availability of Writeable Control Store enables the user to design unique machine instructions. These instructions are based on the 0767XX operation code assignment, as shown in Figure 2.7.

The execution of any of this type of instruction causes control to be transferred to the microinstruction located at address 3001 where the microprocessor begins execution of the user-microprogrammed routine. Note that all instructions of the 076XXX format transfer control to the same microlocation (3001), but only opcodes in the range 076700 to 076777 may be utilized for user instructions. The lower six bit positions may then be employed by the user to differentiate between user instructions or to carry data to the microprocessor.

The Address Generation area serves both the Instruction Address and Operand Address generation functions. Instruction addressing employs dedicated register R7 as the Program Counter (PC) and increments the counter by the number of word addresses required by the machine instruction currently under execution. Instruction addressing may also be modified by Program Control instructions, Trap & Interrupt Service, Power-Up routines, or by operator intervention through Console ODT.

Operand Address generation supports the eight General Purpose Register addressing modes and the four Program Counter addressing modes for determing the source and destination operands. Register R6, dedicated to Stack Pointer use, is employed by the Operand Address generation function, Trap & Interrupt Service operations, and by the Jump & Subroutine machine instructions in the Program Control group.

2.2.5 The LSI-11 Writeable Control Store

The User-Microprogrammed machine instruction format (illustrated in Figure 2.7) transfers control to the user control store area (writeable Control Store). The user control store area is composed of random access Read/Write semiconductor memory which contains the user-programmed microinstructions to be accessed by the LSI-11 microprocessor. The interconnection between the LSI-11 processor module and the Writeable Control Store (WCS) module is shown in Figure 2.8. The WCS module is connected to the LSI-11 machine in two ways:

1. LSI-11 System Bus

2. Micromachine Microinstruction Bus

2.2.5.1 LSI-11 System Bus Connection - This connection is established by the printed circuit contact fingers which insert into the system backplane. The WCS module contains 1024 24-bit microlocations which may be read and written via Programmed I/O operations. The LSI-11 system bus interconnection and WCS module control and data registers are detailed in Chapter 6.

2.2.5.2 Microinstrucion Bus Connection - The connection between the LSI-11 processor control area and the WCS module is established by a special microinstruction bus (MIB) cable. The third MICROM socket on the LSI-11 module (E32 on the M7264), which is usually occupied by the EIS/FIS option, provides access to the MIB. The processor control area sends microlocation address information to the WCS module and the contents of the selected location are returned to processor for use by the microprocessor as a microinstruction. The processor WCS port is Read Only; WCS memory contents can be read but not altered by the LSI-11 processor via this port. The WCS memory performs the same function with respect to the micromachine as do the MICROMs located on the LSI-11 module. The details of microinstruction access are contained in Chapter 3.

2.3 MACHINE OPERATION

2.3.1 Basic Machine Cycle

The basic machine cycle in its simplest form is a repeating sequence of Fetch Machine Instruction - Execute Machine Instruction operations as illustrated in Figure 2.9. The Fetch operation may require one DATI (DATA-IN) bus cycle and the Execute operation requires one or more bus cycles as determined by the instruction being executed.

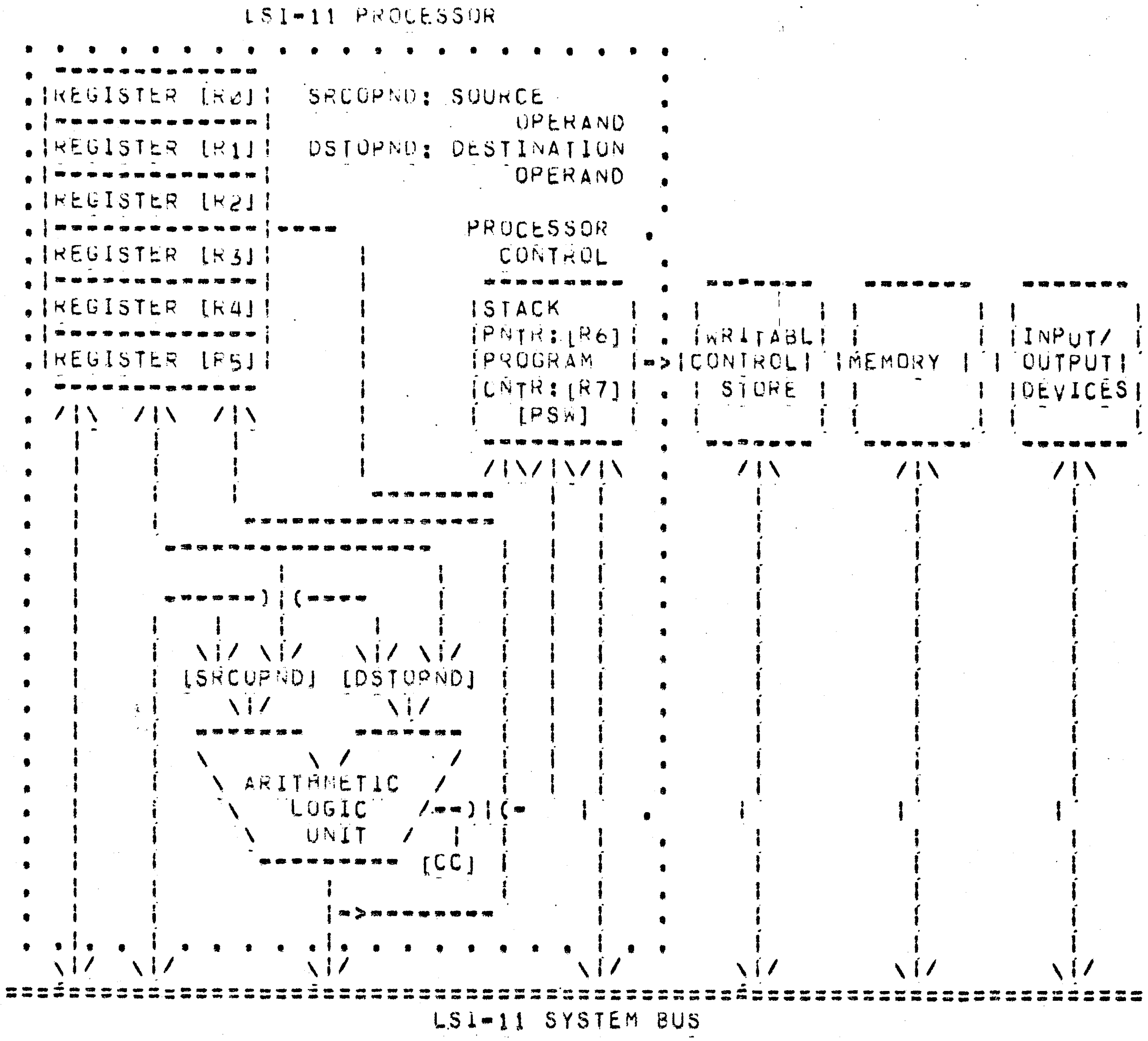
USR
(USER-DEFINED ASSEMBLY MNEMONIC)

```
     USER
     (USER-MICROPROGRAMMED MACHINE INSTRUCTION)

     +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
     !                               |                               !
     ! Ø   1   1   1   1   1   Ø    1   1   1 #Ø/1 Ø/1 Ø/1 Ø/1 Ø/1 Ø/1!
     !+++#+++!+++!+++#+++!+++!+++#+++!+++!+++#+++!+++!+++#+++!+++!+++!
                                         7   6   5   4   3   2   1   Ø
OP CODE OCTAL    :  Ø767ØØ THROUGH Ø76777
OPERATION        :  DESIGNED BY THE USER
```

User-Microprogrammed Machine Instruction Format

Figure 2.7



LSI-11 Processor - Writeable Control Store Interconnection

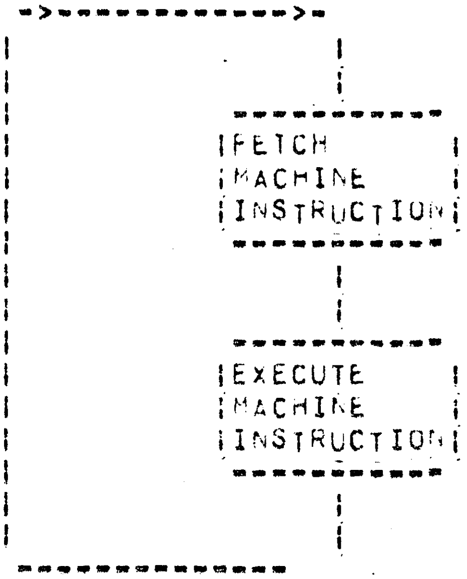
Figure 2.8

2.3.1.1 Bus Error Trap - Implemented with the basic machine cycle is the system bus error trap mechanism, which allows the processor in recovering from a system bus error. A bus error occurs whenever the processor addresses a memory (or device) location which does not exist on the system bus or which does not respond due to a malfunction. The bus error trap is initiated by a timeout sequence which is implemented in the processor bus control circuitry. The bus error condition occurs when no memory or device response is received within 10 microseconds after initiating the bus cycle. A bus error can occur for the Fetch DATI bus cycle as well as for any bus cycles performed during the Execute operation. The trap which responds to the bus error causes the processor to push the PC counter and PSW values onto the stack and to load new values from the trap vector locations (10 and 12). The vector addresses contain the new PC and THE new PSW.

When these operations have been completed, the processor will fetch its next instruction from the location pointed to by the new PC. However, it is also possible that a fetch from the memory location pointed to by the stack pointer will also cause a bus error, resulting in a double bus error condition. The processor response to this condition is to enter the Halt state. The single and double bus error trap operations are illustrated in Figure 2.10.

2.3.1.2 External Interrupt - The basic Fetch-Execute machine cycle can be modified to allow an external event to gain control of the processor, as illustrated in Figure 2.11. Once the execution of a fetched machine instruction is completed, control passes to a decision which interrogates the machine interrupt status. If an interrupt is pending, the processor action is nearly identical to that caused by the trap (the current PC and PSW values will be pushed on the stack and new values loaded from the interrupt vector). The interrupt vector may be automatically known to the processor (as in the Event interrupt case) or the processor can obtain the vector from the interrupting external device (see Section 2.2.1.2).

When the PC and PSW contents are replaced, control is returned to the interrupt decision. This control flow enables the creation of an interrupt (and trap) priority structure that determines which one of a number of simultaneously active interrupts is to receive service. Note that the granting of service to external device interrupts is still dependent upon electrical bus position relative to the processor. Since control is always returned to the top of the decision path, all interrupts are assured of service, in order of decreasing priority, before normal program execution resumes.



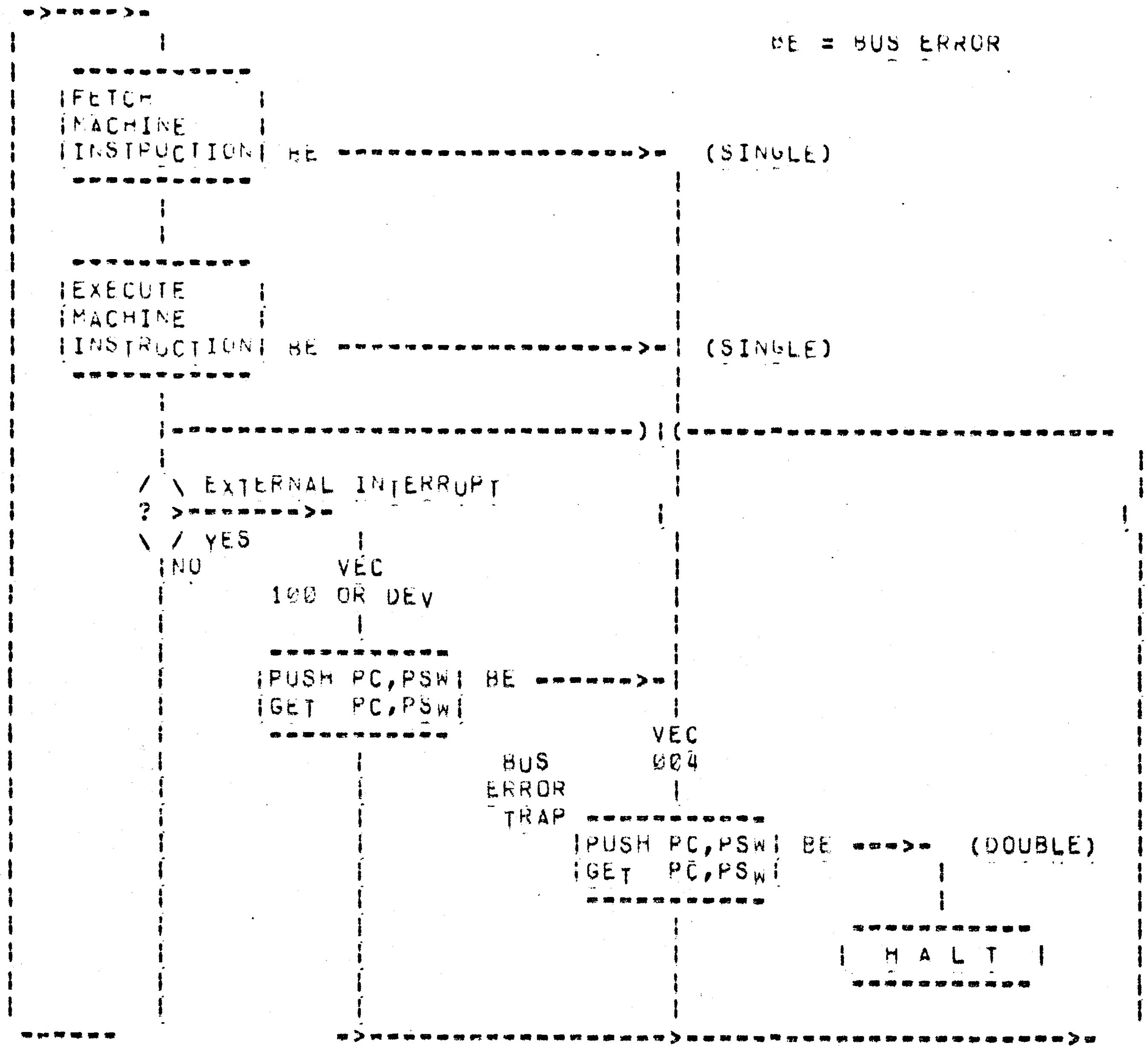
Basic LSI-11 Machine Cycle

Figure 2,9

```
 ->------------>-
 |              |                               BE = BUS ERROR
 |              |
 |         -----------
 |         |FETCH      |
 |         |MACHINE    |
 |         |INSTRUCTION| BE --->-  (SINGLE)
 |         -----------        |
 |              |             |
 |              |             |
 |         -----------        |
 |         |EXECUTE    |      |
 |         |MACHINE    |      |
 |         |INSTRUCTION| BE --->-| (SINGLE)
 |         -----------        |
 |              |             |
 |              |            VEC
 |              |     BUS    004
 |              |    ERROR    |
 |              |     TRAP -----------
 |              |         |PUSH PC,PSW| BE --->-  (DOUBLE)
 |              |         |GET  PC,PSW|        |
 |              |          -----------         |
 |              |             |                |
 |              |             |            -----------
 |              |             |           | H A L T |
 --------------------------------          -----------
```

Single And Double Bus Errors

Figure 2.10



External Interrupt And Bus Error Trap

Figure 2.11

2.3.1.3 Combined Trap and Interrupt Cycle - Because of the similarity between the interruct and trap operations, a single microprogrammed routine implements the required functions. The vector information is stored in an internal register before the routine is entered. An additional flag internal to the processor indicates the double bus error condition (see Section 8.9). The combined interrupt and trap control flow diagram is illustrated in Figure 2.12.

2.3.2 Complete Machine-Level Operating Cycle

The essential principles of the control flow configuration illustrated in Figure 2.12 may be summarized as follows:

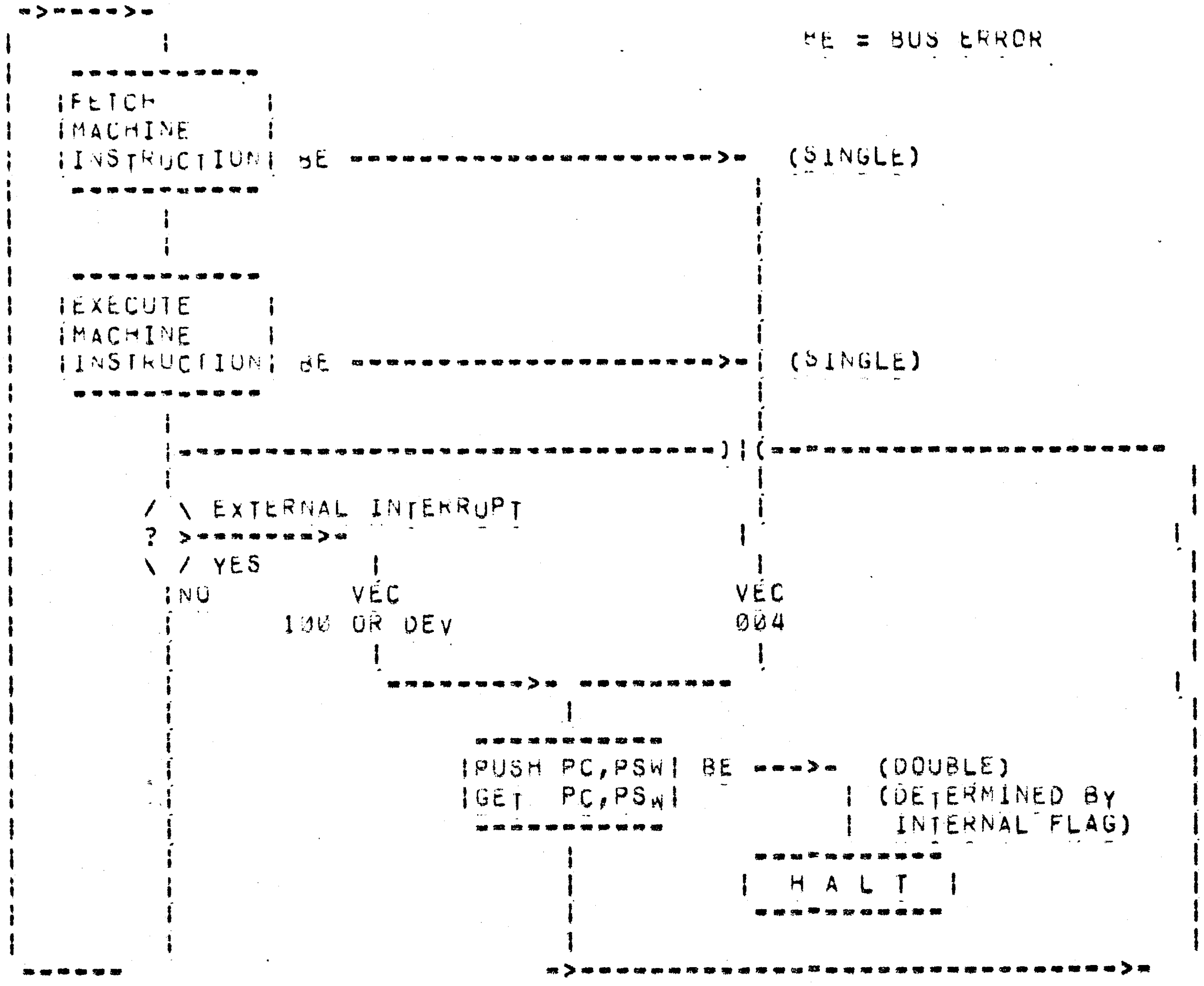
1. The untrapped, uninterrupted machine cycle is a repeating sequnce of Fetch Machine Instruction - Execute Machine Instruction operations.

2. The trap facility allows the processor to recover from a (single) bus error condition.

3. The control flow configuration of the interrupt and trap operations, in conjunction with the hardware stack, implements the interrupt priority hierarchy.

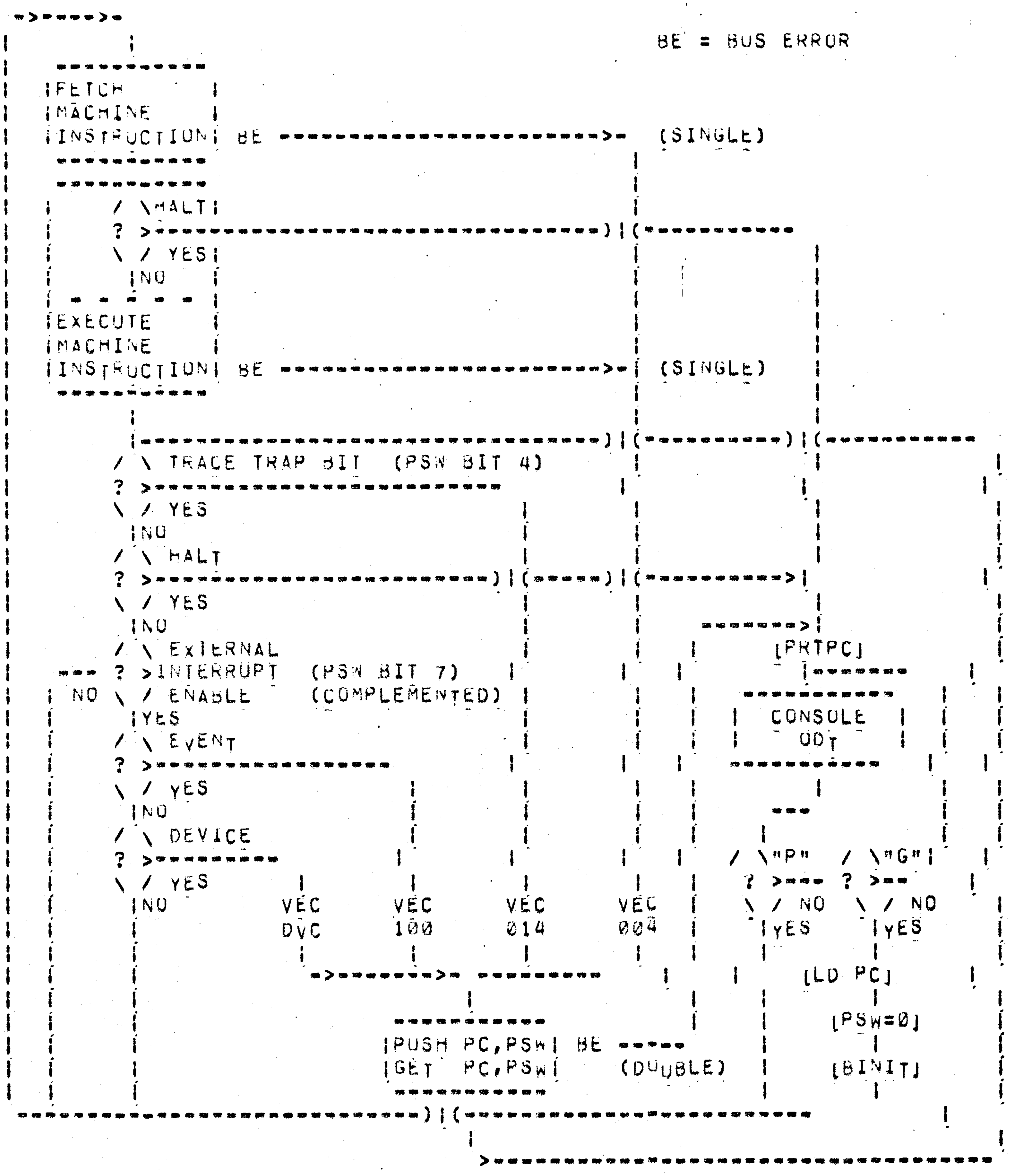
These principles are also apparent in Figure 2.13, which illustrates a more exact machine-level control flow diagram. The complete diagram must detail the transfer of control between the machine and micromachine levels and is presented in a later section. Figure 2.13 illustrates the Fetch-Execute-Trap/Interrupt machine cycle which is sufficient for conventional machine level programming. It shows the location in the control flow of the Trace Trap Bit (PSW BIT <4>), and the External Interrupt Enable Bit (PSW Bit <7>). Also shown are the two paths to the Console ODT/Halt state as well as the two paths which leave ODT and re-enter the Fetch-Execute cycle of the Run state.

2.3.2.1 Run/Halt Portion - The Run/Halt portion of Figure 2.13 is extracted and illustrated in Figure 2.14. The two means of entering the Halt state are: (1) the execution of the Halt machine instruction and (2) the assertion of the Halt interrupt. The latter is asserted via the bus control signal BHALT L by setting the front panel Run/Halt switch to Halt or by pressing the console terminal BREAK key. The PC contents are printed on the terminal immediately upon entering the Halt state. This gives the location of the next instruction to be just executed. Either the "P" or "G" commands may be entered by the operator at the console terminal. Entering the "P" (PROCEED) command passes control directly to the Machine Instruction Fetch operation. The "G" (GO) commandloads a new PC value (nnnnnnG) and zeroes the PSW (which enables external interrupts) before passing control to the Machine Instruction Fetch operation.

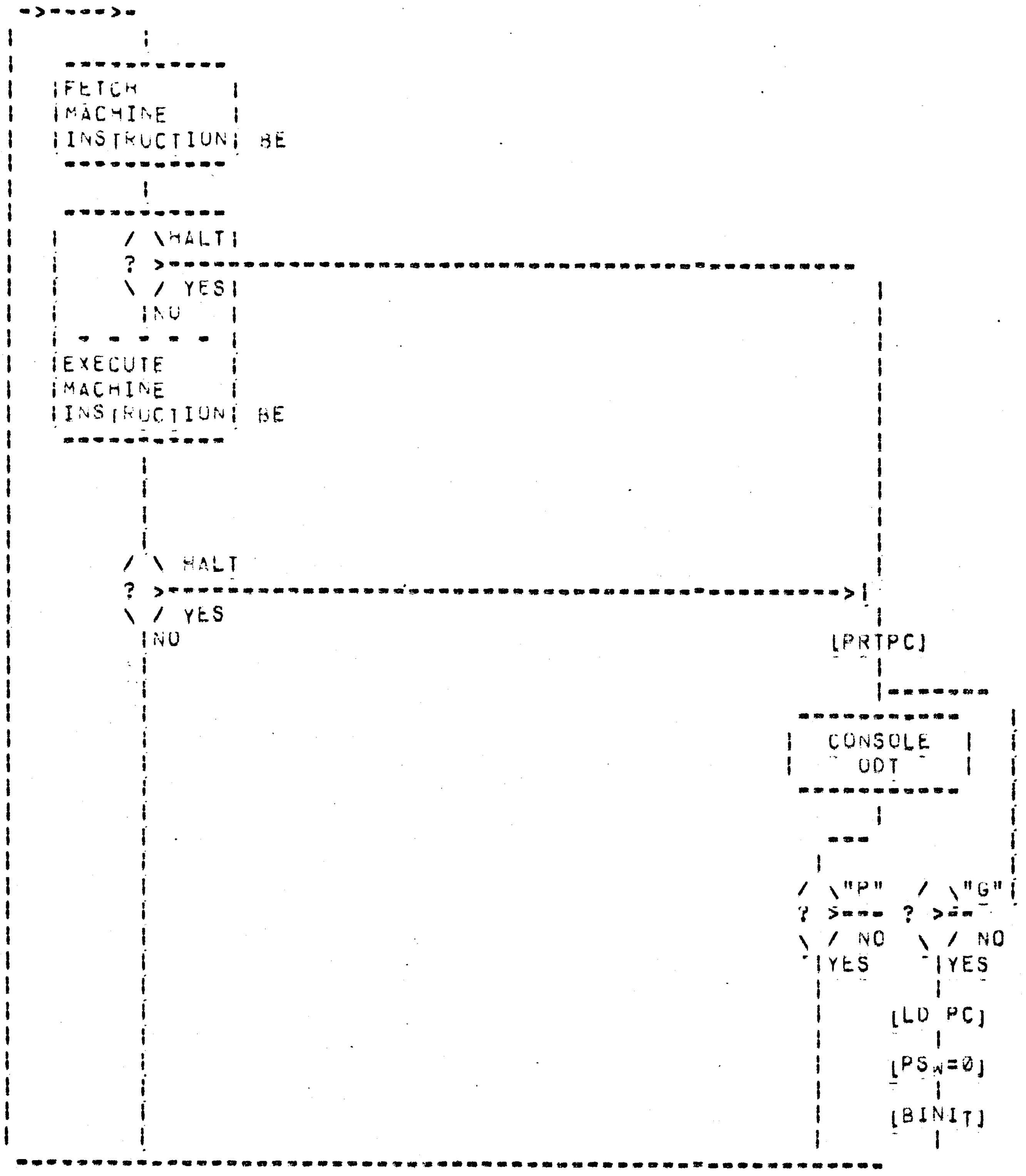


Combined Interrupt And Trap Operations

Figure 2.12



Complete Fetch - Execute - Interrupt Cycle
Figure 2.13



Run/Halt Portion of Machine Cycle
Figure 2.14

2.3.2.2 Trap/Interrupt Portion - The Trap/Interrupt portion of Figure 2.13 is extracted and illustrated in Figure 2.15. The Trace Trap has the highest machine-level priority and affects control flow before any external event. The Trace Trap uses the same vector value, (014), as the breakpoint Trap(BPT) instruction. The hardware Trace Trap, implemented via PSW BIT 4, and the software Trace Trap, implemented via the execution of the BPT instruction, are used to support program debugging.

The External Interrupt Disable PSW Bit <7=1>) can divert control flow around Event and device interrupts. When enabled, the Event interrupt, asserted via the bus signal BEVNT L, will receive service before any device interrupts. All external devices having interrupt capability assert the same interrupt request line, BIRQ L. Interrupt priority external to the processor is determined by the position of the module in the LSI-11 backplane.

2.3.3 Complete Machine-Micromachine Operating Cycle

The complete control flow diagram which makes apparant the transfer of control between the machine and micromachine levels is illustrated in Figures 2.16-1 and 2.16-2. A greater number of machine instruction examples are used to represent the decisions made during the Execute Machine Instruction operation of the basic machine cycle. Many of the examples used demonstrate the internal sharing of the microprogrammed Trap/Interrupt service routine.

2.3.3.1 Bus Error Processing - The entry point at the top of the Power-Up decision flow in Figure 2.16-2 is the result of a hardware reset in the case of a bus error. A wait state occurs due to an unresponding bus device, but the wait is terminated by a 10 microsecond timer on the LSI-11 that CPU module resets the microprocessor. When reset, the microprocessor begins executing microinstructions at microlocation 0001. The FDIN (Fast Data-In) operation is used to to determine how control flow arrived at that entry point, either by bus error or by Power-Up. If a bus error was the cause, only one of the 6 possible bus error types will result in a trap through vector location 004. The first possible bus error is used by a microprogrammed ODT routine to determine memory size (Boot Self-Size). The second bus error type occurs when the operator attempts to examine (using console ODT) a memory or device register which does not respond. In this case, control is returned to a point within the ODT microcode and a "?" is printed on the console terminal. The next three bus error types are regarded as fatal and result in a processor Halt. These errors occur (1) when an interrupting device does not return a vector, (2) when a microprogrammed refresh does not receive a reply), or (3) when a double bus error occurs.

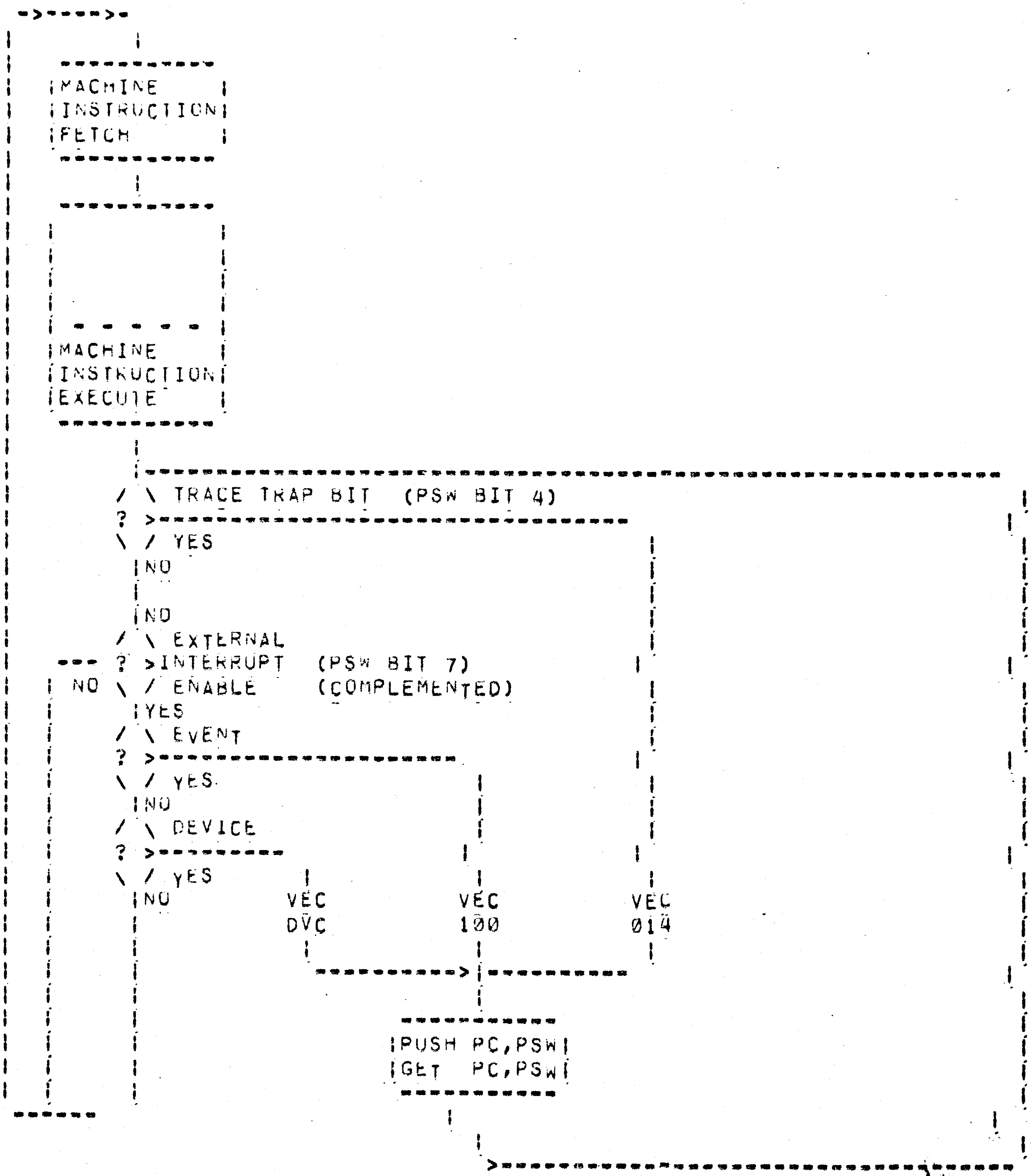
2.3.3.2 Trap/Interrupt Processing - The Trap/Interrupt decision flow indicates the priority with which the micromachine interrupt register is interrogated. This internal micromachine interrupt register is illustrated in Figure 2.17. Of the seven interrupts, four are external and three are internal. An internal interrupt in this context is one which can be set or reset only under microprogram control.

The highest priority interrupt is I6 which is used only at the micromachine level to determine whether an external interrupt (I2: Event, I3: Device) is pending. This facility enables a lengthy microroutine to abort execution and grant interrupt service to the external Event or Device interrupts only. If I2 or I3 is asserted, while I6 is enabled, control is transferred to microlocation 3004 from any microinstruction which has the RSVC bit (bit <17>) set to a "1" after the next subsequent microinstruction is executed (only if neither one of the microinstructions is a Jump or Return from Subroutine microinstruction).

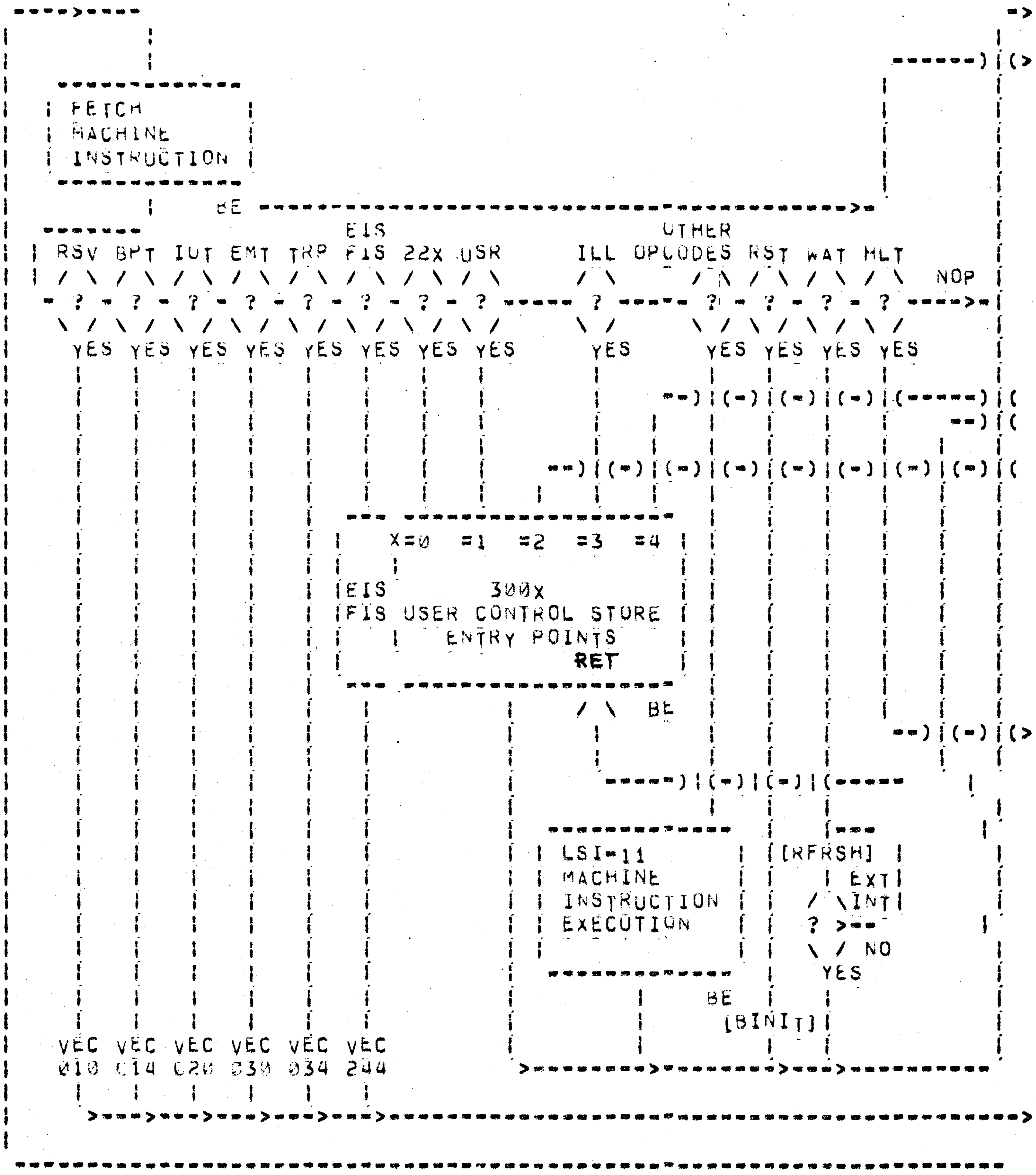
```
 ---  ---  ---  ---  ---  ---  ---
|    |    |    |    |    |    |    |
| 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|----|----|----|----|----|----|----|
```

LISTED IN ORDER OF DECREASING PRIORITY

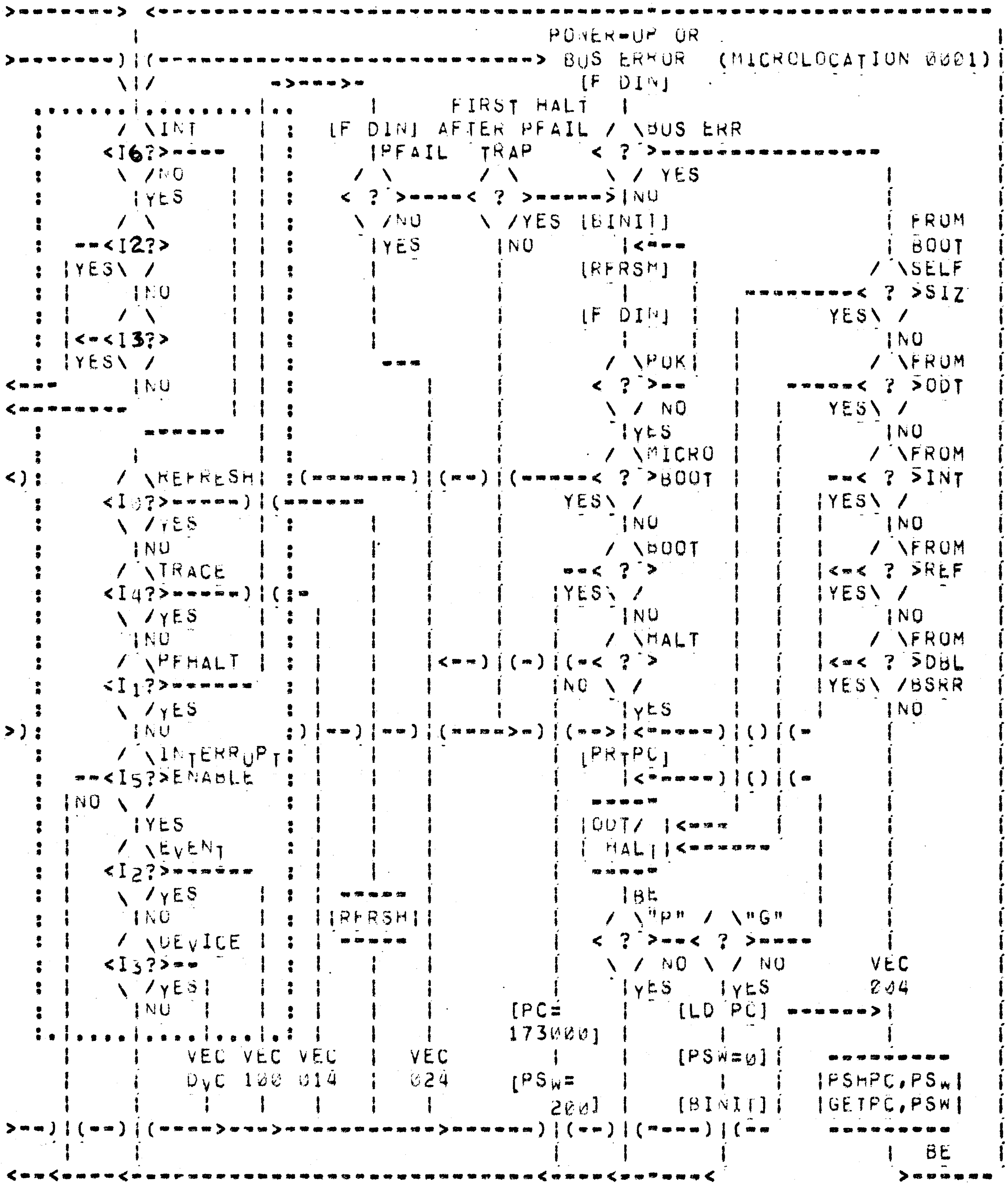
```
I6:  INTERNAL    TEST FOR INTERNAL INTERRUPTS I2 OR I3
I0:  EXTERNAL    DYNAMIC MEMORY REFRESH
I4:  INTERNAL    TRACE TRAP BIT  (PSW BIT-4)
```


MACHINE INSTRUCTION FETCH
MACHINE INSTRUCTION EXECUTE
TRACE TRAP BIT (PSW BIT 4)
YES
NO
NO
EXTERNAL INTERRUPT ENABLE
(PSW BIT 7) (COMPLEMENTED)
NO
YES
EVENT
YES
NO
DEVICE
YES
NO
VEC DVC
VEC 100
VEC 014
PUSH PC,PSW
GET PC,PSW

Interrupt And Trap Portion of Machine Cycle
Figure 2.15



Complete Machine-Micromachine Control Flow Diagram
Figure 2.16-1



Complete Machine-Micromachine Control Flow Diagram

Figure 2.16-2

2.3.3.3 Power-Up Processing - If the top decision in the Power-Up flow determines that no bus error occured, a Power-Up routine begins. The first event is to issue an initialization signal on BINIT L and then wait for bus power to come up (BPOK H active). If enabled, dynamic memory refresh will also occur in this sequence. When bus power arrives, the jumpered Power-Up option is accessed by the microprocessor through the FDIN operation and performed. The four possible Power Up modes are listed below:

MODE 0 - The PC counter and the PSW are loaded from locations 24 and 26, respectively and machine execution begins. If BHALT L is asserted, control will be returned to Console ODT/Halt.

MODE 1 - Control passes immediately to Console ODT/Halt.

MODE 2 - The processor PC is loaded with 173000, the PSW with 200 (External Interrupts Disabled), and Machine execution begins. As in MODE 0, the processor will halt before the first instruction is executed if BHALT L is asserted.

MODE 3 - Control is immediately goes to microlocation 3002, the control store entry point for a user-microprogrammed bootstrap routine. The status of BHALT L has no effect on this control transfer. If control store does not exist at that microaddress, a trap to vector location 10 occurs.

i1:	external	power-fail/halt interrupt
i5:	internal	enable external interrupts i2 and i3 (complement of psw bit-7)
i2:	external	event interrupt
i3:	external	device interrupt

micromachine Interrupt Register

Figure 2.17

The Refresh interruct (I7) has the highest priority of all interrupts external to the micromachine. When enabled via a jumper on the LSI-11 processor, I7 is asserted every 1.6 milliseconds. This interrupt is usually transparant to the machine-level operation of the processor.

The refresh (RFRSH) operation which follows I7 has control transfer links to the WAIT (WAT) machine instruction, to the FDIN-POK sequence and to the UPI/Halt routine. These links exist by means of special translations which are implemented in the microprocessor Control chip.

The Trace Trap Bit (I4), the External Interrupt Enable Bit (I5) and the two external interrupts, Event (I2) and Device (I3), are unchanged from their representation in Figure 2.15. However, the external Halt interrupt shown in the earlier figure is shared with the Power-Fail function (PFHALT). The micromachine employs a Fast Data-In operation to determine which event has occured.

The Trap/Interrupt priority structure is the composite result of the internal micromachine interrupt register priorities, the microprogrammed Power-Up and bus error routines, and the over-all control flow configuration. The combined priorities may be ordered as follows:

1. Bus Error Trap
2. External Interrupt Test (I6)
3. Memory Refresh
4. Machine Instruction Traps
5. Hardware Trace Trap
6. Halt Line
7. Power-Fail Trap
8. Event Line Interrupt
9. Device (Bus) Interrupt Request

CHAPTER 3

THE LSI-11 MICROMACHINE STRUCTURE

3.1 GENERAL

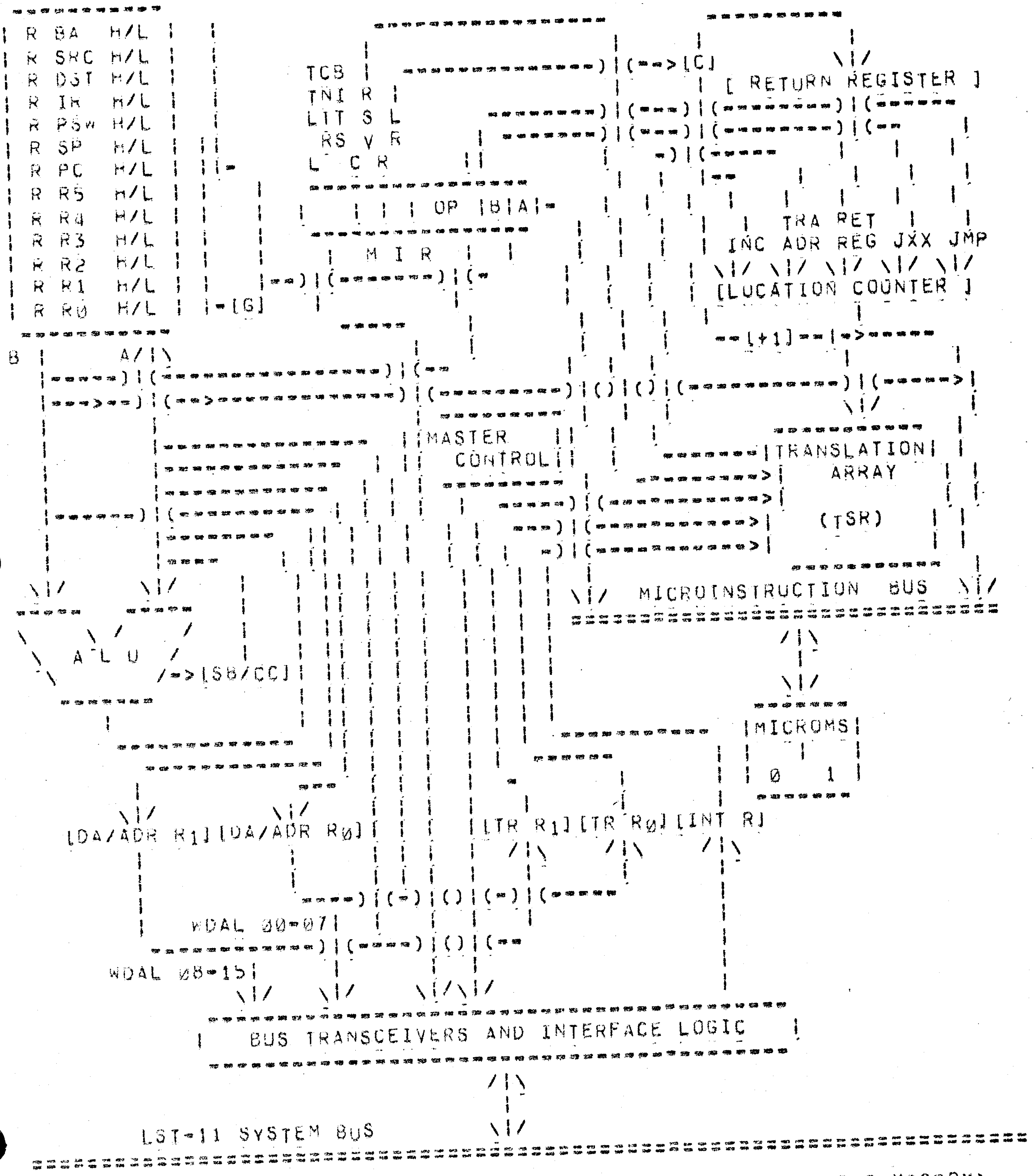
The LSI-11 processor illustrated in Figure 2.3 is implemented by a Large-Scale-Integrated microprocessor which is microprogrammed to emulate the PDP-11 architecture. Emulation is the technique whereby the general resources of the microprocessor are made to serve as the specific architectual components (GP registers, 16-bit ALU, etc.) of the LSI-11. This chapter contains a detailed description of the microprocessor which is made up of the control chip and the data chip on the LSI-11 module.

For the purposes of microprogramming, it is useful to view the control and data chips as a single microprocessor. That the entire microprocessor can be implemented on only two 40-pin LSI chips is due to efficient partitioning of the microprocessor functions. The major interconnection path between the two chips is the MicroInstruction Bus (MIB). In addition to providing a common bus for the micromachine instructions, the MIB is also time-multiplexed to provide auxilliary control paths between the Data, Control, and MICROM chips.

3.2 THE MICROPROCESSOR

The complete microprocessor is illustrated in Figure 3.1. There are several similarities which may be drawn between this illustration and Figure 2.3 (LSI-11 Processor). Both figures contain a register file and an arithmetic logic unit and there is also a memory component evident in both figures. Figure 2.3 contains the system memory while Figure 3.1 contains the MICROMs. However, the programs contained in the MICROMS (along with the facilities of the translation array) efficiently organize the resources of the microprocessor to achieve an emulation of the machine-level architecture shown in Figure 2.3. An example of the differences in the two architectures may be seen in the register files. All six of the general purpose registers in the LSI-11 processor are 16-bits wide and support both byte and word addressing. However, the register file in the microprocessor is composed of 26 8-bit bytes which support a combination of direct and indirect addressing modes. The register labels correspond to the six general purpose registers, the stack pointer, the program counter, and the five internal registers indicated in Figure 2.6.

There is a master control section in Figure 3.1 which is roughly analogous to the processor control section in Figure 2.3. All micromachine instructions, whether fetched from MICROM or from user control store in the KCS module, are loaded into the microinstruction register for execution at the micromachine level. The two Data/Address (DA/ADR) registers, the two Translation (TR) registers and the Interrupt (INT) register provide register interface and buffer functons between the micromachine and the outside world of the LSI-11 system bus. To complete the analogy with Figure 2.3, the LSI-11 system bus is to the microprocessor as the Input/output devices are to the LSI-11 processor.



MICROPROCESSOR CONTROL AND DATA CHIP DETAIL (INCLUDING MICROM)

FIGURE 3.1

Figure 3.2 provides an overview of microprocessor operations. The figure is in the same format as Figure 2.4, (Processor Control Functions). The Compute and Reset controls and the Microlocation Address Generation functions are discussed in Section 3.3.2. The Microinstruction Execution functions are discussed in Section 3.3.1.

3.3 MICROPROCESSOR PARTITIONING

The effectiveness of the microprocessor partitioning is illustrated in Figure 3.3. All chips within the micromachine (Control, Data, MICROMS) receive 2 four-phase micromachine clock from the LSI-11 circuitry. The 22-bit microinstruction bus provides a communication path between all micromachine elements. Sixteen lines of the MIB are common to all three chip types, while MIB<16> and MIB<17> are connected only between the MICROM chips and the Control chip. The last four lines, (MIB<21:18>) lead directly from the MICROM chips to the TTL control logic on the LSI-11 module. The TTL control logic is composed of the Bus Transceivers and Interface Logic shown in Figure 3.1. The operation of this logic is detailed in The Microcomputer Handbook. The signal paths and control functions on the LSI-11 system bus side of this logic were discussed in Chapter 2. The connections on the micromachine side are listed in Figure 3.4.

The Special Control Functions which are generated by the 4 highest MICROM bits (MIB<21:18>) are distributed to the two major logic areas as listed in Figure 3.5.

3.3.1 Microprocessor Data Chip

The components of the microprocessor which are implemented in the Data chip have been extracted from Figure 3.1 and are illustrated in Figure 3.6. The data chip is connected to the sixteen lowest lines of the microinstruction bus (MIB<15:00>) and to the WAIT signal. The data chip's access to the LSI-11 System Bus is provided by WDAL<07:00> and WDAL <15:08>. Microinstructions fetched from MICROM are loaded into the Microinstruction Register (MIR) for execution by the data chip. The MIB also provides a signal path back to the control chip for conditional jump instruction results. The 16 WDAL lines provide bidirectional access to the LSI-11 system bus lines, BDAL L<15:00>. The output path of the WDAL lines is buffered by the two Data/Address Registers, DA/ADR R0 and DA/ADR R1, which hold the output data or address information for the LSI-11 bus drivers.

The major elements of the data chip are listed as follows:

1. Microinstruction Register

2. Register File and Indirect Addressing Register

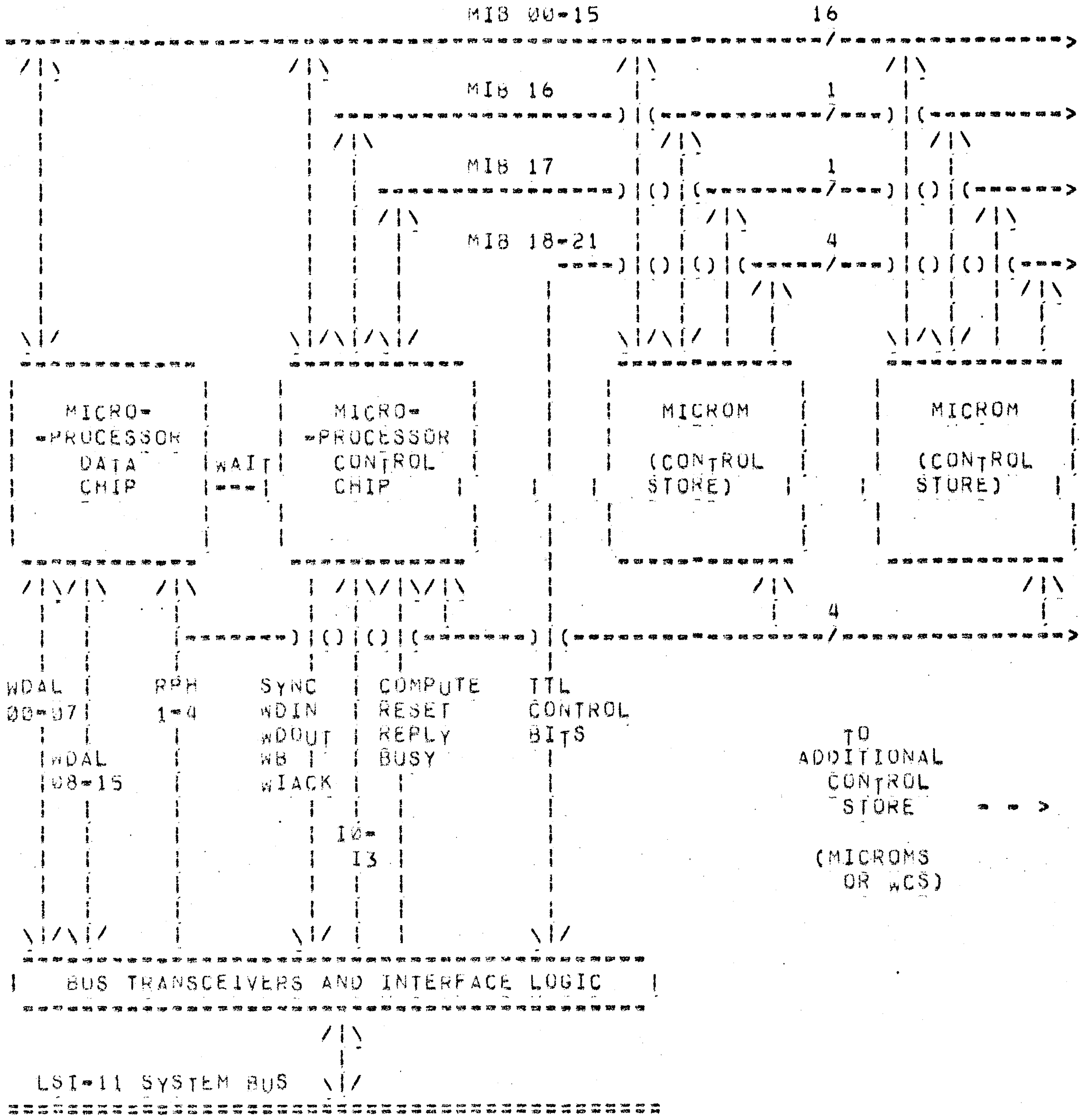
3. Arithmetic Logic Unit

4. Status Bit/Condition Code Flag Register

```
 ---------------------------------------------------------
!                  MICROPROCESSOR CONTROL                 !
!                  ======================                 !
!                                                         !
!           COMPUTE          ALWAYS ASSERTED              !
!           -------                                       !
!                                                         !
!           RESET            POWER-FAIL TRAP              !
!           -----            BUS-ERROR  TRAP              !
!                                                         !
!---------------------------------------------------------!
!                    CONTROL !                       DATA !
!  MICROLOCATION     ------- !     MICROINSTRUCTION  ---- !
!  =============        CHIP !     ================  CHIP !
!  ADDRESS GENERATION   ---- !     EXECUTION         ---- !
!  ==================        !     =========              !
!                            !                            !
!  MICROINSTRUCTION ADDRESS  !     INSTRUCTION TYPES      !
!  ------------------------  !     -----------------      !
!                            !     DATA MANIPULATION      !
!                            !     DATA ACCESS            !
!                            !     MICROPROGRAM CONTROL   !
!                     -------------                       !
!  LOCATION          ! FETCH       !                      !
!  COUNTER         - ! MICROMACHINE!-> MICROINSTRUCTION  !
!                    ! INSTRUCTION !   REGISTER (S)       !
!  INCREMENTER       ! (MICROFETCH)!                      !
!  RETURN REGISTER    -------------                       !
!  UNCONDITIONAL JUMP        !   ARITHMETIC LOGIC UNIT    !
!  CONDITIONAL   JUMP        !                            !
!                            !                            !
!                            !      STATUS  BIT FLAGS     !
!  MACHINE INSTRUCTION       !   CONDITION CODE FLAGS     !
!  TRANSLATION REGISTER      !                            !
!           !                !                            !
!          \ /               !       REGISTER FILE        !
!     -------------          !                            !
!    ! TRANSLATION !         !            26              !
!    !    ARRAY    !- I0-I6 !         8-BIT WORDS        !
!    !    (TSR)    !         !                            !
!     -------------          !                            !
!           !                !                            !
!          \ /               !                            !
!  TRANSLATION ADDRESS       !                            !
!                            !                            !
!---------------------------------------------------------!
!              MICROINSTRUCTION BUS CONTROL               !
!              ============================               !
!---------------------------------------------------------!
```

CONTROL CHIP CONNECTIONS TO SYSTEM BUS CONTROL

MICROPROCESSOR CONTROL FUNCTIONS
FIGURE 3.2



MICROPROCESSOR INTER-CHIP WIRING DETAIL

FIGURE 3.3

LSI-11 BUS DRIVER/ RECEIVER INTERFACE	BUS I/O CONTROL SIGNAL LOGIC	INTERRUPT CONTROL AND RESET LOGIC	SPECIAL CONTROL FUNCTIONS
MICROCOMPUTER HANDBOOK: 4.2.2.3	MICROCOMPUTER HANDBOOK: 4.2.2.4	MICROCOMPUTER HANDBOOK: 4.2.2.6	MICROCOMPUTER HANDBOOK: 4.2.2.7
DATA CHIP	CONTROL CHIP	CONTROL CHIP	MICROM
WDAL 00-07 (IN/OUT)	WSYNC (OUT)	RESET (IN)	MIB 18-21 (OUT)
WDAL 08-15 (IN/OUT)	WDIN (OUT)		
	WDOUT (OUT)	WIACK (OUT)	
	WWB (OUT)		
	REPLY (IN)	I0 RFIRQ (IN)	
	BUSY (IN)	I1 IPIRQ (IN)	
		I2 EVIRQ (IN)	
		I3 IOIRQ (IN)	

NOTE - ALL HANDBOOK REFERENCES IN SECTION I : LSI-11 FAMILY HARDWARE

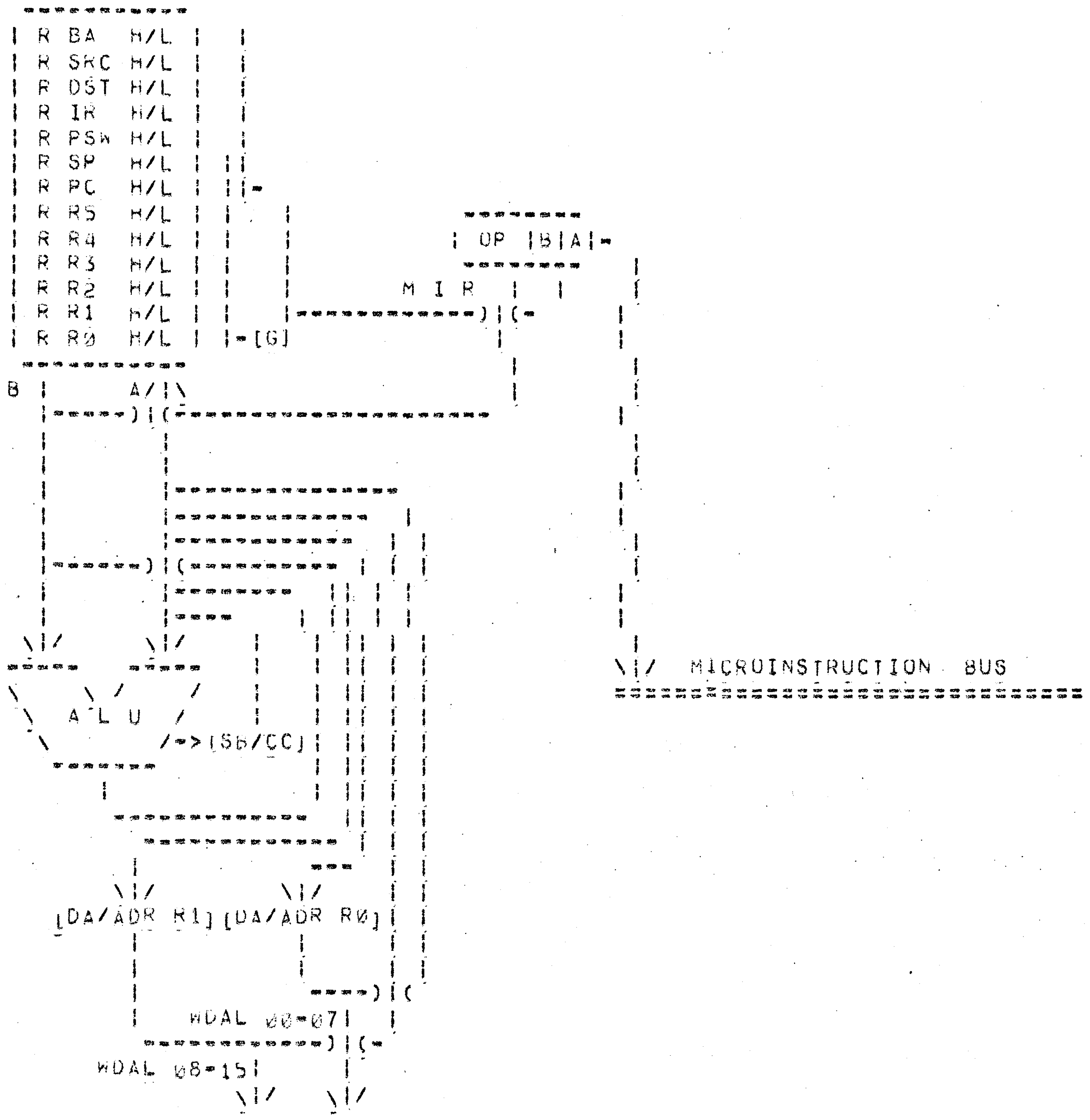
INTERFACE LOGIC REFERENCE CHART

FIGURE 3.4

BUS I/O CONTROL SIGNAL LOGIC	INTERRUPT CONTROL AND RESET LOGIC
INIT (1) L	IFCLR AND SRUN
INIT (1) H	IFCLR L
	RESET L
	FAST DIN
	PFCLR L
	EFCLR L

SPECIAL CONTROL FUNCTION DISTRIBUTION CHART

FIGURE 3.5



MICROPROCESSOR DATA CHIP DETAIL

FIGURE 3.6

3.3.1.1 Microinstruction Register - Micromachine instructions delivered to the data chips are loaded into the MIR for execution. The portion of the MIR contained in the data chip is only 8 bits wide, reflecting the fact that only MicroInstruction Bus lines MIB 00-15 are connected to the data chip. The MIB supports four types of microinstruction opcode formats, as listed below:

1. Jump Format

2. Conditional Jump Format

3. Literal Format

4. Register Format

Jump Format

The Jump format is illustrated in Figure 3.7. The opcode occupies bits <15:12> and is equal to 0 (0000). The jump address is contained in bits <10:00> and this 11-bit jump address can transfer micromachine control to any of the 2048 microlocations addressable within the control store.

The second of the two possible Jump format instructions is determined by bit-11 of the microinstruction. When bit 11 is a 1, the microinstruction executes a Return From Subroutine (RFS) operation. The LSI-11 microprocessor supports only one level of subroutine at the microprogramming level.

Conditional Jump Format

The conditional Jump format is shown in Figure 3.8. The conditional jump microinstruction is differentiated from the previous (unconditional) jump microinstruction by the contents of the opcode field. Conditional jumps are indicated when bits <15:12> are a 1 (0001). The condition field, bits <11:8>, indicate which condition must be satisfied for the jump to take place. The condition tests the ALU status bit and condition code flags as well as an indirect condition Status bit. If jump conditions are satisfied, micromachine control is transferred to the microlocation contained in the 8-bit address field. Since only 8 bits are available for address information, control may be transferred to only locations within the current 128 location page. The remaining 3 address bits which were evident in the (unconditional) jump microinstruction are determined by the (updated) value of the current location counter.

JUMP FORMAT

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|                         |      |                |                                          |
|  0     0     0     0    |1/0|                   JUMP ADDRESS                               |
|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++|
  15    14    13    12    11    10     9     8     7     6     5     4     3     2     1     0
```

(UNCONDITIONAL) JUMP FORMAT

FIGURE 3.7

CONDITIONAL JUMP FORMAT

```
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|                         |                       |                                          |
|  0     0     0     1    |      CONDITION        |             JUMP ADDRESS                 |
|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++#+++|+++|+++|
  15    14    13    12    11    10     9     8     7     6     5     4     3     2     1     0
```

CONDITIONAL JUMP FORMAT

FIGURE 3.8

Literal format

The Literal Format microinstruction, illustrated in Figure 3.9, provides a means for generating constants within a microprogram. The contents of the 8-bit literal field are determined at the time of microprogram assembly and may not be changed by the execution of any microinstruction. The remaining field, bits <3:0>, is called the A field and determines which microprocessor register is accessed through the "A" register port.

Register format

The Register Format microinstruction, illustrated in Figure 3.10, contains a second register field, the "B" field. The B port on the microprocessor register file is a Read Only port, whereas the A port is a Read/Write port. Register format microinstructions are either byte or word instructions. In the case of a byte operation, the microinstruction executes in one micromachine cycle and the operands are determined by the contents of the A and B register fields. A word operation requires two micromachine cycles to execute. The second byte of a word operands is determined by complementing the lowest bit of the register field during the second micromachine cycle. Thus the low byte of a word instruction is supplied by the contents of Ra (or Rb) and the high byte is supplied by the contents of Ra + or - 1 (or Rb + or - 1). The ALU's status bit and condition code flags reflect the result of a word operation in the case of a word microinstruction, just as they reflect the result of a byte operation in the case of a byte microinstruction. Direct and indirect register addressing is supported by both A and B register fields.

3.3.1.2 Register File and Indirect Addressing Register - The register file is composed of 26 8-bit registers which provide temporary storage for data and address information for use by either machine (or micromachine) level programs. These registers supply operands to the ALU without system bus cycles and are therefore accessed at high speed.

The contents of the register file may be accessed either directly or indirectly. One group of registers may be accessed only directly, a second group only indirectly, and a third group may be accessed in either way, as illustrated in Figure 3.11. This illustration uses the convention that directly addressed registers are preceded by "R" and indirectly addressed registers are preceded by "G". Figure 3.11 also indicates the machine- or micromachine-level use of each register.

The three highest bits of the A and B register fields determine whether direct or indirect addressing is in effect. When the three highest bits are all zero, indirect mode is indicated and the G register contributes its three bits to the final, indirect register address. Since the lowest bit in each field may be either a 1 or a 0, indirect word addressing is achieved by complementing the lowest bit during the second cycle of a two-cycle word microinstruction. Figure 3.12 illustrates indirect addressing as used with the Literal microinstruction.

LITERAL FORMAT

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                                                       |                   |
|    OP   CODE      |                 LITERAL FIELD                         |   A   REGISTER    |
|___#___|___|___#___|___|___#___|___|___#___|___|___#___|___|___|
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
```

LITERAL FORMAT

FIGURE 3.9

REGISTER FORMAT

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                           |     |                         |                   |
|                 OP  CODE                  |1/0|    B   REGISTER         |   A   REGISTER    |
|___#___|___|___#___|___|___#___|___|___#___|___|___#___|___|___|
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
```

REGISTER FORMAT

FIGURE 3.10

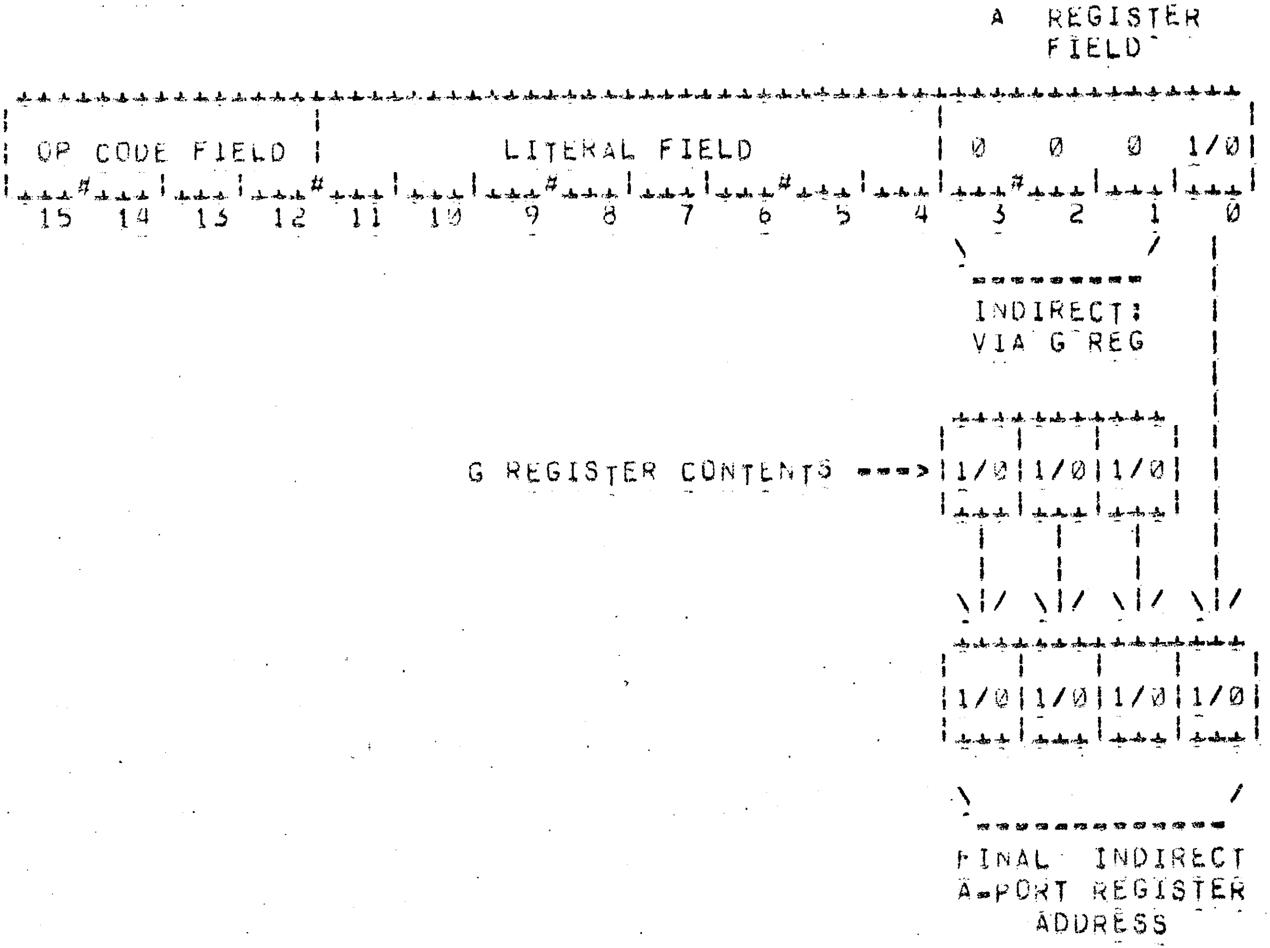
```
PDP-11          DECIMAL        G REG CONTENTS      DECIMAL             PDP-11
 USE              |    OCTAL        BINARY            |    OCTAL         USE
=====             |      |            |               |      |         =======
        -------------------------                -- -- -- -- -- -- --
RØ LO  | (Ø7)  GØØ   GØØ   (ØØ) |     ØØØ       INDIRECT  ADDRESSING    LO BYTE
       |-------------------------|             | -- -- -- -- -- -- -- |
RØ HI  | (15)  GØ1   GØ1   (Ø8) |     ØØØ       INDIRECT  ADDRESSING    HI BYTE
       |-------------------------|             |-----------------------|
R1 LO  | (Ø7)  GØ2   GØ2   (ØØ) |     ØØ1      | (Ø7)  RØ2   RØ2  (ØØ) | RBA   LO
       |-------------------------|             |-----------------------|
R1 HI  | (15)  GØ3   GØ3   (Ø8) |     ØØ1      | (15)  RØ3   RØ3  (Ø8) | RBA   HI
       |-------------------------|             |-----------------------|
R2 LO  | (Ø7)  GØ4   GØ4   (ØØ) |     Ø1Ø      | (Ø7)  RØ4   RØ4  (ØØ) | RSRC  LO
       |-------------------------|             |-----------------------|
R2 HI  | (15)  GØ5   GØ5   (Ø8) |     Ø1Ø      | (15)  RØ5   RØ5  (Ø8) | RSRC  HI
       |-------------------------|             |-----------------------|
R3 LO  | (Ø7)  GØ6   GØ6   (ØØ) |     Ø11      | (Ø7)  RØ6   RØ6  (ØØ) | RDST  LO
       |-------------------------|             |-----------------------|
R3 HI  | (15)  GØ7   GØ7   (Ø8) |     Ø11      | (15)  RØ7   RØ7  (Ø8) | RDST  HI
       |-------------------------|             |-----------------------|
R4 LO  | (Ø7)  GØ8   G1Ø   (ØØ) |     1ØØ      | (Ø7)  RØ8   R1Ø  (ØØ) | RIR   LO
       |-------------------------|             |-----------------------|
R4 HI  | (15)  GØ9   G11   (Ø8) |     1ØØ      | (15)  RØ9   R11  (Ø8) | RIR   HI
       |-------------------------|             |-----------------------|
R5 LO  | (Ø7)  G1Ø   G12   (ØØ) |     1Ø1      | (Ø7)  R1Ø   R12  (ØØ) | RPSW  LO
       |-------------------------|             |-----------------------|
R5 HI  | (15)  G11   G13   (Ø8) |     1Ø1      | (15)  R11   R13  (Ø8) | RPSW  HI
       |-------------------------|             |-----------------------|
SP LO  | (Ø7)  G12   G14   (ØØ) |     11Ø      | (Ø7)  R12   R14  (ØØ) | SP    LO
       |-------------------------|             |-----------------------|
SP HI  | (15)  G13   G15   (Ø8) |     11Ø      | (15)  R13   R15  (Ø8) | SP    HI
       |-------------------------|             |-----------------------|
PC LO  | (Ø7)  G14   G16   (ØØ) |     111      | (Ø7)  R14   R16  (ØØ) | PC    LO
       |-------------------------|             |-----------------------|
PC HI  | (15)  G15   G17   (Ø8) |     111      | (15)  R15   R17  (Ø8) | PC    HI
        -------------------------               -----------------------
            |             /|\                       |           /|\
            |              |                        |            |
         B  | PORT      A  | PORT                B  | PORT    A  | PORT
            |              |                        |            |
           \|/            \|/                      \|/          \|/

        READ ONLY    READ/WRITE               READ ONLY    READ/WRITE
```

DIRECT AND INDIRECT REGISTER ADDRESSING

FIGURE 3.11

LITERAL FORMAT



INDIRECT REGISTER ADDRESSING : LITERAL MICROINSTRUCTION FORMAT

FIGURE 3.12

format. It should be noted that all Literal format microinstructions are single-cycle and also that since the register field is in the A position, register access is obtained through the Read/Write register file port.

Indirect addressing with a Register Format microinstruction is illustrated in Figure 3.13. The G register contents are the same for both the A and B field indirect addresses. The register addressing modes used in the Register Format may be any combination of direct and indirect modes. Additional specific details of Register Format microinstructions are supplied in Chapter 4, The Micromachine Instruction Set.

Data may be written into specified register locations only through the A port. There are four possible data sources which lead to the A port, listed as follows:

1) ALU Output
The output of the ALU is returned to the register file.

2) ALU Status Bit and Condition Code Flags
The result of ALU operations as monitored by the status bit and condition code flags may be stored in a specified register.

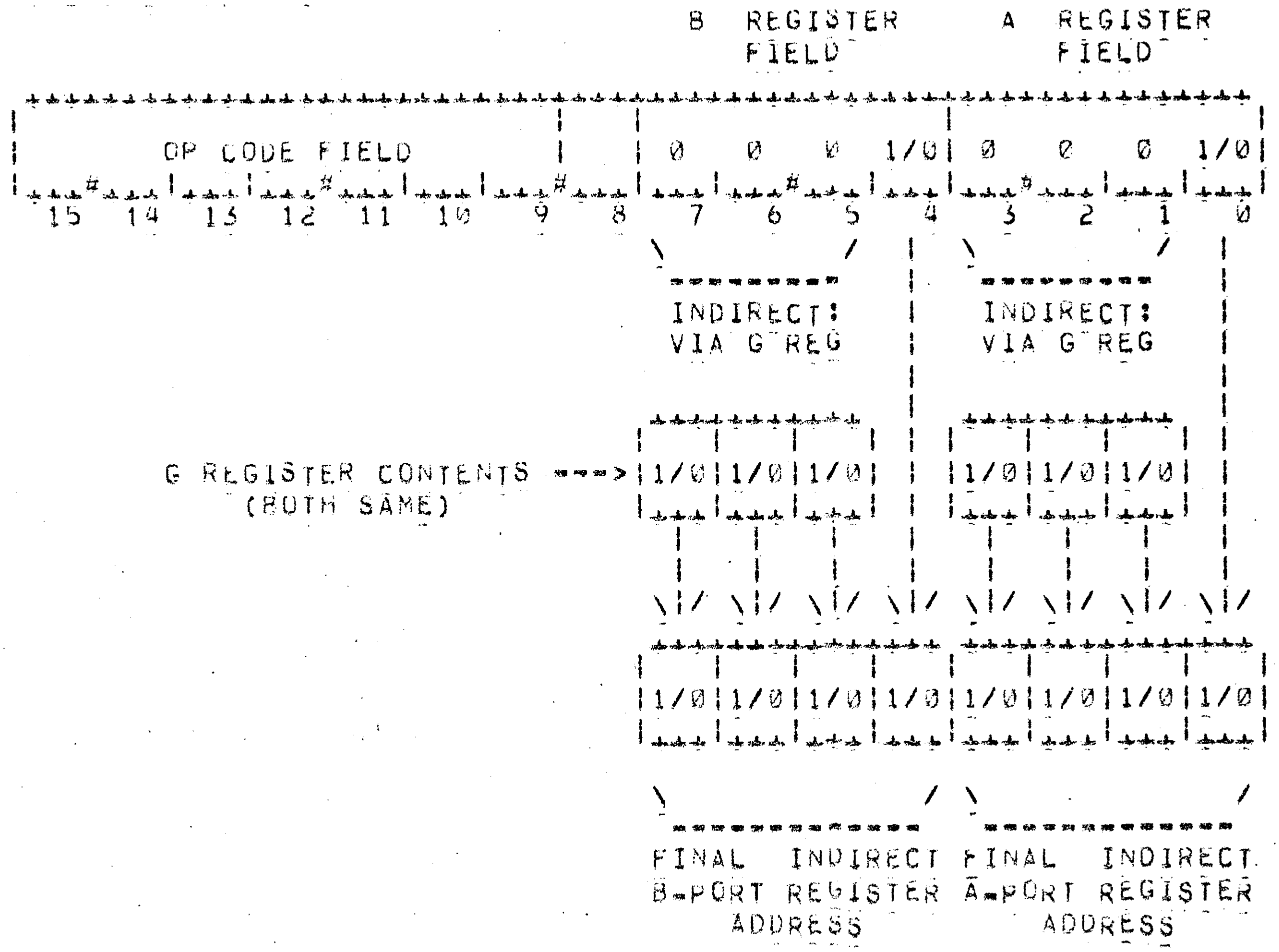
3) WDA <07:00>

4) WDAL <15:08>
Since the write access to the register file is only 8 bits wide, an operation which loads a complete word (low and high byte) from the LSI-11 system bus must require at least two micromachine cycles.

The Indirect or G register supplies the 3 highest bits of the final 4-bit indirect register address. The G register is loaded only via the Load G Low (LGL) or Input Word (IW) microinstructions which are described in the next chapter. Note that there is no mechanism which increments, decrements, or clears the G register contents.

3.3.1.3 Arithmetic Logic Unit - The microprocessor data chip ALU accepts an 8-bit operand each from the A and B ports of the register file. Both register ports operate in the Read mode to supply the operands and the ALU result is stored in a register through the A port which then operates in the Write mode. The ALU performs extensive logical and arithmetic operations. Arithmetic operaions executed at the micromachine level may use two's complement or decimal (BCD) number representation. Byte operations are executed in a single micromachine cycle while word operations require two cycles.

REGISTER FORMAT



INDIRECT REGISTER ADDRESSING : REGISTER MICROINSTRUCTION FORMAT

FIGURE 3.13

3.3.1.4 Status Bits and Condition Code Flags - The results of ALU operations are monitored by the status bit and condition code flags. These flags are organized in a register format and are updated after each ALU operation. Figure 3.14 illustrates the flag register organization. The condition code flags are updated selectively. A given Register Format microinstruction will leave the condition code flags unmodified if the opcode is an even number, whereas an opcode having an odd number will update the flags. This feature is useful in performing intermediate data manipulations which do not affect the PDP-11 condition code flag results. The ALU status bit flags are updated after each ALU operation.

The operating rules for the condition code flags are listed below:

CONDITION CODE FLAGS

Z : This flag is set if the most signficant bit of the result of a byte or a word operation is a zero. This flag is cleared otherwise.

N : This flag is set if the most significant bit of the result of a byte or a word operation is a 0. This flag is cleared otherwise.

C : This flag monitors bits which are carried, borrowed, or shifted according to the following-

Add and Increment

This flag is set if there is a carry from the most significant bit of a byte or a word operation. This flag is cleared otherwise.

Subtract and Increment

This flag is set if there is a borrow from the most significant bit of a byte or a word operation.

This flag is cleared otherwise.

Shift

This flag is set if the result of a right or left shift operation is to shift a 1 off the end of the byte or word. This flag is cleared otherwise.

Note that the C flag is affected only for operations in the areas listed above.

V : This flag is set if an arithmetic operation results in an overflow. This flag is cleared if no overflow occurs or if the operation performed is not an arithmetic operation.

```
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |       |       |       |       |       |       |       |       |
 |  N8   |  Z8   |  C4   |  C8   |   N   |   Z   |   V   |   C   |
 |_+_+_+_|_+_+_+_|_+_+_+_|_+_+_+_|_+_+_+_|_+_+_+_|_+_+_+_|_+_+_+_|
     7       6       5       4       3       2       1       0
 \                           /   \                           /
  ---------------------------     ---------------------------
             ALU                             ALU
       STATUS  BIT                 CONDITION  CODE
            FLAGS                          FLAGS
```

ARITHMETIC LOGIC UNIT STATUS BIT AND CONDITION CODE FLAG REGISTER

FIGURE 3.14

Since the ALU status bit flags are updated after each 8-bit ALU operation, they are used by the microprocessor to handle carries and borrows (for example, during a word operation). Consequently they are updated only for arithmetic, shift, test and compare microinstructions.

STATUS BIT FLAGS

ZB : This flag is set if the result of a byte or a word operation is 0. This flag is cleared otherwise.

NB : This flag is set if the most significant bit of the result of a byte or a word operation is a 1. This flag is cleared otherwise.

C4 : This flag is used only for operations involving decimal arithmetic. It is set if a carry from the third bit to the fourth bit is a 1. This flag is cleared otherwise.

C8 : This flag is set if the carry from the most signficant bit is a 1 or if the result of a shift operation is to shift off a 1. This flag is cleared otherwise.

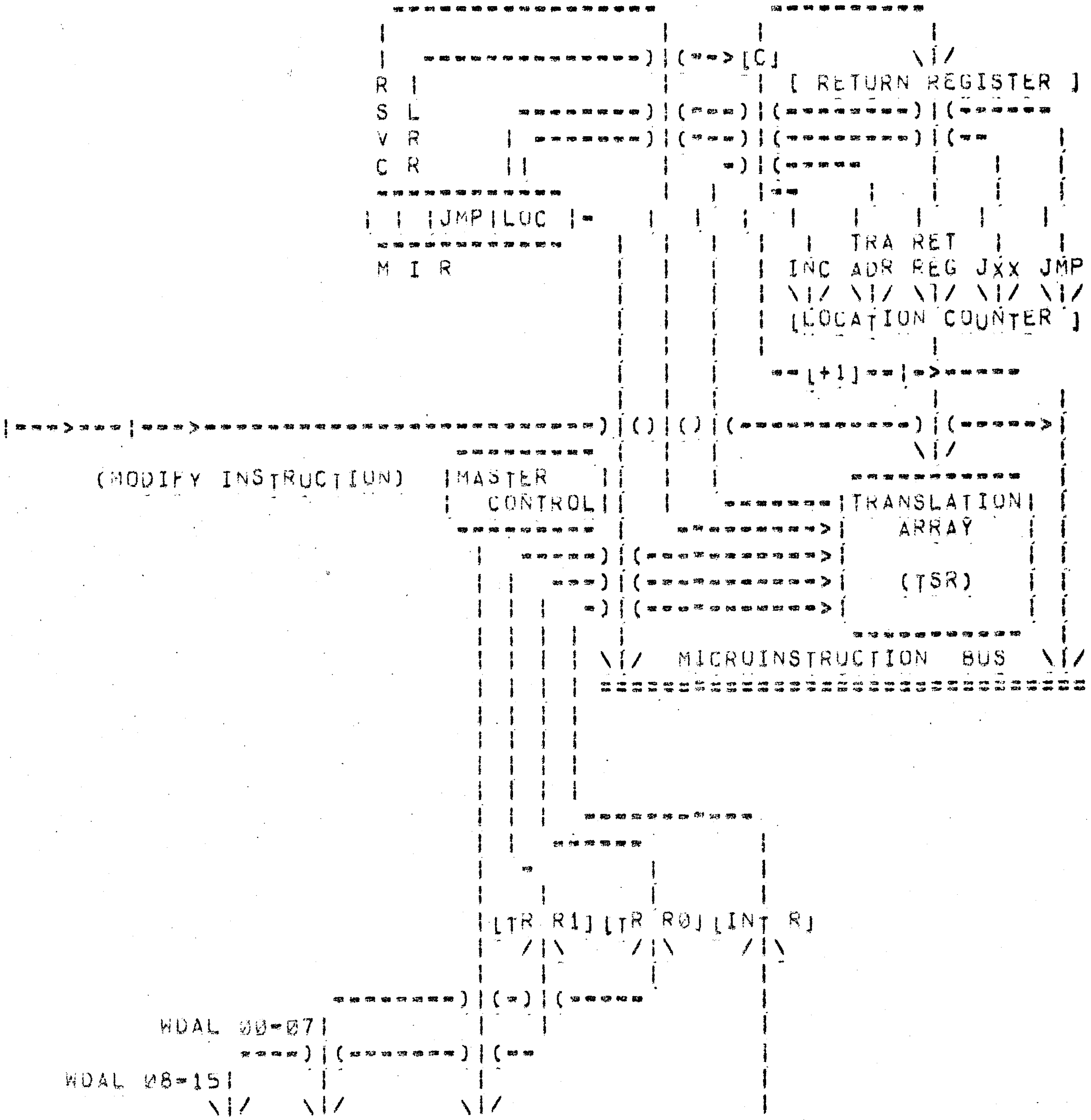
Each of the status bit and condition code flags shown in Figure 3.14 may be tested by a conditional jump instruction, except for the C4 status bit. In place of this bit, there is a test of the Indirect Condition Status (ICS) bit which is used to determine a specific microprogram-level jump conditions. The jump on ICS microinstructions are used to directly implement the machine-level Branch instructions. The 16 Conditonal Jump microinstructions and the Indirect Conditon Status rules are given in the next chapter.

The Status Bit/Condition Code flag register contents may be written into a register using the Copy Condition Flags (CCF) microinstruction. Conversely, the 8-bit contents of a specified register may be loaded into the flag register by the Load Condition Flags (LCF) microinstruction. In the latter case, the flags which are loaded are individually controllable.

3.3.2 Micrprocessor Control Chip

The microprocessor Control chip functions primarily in two areas; (1) it controls the flow of micromachine instructions which are fetched from control store (MICROM) and delivered to the Data chip for execution and (2) it administrates all LSI-11 system bus transactions by means of its connection to the Bus Transceivers and Interface Logic shown in Figure 3.15. It should be emphasized that Figure 3.15 is a functionally accurate Control Chip representation, but that the actual control chip circuitry is somewhat different. The difference arises because the data/address line inputs (WDAL <07:00> and WDAL <15:08>)

which lead to the translation registers (IR R0 and IR R1) do not appear on the control chip.



MICROPROCESSOR CONTROL CHIP DETAIL
FIGURE 3.15

The 16-line path between the WDAL lines on the data chip and the two translation registers on the control chip is established by time-multiplexing the MicroInstruction Bus. The primary function of the MIB is to deliver control store addresses to the MICROMS (or to the WCS module) and to retrieve the selected microinstruction for execution. When the MIB is not occupied with its primary function, it provides the 16-line path to the translation registers. This path is established during the the execution of the Input Word microinstruction which is part of the Fetch Machine Instruction operation discussed in the previous chapter. The fetched machine instruction is loaded into the translation register(s) for examination by the translation array.

3.3.2.1 Microinstruction Register - The microinstruction register shown in the control chip detail of Figure 3.15 contains 18 bits. The additional 2 bits over the Data chip microinstruction register) correspond to the LRR (Load Return Register) and RSV$_C$ (Read Next Instruction - Interrupts & Trap Service) function.

Load Return Register

As shown in Figure 3.15, the path leading into the return register originates at the location counter incrementer. When the LRR bit (bit<16>) is set it causes a value of LC + 1 to be loaded into the return register. This bit is part of the microinstruction at microprogram assembly time. The return register is loaded in preparation for the execution of a microprogram subroutine. Subroutine execution is terminated by the Return From Subroutine (RFS) microinstruction.

Read Next Instruction-Interrupt & Trap Service

Bit <17> of the microinstruction register is a direct input to the translation array. The RSV$_C$ bit is assembled into the microinstruction sequence one location before the translation is to be invoked. This translation causes machine-micromachine control to enter the top of the interrupt/trap decision chain shown in Figure 2.16-2, which will cause all traps and interrupts to be serviced according to their priorities. If the control flow does not encounter any traps or interrupts, control proceeds to the Fetch Machine Instruction event, which causes the next machine instruction to be read into the translation registers for examination by the translation array.

3.3.2.2 Microinstruction Address Generation - Each valid microinstruction address produced by the control chip is stored in the Location Counter. The Location Counter contents are gated onto the MicroInstruction Bus at the appropriate phase of the micromachine cycle as the first part of the microinstruction fetch (microfetch). The Location Counter may be loaded from any one of five possible sources.

which are listed and described below:

Incrementer : A value of 1 is added to the Location Counter output and written into the location counter. The next microinstruction is therefore fetched from the next sequential control store location.

Translation Array : The translation array, in response to its inputs, determines the location in control store from which the next microinstruction is to be fetched. The address of this location (shown as TRA ADR) is loaded into the Location Counter.

Return Register : Upon execution of the Return From Subroutine microinstruction, the contents of the Return Register are loaded into the Location Counter.

Conditional Jump Field : When the conditional jump criteria is met, the 8-bit contents of the conditional jump field are loaded into the lowest 8 bits of the Location Counter. The three highest bits are unchanged from the updated values.

Unconditional Jump Field : Upon execution of the JUMP microinstruction instruction, the 11-bit contents of the unconditional jump field are loaded into the Location Counter.

A microlocation address may be determined in one additional way, by the execution of the Modify Instruction (MI) microinstruction. The Modify Instruction path which is labeled in Figure 3.15 enables the contents of a specified register pair to be OR'ed with the unconditional jump microinstruction (or any other microinstruction) on the Microinstruction Bus. This microinstruction is discussed in the next chapter.

Translation Array

The translation array is a large combinatinal logic network which performs rapid examination of the fetched machine instruction in the Translation Register to determine where microprogram execution is to begin. The examination is completed in one machine cycle and the result is a translation address which is loaded into the location counter. The translation array can examine only 8 bits of a 16-bit machine instruction at one time. The translation state register, which is shown as (TSR) in Figure 3.15, determines which of the two translation registers (TR R0 or TR R1) is input to the translation array. When a new machine instruction is fetched, the TSR is reset and causes the upper byte of the instruction, which is contained in TR R1 to be examined first. This allows the machine instruction type to be determined before the the address modes are examined. The TSR contains 3 bits, one of which controls the translation register input, while the other two are used for purposes internal to the translation array.

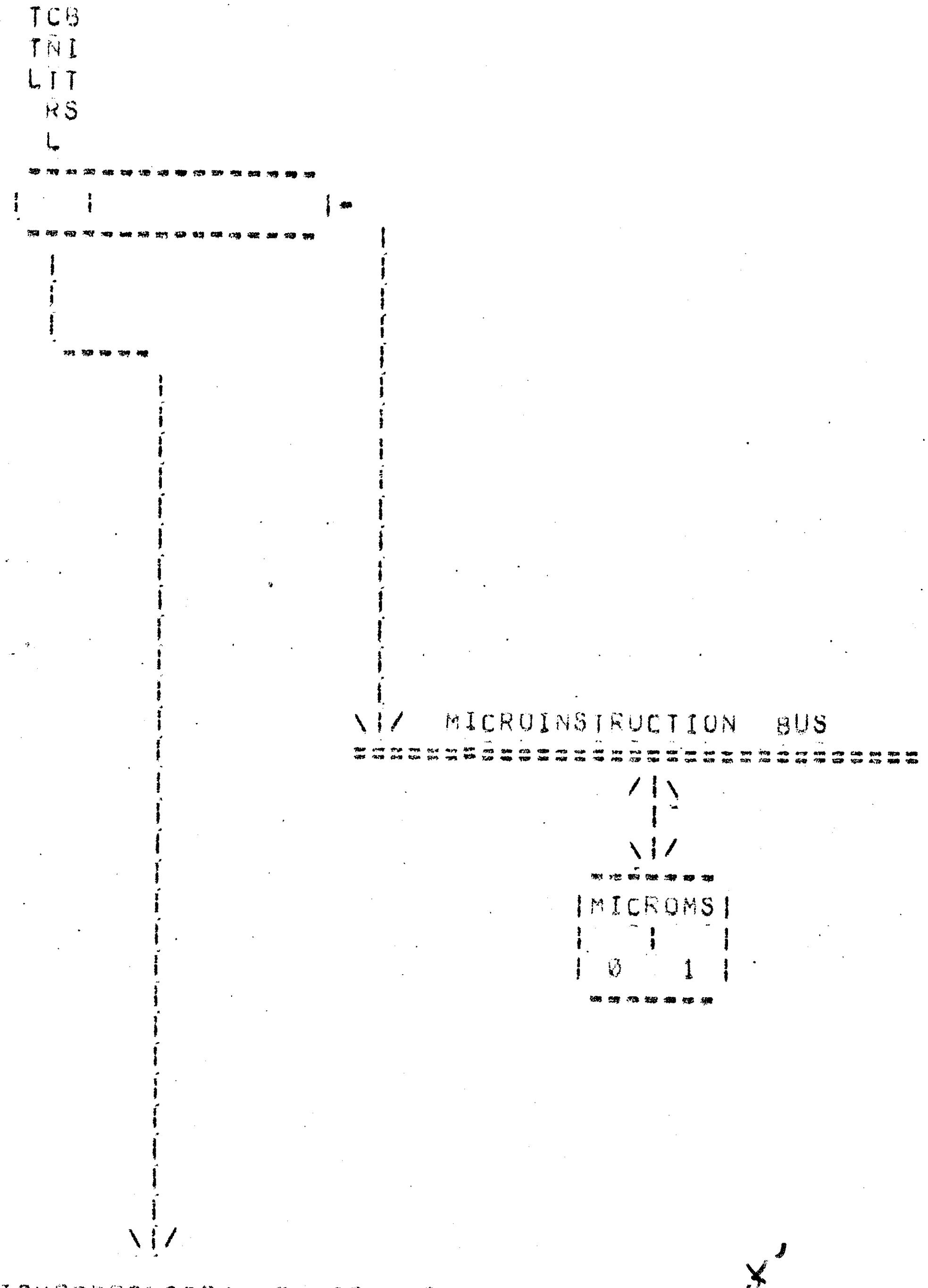
In addition to providing the translation address, the translation array also controls the loading of the microprogram subroutine address into the location counter. This control path is not shown in Figure 3.15.

The interrupt register, INT R, is composed on three internal interrupts and 4 external interrupts which were illustrated earlier in Figure 2.17. The external interrupts are connected to the LSI-11 system bus interface logic, The 3 internal interrupts may be set or cleared by the Set Interrupt (SI) or the Reset Interrupt (RI) microinstruction, respectively. When the translation array initiates service for a pending external interrupt, an early step in the service microprogram is to reset the interface logic flip-flop which had stored the external interrupt request. The flip-flop reset pulse is produced by the TTL Control Bits as decoded by the Special Function Logic (Figure 3.5). Figure 3.16 illustrates the abstracted Microprocessor TTL Control Bit Path as extracted from Figure 3.1. An inspection of Figure 3.2 will reveal that the TTL control Bits do not actually appear in the MicroInstruction Register of either the data chip or the control chip. However, Figure 3.15 emphasizes the association of the TTL CONTROL BITS with the microinstructions. These bits must be assembled at the proper postions in the microinstruction flow to assure proper microprogram execution. The exact details of timing and sequencing are discussed in the following sections.

3.3.2.3 Control Signals - In addition to the 4 external interrupts, eight other signals pass between the control chip and the LSI-11 processor interface logic. The operation of these signals are explained below and additional specific details are available in the references cited in Figure 3.4.

RESET

This line is an input to the control chip and when active, causes microprocessor operation to be suspended. When the RESET input subsequently goes passive, the first microinstruction to be executed will be fetched from control store location 0001. The RESET line is asserted whenever the power supply signal BDCOK H goes passive or when a bus timeout error occurs. The microprogram executed after the RESET line goes passive is flow-charted in Figure 2.16-2.



MICROPROCESSOR TTL CONTROL BIT PATH

FIGURE 3.16

this is only
functionally correct.

COMPUTE

The COMPUTE line is sampled during PH 3 of the micromachine cycle. This line is maintained in an active high state, and is used during manufacturing for test purposes.

The remaining signals are concerned with the control chip's administration of the three types of LSI-11 system bus I/O transfers, Programmed I/O, Interrupt-Driven I/O, and DMA I/O. These signals are interfaced to the LSI-11 bus system by the Bus I/O Control Signal Logic described in The Microcomputer Handbook.

SYNC

The transition of the SYNC signal to its active state indicates that the address data on WDAL <15:00> is valid. The sync signal remains asserted until the I/O transfer is completed. Additional interface circuitry assures that the system bus signal BSYNC L remains active until after the bus slave device terminates its reply signal.

REPLY

The REPLY originates in the addressed memory or I/O device and informs the Control chip that the I/O operation should be continued. Specifically, this signal must be asserted during PH 3 of the micromachine cycle while an Input or an Output microinstruction is being executed. The state of the reply signal is also interrogated before a READ or write operation is initiated. This is to assure that a previously-addressed device has completely released the system bus.

DIN

The DIN (Data-In) signal is asserted by the control chip after the address information is removed from WDAL <15:00>, or one micromachine cycle after the SYNC signal is asserted. The DIN signal returns to the passive state at the completion of the Input Byte (IB) or Input Word (IW) microinstruction, or when SYNC is made passive. This signal causes the addressed device to place its data on BDAL<15:00> for input to the processor.

DOUT

The DOUT (Data-Out) signal is asserted by the Control chip when output data is placed on WDAL <15:00> by the Data chip and remains asserted until the Output microinstruction is completed.

WB

The WB (Write/Byte) signal is asserted when the address information is placed on WDAL <15:00> to indicate that a Write operation follows. If it remains active during the Data-Out portion, a Byte operation (DAIOB) is signified.

BUSY

The BUSY signal is samole during PH 3 of the micromachine cycle during the first cycle of a Read or Write microinstruction. If the busy signal is asserted, the microorocessor enters a wait state and suspends operation until BUSY goes passive. This mechanism is used by the direct memory access interface logic to gain control of the system bus.

Interrupt Acknowledge

The WIACK signal is asserted by the Control chip along with the SYNC line when a Read Acknowledge or Write acknowledge microinstruction is executed. The WIACK signal indicates that the microprocessor is responding to an interrupt. The LSI-11 interface logic delays the appearance of WIACK as BIACK H until BDIN L is asserted. This is to allow the BDIN L signal to stabilize priorities between the two possible requests in the interrupting device (see The Microcomputer Handbook).

3.4 MICROMACHINE OPERATION

All micromachine operations are organized with respect to the four-phases of the micromachine clock. Each phase has a nominal period of 95 nanoseconds giving a cycle period of 380 nanoseconds. The clock phases are distributed to the Control, Data, and MICROM chips of the microprocessor as MOS-level non-overlapping pulses (RPH <4:1> True and complement clock phase pulses are distributed to the Bus Transcievers and Interface Logic circuitry for synchronization with the micromachine.

Micromachine operating speed is maximized by the use of pipelining techniques to determine microlocation addresses and to access microinstructions for execution. The pipeline technique are implemented in the context of a time-multiplexed Microinstruction Bus, which reduces inter-chip wiring.

3.4.1 Microinstruction Bus Data Transfer

The data transfer method employed on the microinstruction bus is the precharge-conditional discharge technique which is compatible with LSI MOS memory and microprocessor devices. Data is transmitted on the microinstruction bus in logical complement form in which the lower voltage represents the HI state or a Logical 1. Each microinstruction bus line is unconditionally precharged HI at one phase and selectively discharged at a later phase. The receiving device senses the discharged state during the appropriate phase and interprets the lower voltage as a logical 1. All precharging is performed by the MICROMs. Depending upon the microinstruction bus line and signal usage, the receiving device may be the Control chip, the Data chip, or a MICROM. In the case of a microinstruction fetch, the Control and Data chips receive the microinstruction simultaneously, each chip storing the needed portions in its microinstruction register.

3.4.1.1 Control Store (MICROM) Microinstruction Bus Cycle - The MICROM operations as a function of micromachine clock phases are illustrated in Figure 3.17. The location counter contents are gated onto the microinstruction bus by the Control Chip during PH 2 and remain valid during PH 3. All microinstruction bus lines are then unconditionally precharged at PH 4 (by the MICROMs) with the exception of MIB <15>, which is crecharged at PH 3. MIB 16 was also precharged at PH 2 in addition to being precharged at PH 4.

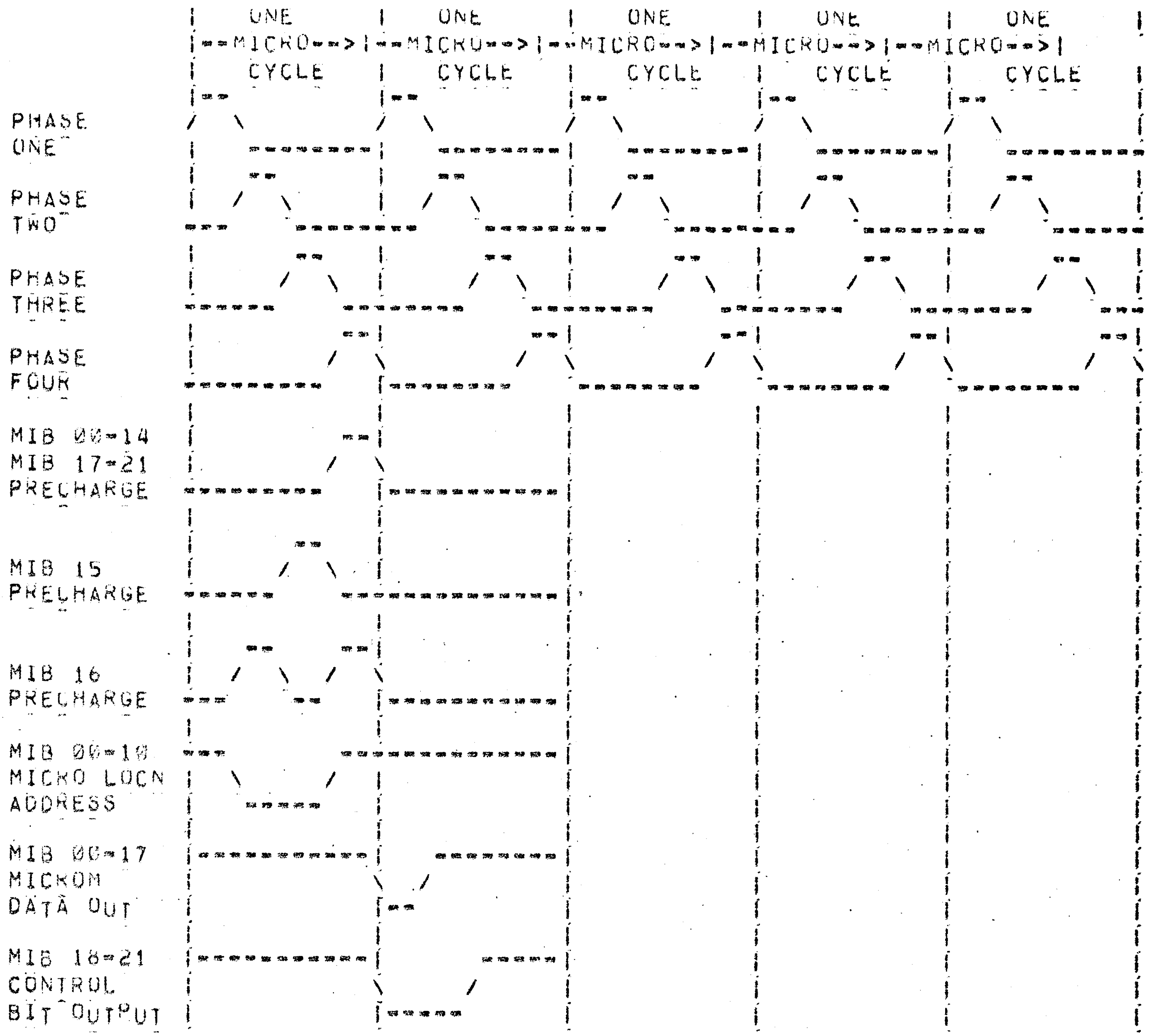
During the execution of a microinstruction which requires only one micromachine cycle, the contents of the microlocation addressed during PH 2 and PH 3 are returned to the data and control chips during PH 1 of the following micromachine cycle. However, if the control chip discharges MIB 16 at PH 3, the selected MICROM will not conditionally discharge MIB <15:00> and MIB <21:18> during the following PH 1. This circumstance occurs during the second cycle of a two-cycle microinstruction, or in response to a WAIT state. Figure 3.18 illustrates the disabling of the MICROM during the first cycle, resulting in a delay of one micromachine cycle.

As shown is the figures, the TTL Control Bits are valid beginning with PH 1 throughout PH 3. The LSI-11 circuitry employs PH 3 as a strobe pulse which produces the ROM Codes during PH 3.

3.4.1.2 Control Chip Microinstruction Bus Cycle - The Control chip determines the micromachine instruction flow by selectively loading the Location Counter. The basic micromachine cycle of control chip operations is shown in Figure 3.19. The Location Counter is loaded with a new value during PH 1. This value can originate from 1 of 5 sources as discussed in Section 3.3.2. The contents of the Location Counter are output onto the microinstructin bus at the beginning of PH 2 and remain valid through PH 3. Also during these two phases, the translation array is processing its inputs to see if a translation is required. During PH 4 the translation address becomes ready for loading into the location counter and the translation state register may be loaded.

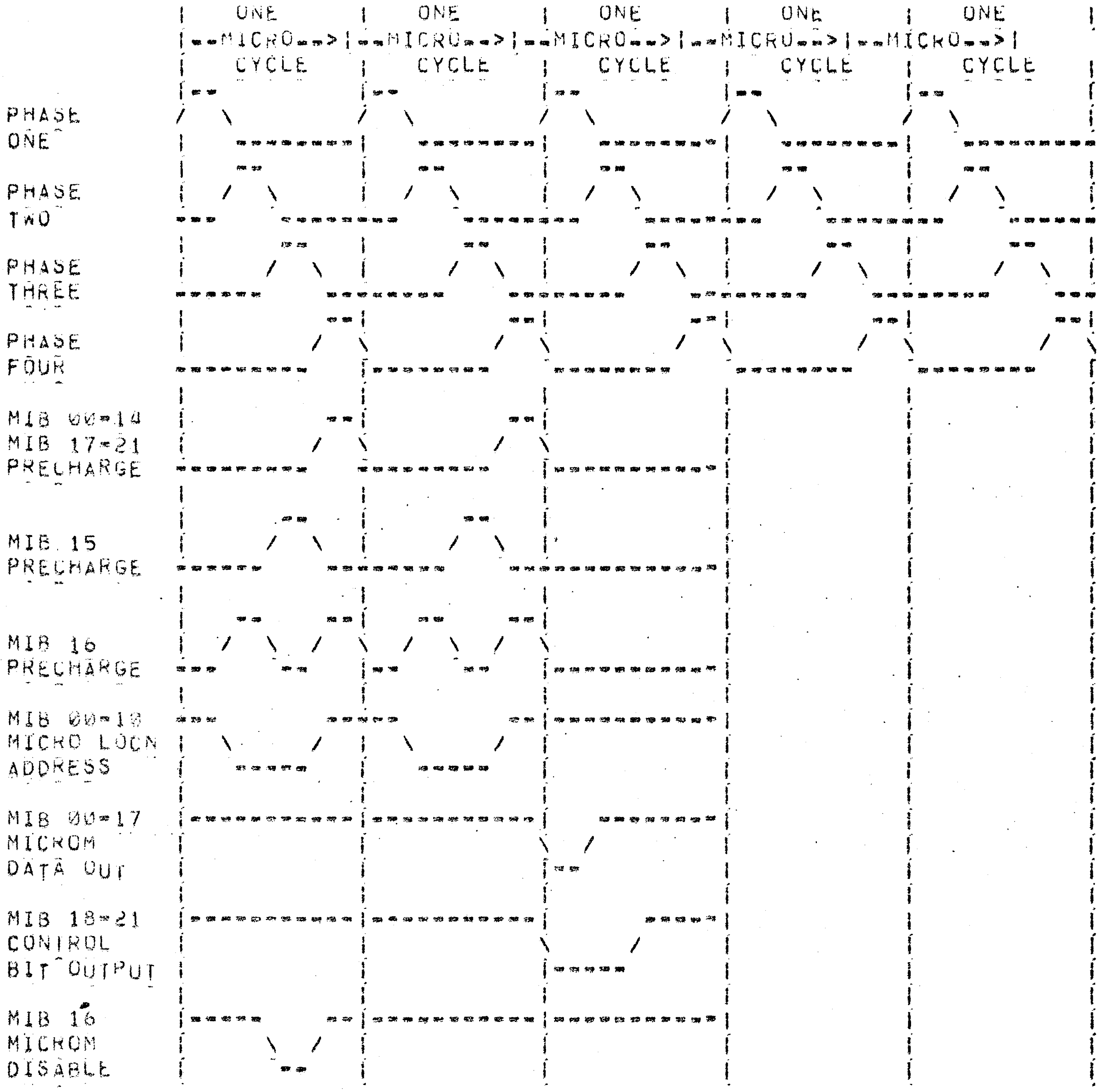
An example of Control chip operation effected by the WAIT state is shown in Figure 3.20. Events proceed normally until PH 3 when the

Control chip determines a WAIT state, This determination is the result of sampling the BUSY and REPLY line during PH 3. The Control chip's response to the WAIT state is to discharge MIB<16> during PH 3 to disable the MICROMS and to assert the WAIT line during PH 4, which prevents the Data chip from loading a new microinstruction.



MICROM ACCESS MICROINSTRUCTION BUS CYCLE

FIGURE 3.17



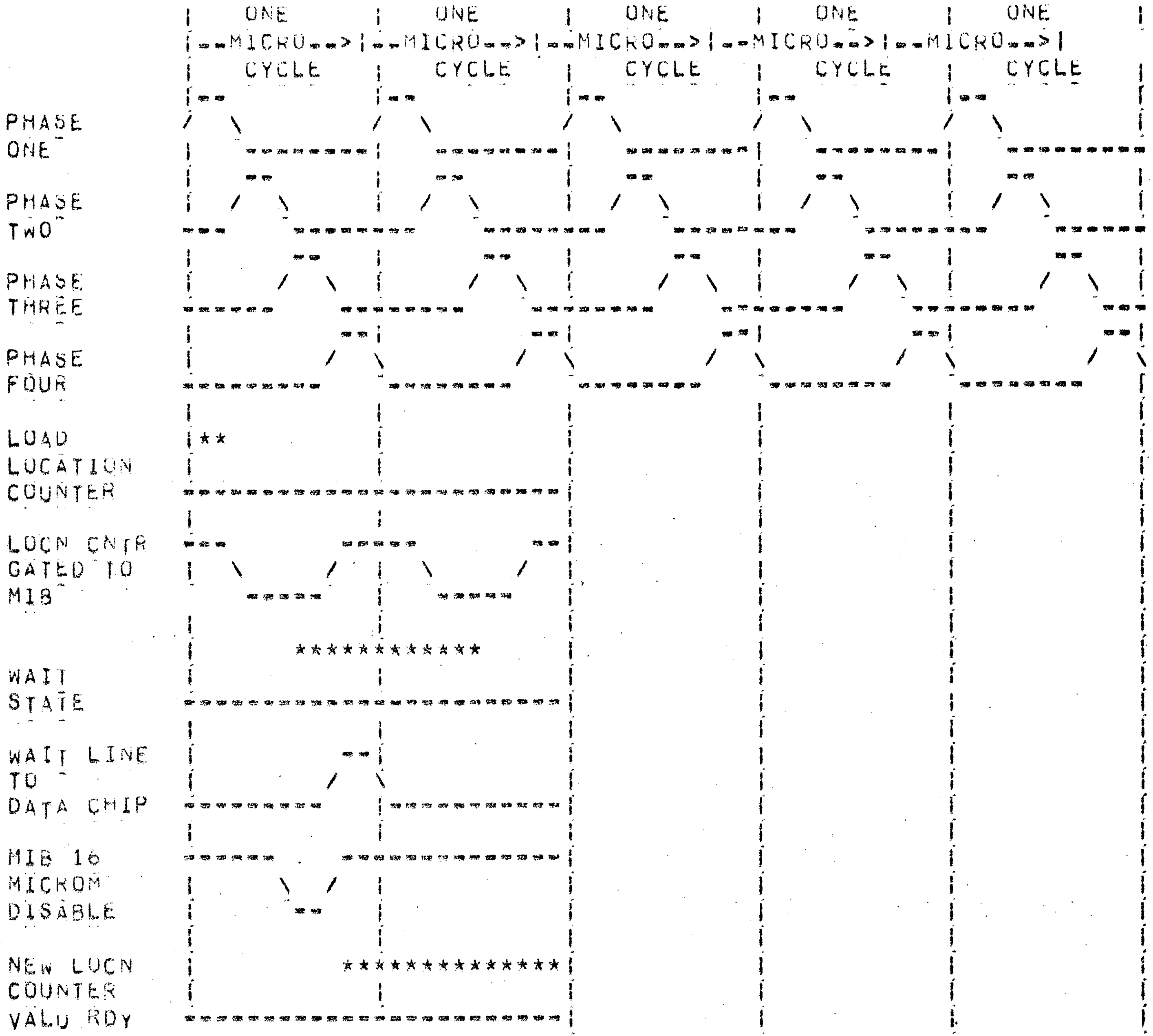
MICROM ACCESS MICROINSTRUCTION BUS CYCLE (DISABLED)

FIGURE 3.18

```
          |   ONE    |   ONE    |   ONE    |   ONE    |   ONE    |
          |--MICRO-->|--MICRO-->|--MICRO-->|--MICRO-->|--MICRO-->|
          |   CYCLE  |   CYCLE  |   CYCLE  |   CYCLE  |   CYCLE  |
          |--        |--        |--        |--        |--        |
PHASE     /  \       /  \       /  \       /  \       /  \       |
ONE       |   -------|   -------|   -------|   -------|   --------
          |   --     |   --     |   --     |   --     |   --     |
PHASE     |  /  \    |  /  \    |  /  \    |  /  \    |  /  \    |
TWO       ---    -------    -------    -------    -------    -----
          |      --  |      --  |      --  |      --  |      --  |
PHASE     |     /  \ |     /  \ |     /  \ |     /  \ |     /  \ |
THREE     ------    -------    -------    -------    -------    --
          |        --|        --|        --|        --|        --|
PHASE     |       /  \       /  \       /  \       /  \       /  \
FOUR      --------    -------    -------    -------    -------   |
          |          |          |          |          |          |
LOAD      |**        |          |          |          |          |
LOCATION  |          |          |          |          |          |
COUNTER   -----------|          |          |          |          |
          |          |          |          |          |          |
LOCN CNTR ---      --|          |          |          |          |
GATED TO  |  \    /  |          |          |          |          |
MIB          ----    |          |          |          |          |
          |          |          |          |          |          |
NEW LOCN  |        **|          |          |          |          |
COUNTER   |          |          |          |          |          |
VALU RDY  -----------|          |          |          |          |
```

CONTROL CHIP SINGLE-CYCLE MICROINSTRUCTION BUS OPERATION

FIGURE 3.19



CONTROL CHIP WAIT RESPONSE

FIGURE 3.20

Because of the WAIT state, the Location Counter value remains unchanged during the second micromachine cycle and the translation array processing produces the same output which may be a conditional loading of the translation state register or a translation address. Since the WAIT state was not re-established during the second micromachine cycle, a new location counter value will be loaded during the following PH 1 and a new microinstruction will be sent from the selected MICROM.

The execution of a two-cycle microinstruction produces a Control chip sequence similar to the above. However, since both the Control chip and the Data chip independently recognize two-cycle machine instructions, the WAIT signal is unneccessary and therefore unasserted at PH 4.

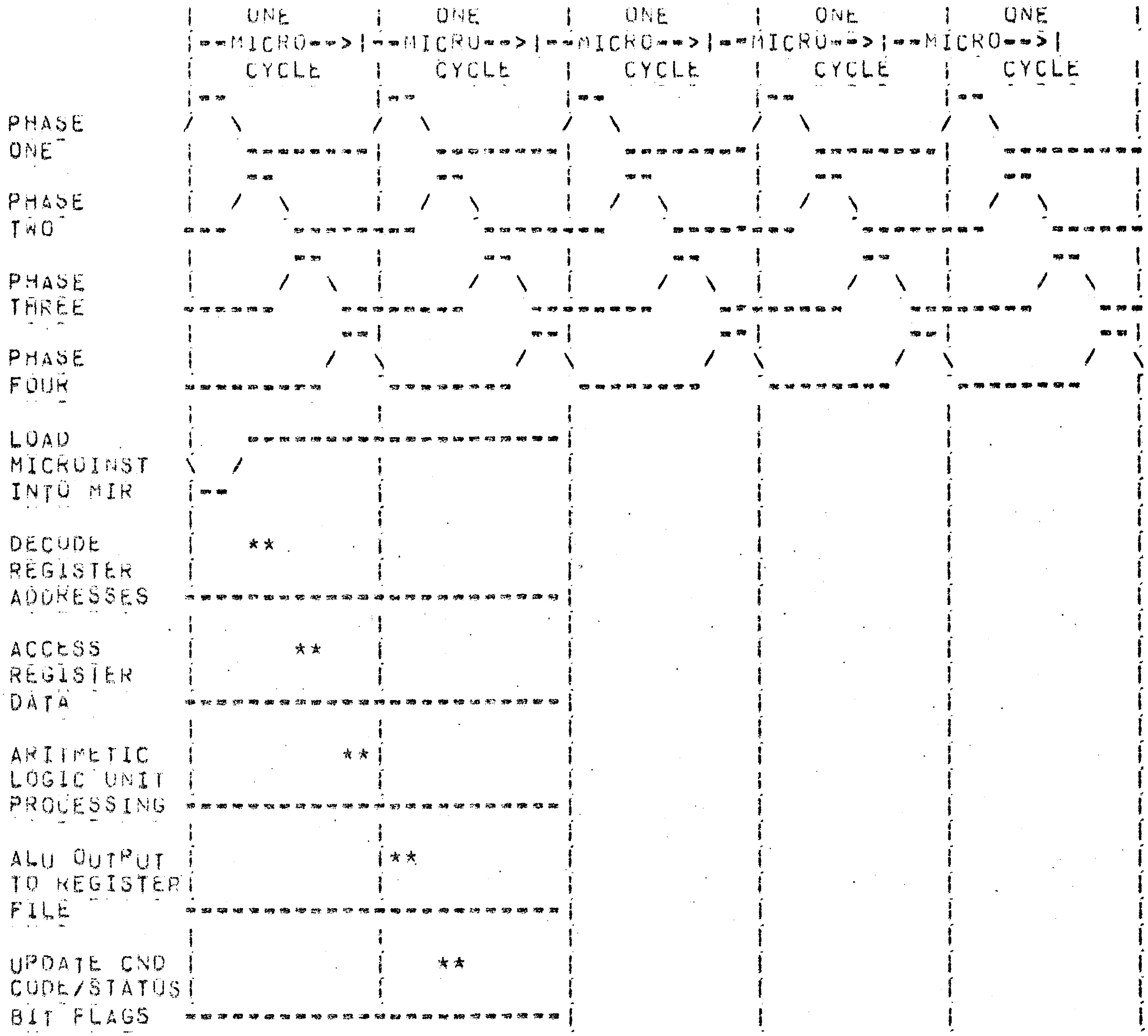
3.4.1.3 Data Chip Microinstruction Bus Cycle - The microprocessor Data chip is completely controlled by the microinstruction in its microinstruction register and by the WAIT line which originates in the Control chip. The basic micromachine single-cycle sequence is shown in Figure 3.21. During PH 1 the Location Counter contents placed on MIB<15:00> by the selected MICROM are loaded into the Data chip's microinstruction register. The register address fields are decoded during PH 2. The address registers are accessed during PH 3 and the ALU processes the operands input to it during PH 4. The ALU result is not ready until PH 1 of the subsequent micromachine cycle, when it is written into the register file via the A port. The status bit and condition code flags which monitor the ALU result are updated during PH 2.

When a two-cycle data manilpulation microinstruction is executed, the Data chip retains its microinstruction register contents during the second cycle as shown in Figure 3.22. The low bit of each register field is complemented during PH 1 of the second cycle which allows the second cycle of a word microinstruction to be processed in the ALU.

All Jump microinstructions require two micromachine cycles for execution. The Unconditional Jump instruction requires two cycles because the Location Counter contents cannot be loaded with a new value until PH 1 of the second cycle.

A Conditional Jump microinstruction examines the flags which have settled during PH 2 and sends the result to the control chip over MIB 15 during PH 4. If the result is postive, the 8-bit page address will be loaded into the Location Counter during the following PH 1.

Normal operation of the Data chip is suspended if the Control chip asserts the WAIT signal during PH 4 of any micromachine cycle. The data chip's response is to maintain its previous microinstruction contents, which causes the operation just executed to be repeated.



DATA CHIP SINGLE-CYCLE MICROINSTRUCTION EXECUTION

FIGURE 3.21

```
           |  ONE     |  ONE     |  ONE     |  ONE     |  ONE     |
           |--MICRO-->|--MICRO-->|--MICRO-->|--MICRO-->|--MICRO-->|
           |  CYCLE   |  CYCLE   |  CYCLE   |  CYCLE   |  CYCLE   |
           |--        |--        |--        |--        |--        |
PHASE      /  \       /  \       /  \       /  \       /  \       |
ONE        |   -------|   -------|   -------|   -------|   --------
           |  --      |  --      |  --      |  --      |  --      |
PHASE      | /  \     | /  \     | /  \     | /  \     | /  \     |
TWO        --    -------    -------    -------    -------    ------
           |     --   |     --   |     --   |     --   |     --   |
PHASE      |    /  \  |    /  \  |    /  \  |    /  \  |    /  \  |
THREE      -----    -------    -------    -------    -------    ---
           |        --|        --|        --|        --|        --|
PHASE      |       /  \       /  \       /  \       /  \       /  \
FOUR       --------   --------   --------   --------   --------   |
           |          |          |          |          |          |
LOAD       |   -----------------------------|          |          |
MICROINST  \  /       |          |          |          |          |
INTO MIR   |--        |          |          |          |          |
           |       1ST|       2ND|          |          |          |
DECODE     |  **  BYTE|  **  BYTE|          |          |          |
REGISTER   |          |          |          |          |          |
ADDRESSES  ---------------------------------|          |          |
           |1ST       |2ND       |          |          |          |
ACCESS     |BYTE **   |BYTE **   |          |          |          |
REGISTER   |          |          |          |          |          |
DATA       ---------------------------------|          |          |
           |1ST       |2ND       |          |          |          |
ARITHMETIC |BYTE    **|BYTE    **|          |          |          |
LOGIC UNIT |          |          |          |          |          |
PROCESSING ---------------------------------|          |          |
           |          |       1ST|       2ND|          |          |
ALU OUTPUT |          |**    BYTE|**    BYTE|          |          |
TO REGISTER|          |          |          |          |          |
FILE       ---------------------------------|          |          |
           |          |       1ST|          |          |          |
UPDATE CND |          |  **  BYTE|  **  WORD|          |          |
CODE/STATUS|          |   RESULTS|   RESULTS|          |          |
BIT FLAGS  ---------------------------------|          |          |
           |          |          |          |          |          |
MIB 16     ----------------    -------------|          |          |
MICROM     |          |    \  /  |          |          |          |
DISABLE    |          |     --   |          |          |          |
```

DATA CHIP DOUBLE-CYCLE MICROINSTRUCTION EXECUTION

FIGURE 3.22

3.4.1.4 Complete Micromachine Cycle - The inter-relationship between the cycles of the Control, Data and MICROM chips is illustrated in Figure 3.23 for the case of a single-cycle microinstruction. The operation sequences listed for each chip are repetitive, but only the events related to the execution of a single microinstruction are shown in the figure. The complete sequence begins with a new Location Counter value being loaded during PH 1 of the first micromachine cycle and ends when the status bit and condition code flags are updated during PH 2 of the third micromachine cycle. An accurate representation of micromachine operations can be constructed by combining the indiviidual chip operaion sequences dicussed previously. The sequences chosen depend upon the microinstruction flow being represented.

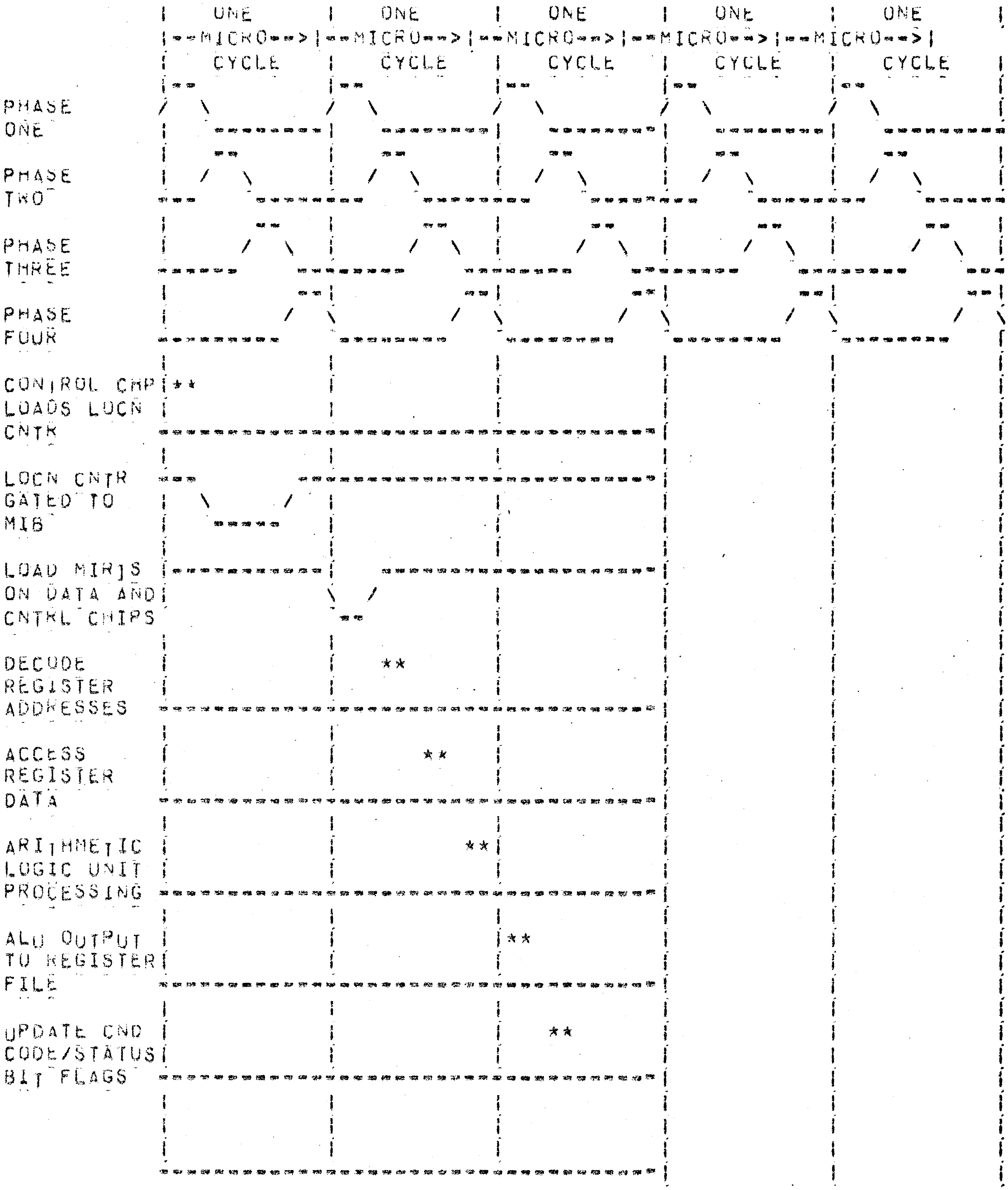
The Execution of the Input Word microinstruction causes the function of the microinstruction bus to be changed during the second cycle of this two-cycle microinstruction. Since P_H 1 of the second cycle is not used to update the microinstrucion registers with new contents, the Data chip uses MIB<15:00> to send the fetched machine instruction to the translation register for examination by the transalaion array. This is necessary because the ~~data~~ control chip has no direct connection to the WDA$_L$ lines. It should be remembered that Figure 3.15, The Microprocessor Control Chip Detail, is accurate only functionally in that it indicates a path from the WDA$_L$ lines to the translation register.

3.5 MICROPROGRAMMING THE BASIC LSI-11 MACHINE CYCLE

The essential characteristics of the microprogram which emulates the LSI-11 microcomputer architecture will be discussed for two simple cases. The first case is the basic two-event machine cycle and the second case is the basic machine cycle including an external interrupt and bus error traps.

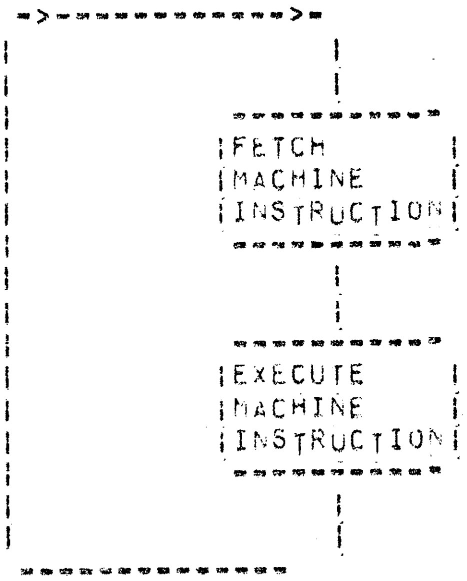
3.5.1 Fetch - Execute Machine Cycle Microprogramming

The basic LSI-11 machine cycle, repeated here as Figure 3.24, consists of a DATI operation followed by a microppogrammed rountine which interprets and executes the fetched machine instruction. The DATI operation is itself microprogrammed as a Read/Input sequence which is discussed in Chapter 5. The specific Read microinstruction employed is Read And Increment Word By 2 (RIW2). This microinstruction places the contents of the register pair designated within it, R PC L and R PC H, on BDA$_L$ L <15:00> and signals a Data-In system bus cycle. It subsequently increments the word contained in PCH and PCL by 2, causing the program counter to point to the next address from which machine instruction is to be fetched. The specific Input microinstruction used, which comprises the second half of the Read/Input sequence is Input Word. This microinstruction places the fetched machine instruction into the instruction register RIRL and RIRH and also loads it into the translation registers TR R0 and TR R1.


ONE MICRO CYCLE
ONE MICRO CYCLE
ONE MICRO CYCLE
ONE MICRO CYCLE
ONE MICRO CYCLE
PHASE ONE
PHASE TWO
PHASE THREE
PHASE FOUR
CONTROL CHP LOADS LOCN CNTR
LOCN CNTR GATED TO MIB
LOAD MIRS ON DATA AND CNTRL CHIPS
DECODE REGISTER ADDRESSES
ACCESS REGISTER DATA
ARITHMETIC LOGIC UNIT PROCESSING
ALU OUTPUT TO REGISTER FILE
UPDATE CND CODE/STATUS BIT FLAGS
**

COMPLETE MICROMACHINE CYCLE

FIGURE 3.23



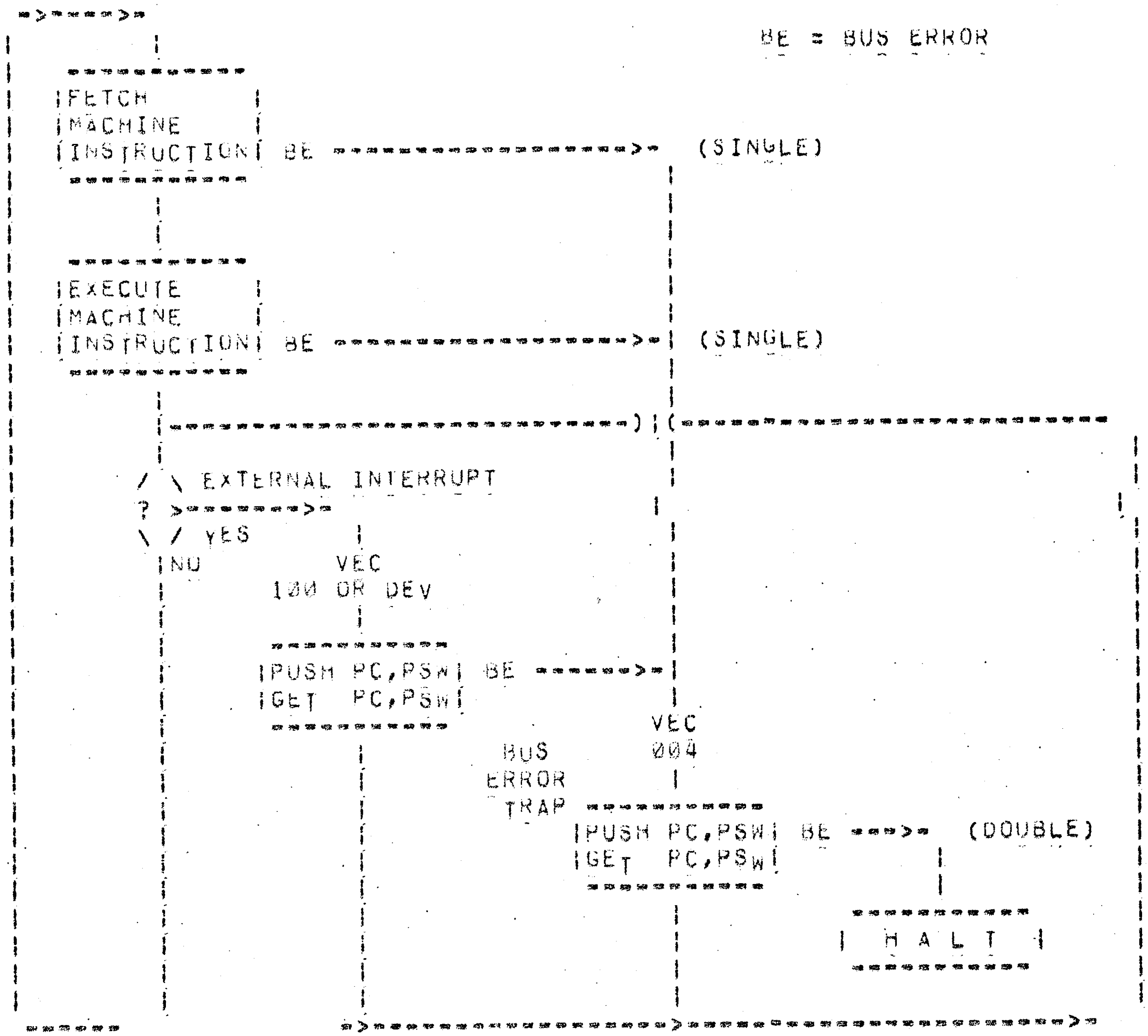
BASIC LSI-11 MACHINE CYCLE

FIGURE 3.24

The Execute Machine Instruction portion of the basic machine cycle begins with the examination of the translation register contents by the translation array. The translation array output, which is available after one micromachine cycle, produces the address of a control store or MICROM location which contains the first instruction of the microprogrammed routine which will execute the machine instruction. The translation array may be used one, two or three times during the Execute Machine Instruction operation, depending upon the type of instruction being executed. In preparing the ALU operands, zero, one, or several DATI operations need to be performed, depending upon the addressing modes used in the machine instruction source and destination fields.

3.5.1.1 Fetch-Execute-Trap & Interrupt Machine Cycle Microprogramming - The basic machine cycle of Figure 3.24 is expanded in Figure 3.25 to include an external interrupt and the single and double bus error traps. As explained in Chapter 2, a bus error occurs when an addressed device does not respond within 10 microseconds. This leaves the Read/Input sequence of the DATI operation uncompleted and the Bus Interface Logic asserts the control chip's reset line. As a result, the location counter is loaded with 0001 and a microprogrammed bus error recovery routine is executed. Two DATO cycles are performed to push the PC and PSW contents onto the stack, followed by two DATI cycles which load new values into the program counter and processor status word registers.

The external interrupt status is interrogated by executing a microinstruction which has bit <17> set. This bit invokes the Read Next Instruction-Trap & Interrupt Service (RSVC) function of the translation array. If an interrupt is asserted, the translation array will load the Location Counter with the first address of the microprogrammed routine which services the interrupt. The simple example contains only one interrupt, but the actual control chip translation array interrogates 3 internal interrupts and 4 external interrupts and produces control store locations according to the priorities illustrated in Figure 2,16-2.



EXTERNAL INTERRUPT AND BUS ERROR TRAP

FIGURE 3.25

CHAPTER 4

THE LSI-11 MICROINSTRUCTION SET

4.1 GENERAL

An LSI-11 microinstruction is composed of 22 bits: a 16-bit Vertical Microinstruction field; a 1-bit Load Return Register control field (LRR); a 1-bit Read Next Instruction, Trap and Interrupt Service control field (RSVC); and a 4-bit logic control (TTL Control Bits) field. The complete microinstruction is illustrated in Figure 4.1 and the bit fields are defined below:

TTL Control Bits	MI<21:18>
Read Next Instruction, Trap + Interrupt Service	MI<17>
Load Return Register	MI<16>
Vertical Microinstruction	MI<15:0>

The contents of these four bit fields are independent of each other. The 16-bit Vertical Microinstruction field can contain any of the 149 possible microinstructions. These microinstructions are the subject of this chapter. The functions of the LRR, RSVC and TTL control bit fields were described in Chapter 3 and the combined operation of all 4 fields will be discussed in Chapter 8, Microprogramming Techniques. This chapter contains an overview of the microinstructions, Appendix A contains detailed descriptions. Appendix B contains a Microinsruction Summary.

4.2 VERTICAL MICROINSTRUCTIONS

Vertial microinstructions have a strong resemblance to LSI-11 machine instructions due to the following similarities:

1. Both instruction types are fetched and executed in sequential fashion, unless a program control mechanism modifies the in-

struction flow.

2. Both instruction types consist of relatively few internal formats (bit field arrangements, etc.).

3. Both instruction types normally execute a "complete" operation, i.e., operands are accessed in the microprocessor register file, delivered to the ALU, processed, and returned to the register file before the next microinstruction is executed.

4.2.1 Jump Microinstruction Format

The Jump microinstruction Format is composed of three fields and is illustrated in Figure 4.2. The opcode field (MI<15:12>) always contains all zeroes. The R field (MI<11>) indicates a Return From Subroutine microinstruction. The unconditional jump address field (MI<10:0>) contains 11 bits and execution of the JMP microinstruction can transfer microprogram control to any one of the 2048 microinstructions addressable by the microprocessor Control Chip.

The JUMP format microinstructions require two microcycles for execution because the normal microaddress operation procedure is modified.

4.2.2 Conditional Jump Microinstruction Format

The Conditional Jump Microinstruction Format is shown in Figure 4.3. These microinstructions conditionally affect microprogram control based on the flag register contents. The flags are tested singly or in combinations, depending upon the condition field, MI<11:8>, contents. All conditional jump microinstructions contain the same opcode (MI<15:12>) value (01). The opcode and condition fields, taken together, determine an effective 8-bit opcode value which is used to uniquely identify the microinstruction.

The four condition field bits (MI<11:8> determine one of 16 possible jump conditions. The conditions correspond to the set or cleared state of any one of the ALU status bit and condition code flags, with the exeception of C4. In addition, two conditional jump microinstructions test the state of an Indirect Condition Code flag. The ICC flag is either controlled directly by the contents of an LSI-11 machine instruction, or by a combination of ALU condition code flags in response to a machine instruction. The ICC flag is updated only when an Input Word microinstruction is executed, normally during a Fetch machine instruction operation. The ICC test mechanism allows machine-level control of the micromachine operation and is used to implement the LSI-11 Branch instructions. The condition code flag processing is described in section 4.4.

All conditional jump microinstructions require 2 micromachine cycles to execute regardless of whether the condition is satisfied (and a jump is performed).

4.2.3 Literal Microinstruction Format

The Literal Microinstruction Format is shown in Figure 4.4. These microinstructions provide 8-bit constants in the microprogram. There are 5 microinsructions in this group and the octal opcode values (MI<15:12>) in the range 02 to 06. An 8-bit literal field, (MI<11:3>), is one operand The other operand is specified by the A Register Field (MI<3:0>) which supports direct or indirect addressing of any 8-bit byte register through the A Read/write port. All literal format microinstructions require 1 microcycle for execution.

4.2.4 Register Microinstruction Format

Most of the LSI-11 microinstructions belong to the Register Microinstruction Format which is illustrated in Figure 4.5. The opcode field occupies 8 bits, MI<15:8>, and opcode values range from 07 to 37 (octal). Most register format microinstructions unconditionally update the status bit flags relevant to the particular operation. The condition code flags are selectively updated by opcode choice, the odd opcode value (in general) affecting the condition code flags in addition to the status bit flags.

4.2.4.1 Byte and word Microinstructions - All register format microinstructions belonging to the data manipulation and the data access classifications are either byte or word microinstructions. The byte/word distinction is a direct consequence of the fact that the register file access ports are only 8-bits wide. This 8-bit width is also shared by the ALU inputs and output. A data manipulation microinstruction which produces a single 8-bit byte as its result can be completed in one microcycle. An example is the Add Byte microinstruction which forms the binary addition of the bytes addressed by the B and A register fields and places the result in the register designated by the A field. The microprocessor Data chip simultaneously presents the two 8-bit operands to the ALU and restructures the A register port path to load the ALU output when the result is formed.

In the case of a word microinstruction, the 8-bit register port and ALU width requires a 2-microcycle sequence to complete processing of 16-bit word operands. Carry or borrow information which needs to be transmitted between byte operations is stored in the ALU status bit flags. The ALU status bit flags are unconditionally updated by each byte operation. The Add Word microinstruction is the 16-bit, 2-microcycle counter-part of the Add Byte microinstruction. Since the

B and A register fields of the microinstructions each point to a single 8-bit byte within the register file, special techniques are used to form the 16-bit operands for word microinstructions. These techniques are explained in the next section.

There are several microinstructions which produce a word operand but still fall into the byte classification. This is because the Data chip organization allows the required 16-bit path to be simultaneously formed from two 8-bit paths. An example is the WRITE microinstruction, which is able to simultaneously present the contents of two independently specified registers to the data address buffers, and thus to the WDAL data chip outputs. No special register addressing techniques are needed in such cases and the microinstruction can complete execution in a minimum of one microcycle.

4.2.4.2 Word Operand Formation - The microinstruction register fields, MI<7:4> and MI<3:0>, are interpreted as shown below to form 16 bit operands:

All word operations other than right shifts:

A Register Field - (A) Even	Normal Use
A Register Field - (A) Odd	Bytes Reversed
B Register Field - (B) Even	Normal Use
B Register Field - (B) Odd	Sign Extension

Right shift word operations:

A Register Field - (A) Odd	Normal Use
A Register Field - (A) Even	Not Recommended
B Register Field - (B) Odd	Normal Use
B Register Field - (B) Even	Not Recommended

Normal Use: A and B Register Field argument is even.

The majority of word operand processing will be performed when both the A and B field register arguments are even. During the first microcycle of a word microinstruction, the 2 register fields designate the 8-bit operands. During the second microcycle the low-order bit in each register of the fields is logically complemented, thus pointing to the next higher register locations. For example, if A and B are 00 and 10 (octal) during the first microcycle, the effective arguments become 01 and 11 during the second microcycle. The actual microinstruction fields remain unchanged. This case is illustrated in Figure 4.6.

The register field assembly arguments reflect this convention. The bus address register symbol, RBA is equivalent to 12. To perform an operation on the lower byte of this register, the symbol RBAL would be used, which also assembles as 12. RBAH, the corresponding high byte assembles as 13.

Sign Extension: B register field argument odd

If the B field argument is assembled with the low-order bit equal to 1, the B field contents point to the first byte to be processed. During the second cycle, however, the high byte is formed by extending the high-order bit of the first or low byte through 8 bit positions. This case is illustrated in Figure 4.7.

Bytes Reversed: A register field argument odd

The sign extension procedure presented above does NOT apply in the case of an A register field odd argument. Instead, the low-order bit of the A field is complemented for the second cycle, and the byte register accessed is the Next Lower byte. Thus the word operand input via the A field is byte-swapped, as illustrated in Figure 4.8.

Since the A field also determines the destination of the ALU output, the abnormal reversed register coding sequence is apparant in the final register file contents. This case is illustrated in Figure 4.9, which shows that the low-order 8-bits of the word result are deposited in the high or odd byte and the high-order 8-bits are deposited in the low byte.

Right Shift Word Operations

Since all Right Shift Word operations begin with the left-most part of the word operand, both the A and B field argument must be odd. Left Shift Word operations follow the Normal Use conventions above.

4.3 MICROINSTRUCTION SET FUNCTIONAL ORGANIZATION

Because of the large number of microinstructions in the microprocessor repetroire, it is useful to establish a system of organization along functional lines. The three major functional classifications are Data Manipulation, Data Access, and Microprogram Control. Subclassifications of microinstructions exhibit variations of a single basic operation. For example, under the Data Manipulation classification there is the general subclassification "Move". The Move operation may be employed by the microprogrammer in 8 different variations as listed below:

Move Byte

Move Byte (Flags)

Move Word

Move Word (Flags)

Conditionally Move Byte

Conditionally Move Word

Conditionally Move Byte (Flags)

Conditionally Move Word (Flags)

The final description of a microinstruction is thus determined by whether it is a byte or word microinstruction, whether the indicated operation is performed conditionally or unconditionally, and whether it updates the ALU condition code flags.

4.3.1 Data Manipulation Microinstructions

The microinstructions in this group provide the bulk of the micromachine data manipulation ability. The subclassifications here are: Move, Increment/Decrement, Logical, Shift, and Arithmetic. The operations performed by all microinstructions in this classification are completed within the micromachine and do not require I/O. These microinstructions affect microprogram control only in that they update the ALU status bit and condition code flags and that some are executed conditionally.

4.3.1.1 Move Microinstructions - The common feature of the Move Microinstructions is that they move data between locations within the micromachine. These locations are exclusively within the microprocessor Data and Control chips, with the exception of the Load Literal microinstruction which moves a constant from the Load Literal microinstruction to a designated register. The Microinstruction mnemonics are shown below:

LL	Load Literal
LGL	Load G(Register) Low
LTR	Load Translation Register
CFR CCF	Copy Flag Register
LFR LCF	Load Flag Register
MB	Move Byte
MBF	Move Byte (Flags)
CMB	Conditionally Move Byte
CMBF (Flags)	Conditionally Move Byte
MW	Move Word

MWF	Move Word (Flags)
CMW	Conditionally Move Word
CMWF (Flags)	Conditionally Move Word

4.3.1.2 Increment / Decrement Microinstructions - The Increment/Decrement microinstructions operate only on register contents. Note that there is no means of incrementing or decrementing the G register contents. The microinstruction mnemonics are shown below:

ICB1	Increment Byte By 1
ICB1F	Increment Byte By 1 (Flags)
ICB2	Increment Byte By 2
ICB2F	Increment Byte By 2 (Flags
CICB1 By 1	Conditionally Increment Byte
DB1	Decrement Byte By 1
DB1F	Decrement Byte By 1 (Flags)
CDB1 By 1	Conditionally Decrement Byte
ICW1	Increment Word By 1
ICW1F	Increment Word By 1 (Flags)
ICW2	Increment Word By 2
ICW2F	Increment Word By 2 (Flags)
DW1	Decrement Word By 1
DW1F	Decrement Word By 1 (Flags)

4.3.1.3 Logical Microinstructions - The Logical Microinstructions perform variations of the basic unary and binary operations: And, And Complement, Or, Exclusive Or, Ones Complement, and Twos Complement. The microinstruction mnemonics are shown below:

NL	And Literal
NB	And Byte
NBF	And Byte (Flads)
NCB	And Complement Byte
NCBF	And Complement Byte (Flags)
NW	And Word
NWF	And Word (Flags)
NCW	And Complement Word
NCWF	And Complement Word (Flags)
ORB	Or Byte
ORBF	Or Byte (Flags)
ORW	Or Word
ORWF	Or Word (Flags)
XB	Exclusive Or Byte
XBF	Exclusive Or Byte (Flags)
XW	Exclusive Or Word

XWF	Exclusive Or Word (Flags)
OCB	Ones Complement Byte
OCBF	Ones Complement Byte (Flags)
OCW	Ones Complement Word
OCWF	Ones Complement Word (Flags)
TCB	Twos Complement Byte
TCBF	Twos Complement Byte (Flags)
TCW	Twos Complement Word
TCWF	Twos Complement Word (Flags)

4.3.1.4 Shift Microinstructions - The Shift Microinstructions are divided into 8 right shift and 8 left shift operations. Those shift operations which execute "With Carry" implement 8 or 16 bit shift registers using the C condition code flag as the seventh or seventeenth bit respectively. The microinstruction menmonics are shown below:

SLB	Shift Left byte
SLBF	Shift Left Byte (Flags)
SLBC	Shift Left Byte With Carry
SLBCF (Flags)	Shift Left byte With Carry
SLW	Shift Left Word
SLWF	Shift Left Word (Flags)
SLWC	Shift Left Word With Carry
SLWCF (Flags)	Shift Left Word With Carry
SRB	Shift Right Byte
SRBF	Shift Right Byte (Flags)
SRBC	Shift Right Byte With Carry
SRBCF (Flags)	Shift Right Byte With Carry
SRW	Shift Right Word
SRWF	Shift Right Word (Flags)
SRWC	Shift Right Word Carry
SRWCF	Shift Right Word Carry (Flags)

4.3.1.5 Arithmetic Microinstructions - The Arithmetic Microinstructions consist mainly of operations which either add or subtract values expressed in twos complement representation. One microinstruction facilitates decimal arithmetic (CAD). The microinstruction mnemonics are shown below:

AB	Add Byte
ABF	Add Byte (Flags)
ABC	Add Byte with carry
ABCF	Add Byte with Carry (Flags)
CAB	Conditionally Add Byte
CABF	Conditionally Add Byte (Flags)
AW	Add Word

AWF	Add Word (Flags)
AWC	Add Word With Carry
AWCF	Add Word With Carry (Flags)
CAW	Conditionally Add Word
CAWF	Conditionally Add Word (Flags)
CAWI	Conditionally Add Word On ICC
CAWIF	Conditionally Add Word On ICC
(Flags)	
CAD	Conditionally Add Digits
SB	Subtract Byte
SBF	Subtract Byte (Flags)
SBC	Subtract Byte With Carry
SBCF	Subtract Byte With Carry
(Flags)	
SW	Subtract Word
SWF	Subtract Word (Flags)
SWC	Subtract Word With Carry
SWCF	Subtract Word With Carry
(Flags)	

4.3.2 Data Access Microinstructions

The Data Access Microinstructions provide the only means for transferring data into or out of the micromachine. The 16-bit transfer path is formed by the microprocessor Data chip WDAL connections which are bidirectionally buffered to the 16 LSI-11 system bus lines, BDAL <15:00>.

The LSI-11 machine input operation (DATI) is implemented by a Read microinstruction followed by an Input microinstruction. Similarly, the machine output operation (DATO) is implemented by a Write-Output sequence of microinstructions. These two basic sequences are available in seveval variations to accomodate Byte transfers (DATOB) and read-modify write operations (DATIO, DATIOB). Further variations allow for special handling of interrupt transactions.

It is the data access group of microinstructions (only) which activate the microprocessor control chip I/O control signal lines. The lines which carry control information to the LSI-11 system bus interface logic are WSYNC, WWB, WDIN, WDOUT, and WIACK. The REPLY and BUSY inputs to the control chip are monitored during a data access operation. The logic circuitry which interfaces these signals to the LSI-11 system bus is discussed in The Microcomputer Handbook.

These microinstructions are organized as four basic families of Read, Input, Write and Output. Further sequencing and timing details are presented in Chapter 5.

4.3.2.1 Read Microinstructions - The Read Microinstruction group consists of 6 microinstructions, all of which place a system bus device address on the data address lines. To facilitate sequential addressing, four microinstructions automatically increment the registers containing the address arguments. Because the A and B register ports may be read simultaneously, all read microinstructions place both the upper and lower Byte of the device address on BDAL <15:00> at the same time. All Read microinstructions except for those which increment an address word execute in one microcycle. However, before execution is initiated, the control chip checks REPLY H and BUSY H at PH3 to verify that no other system bus operations are still in progress. Because of the requirment to check REPLY H and BUSY H during PH3, the address information is not be placed on the bus until the following microcycle.

The address information is placed on WDAL <15:00> during PH1 and is cancelled at the end of PH4, thus making the address valid for one microcycle. All read microinstructions assert WSYNC during PH2 of the microcycle in which the address becomes valid. The Read Acknowledge (RA) microinstruction asserts WIACK simultaneously with WSYNC. In all cases WDIN is asserted during the PH4 which follows removal of the address data. None of the read group microistructions effect the condition code flags. The microinstruction mnemonics are shown below:

R	Read
RIB1	Read And Increment Byte By 1
RIB2	Read And Increment Byte By 2
RIW1	Read And Increment Word By 1
RIW2	Read And Increment Word By 2
RA	Read Acknowledge

4.3.2.2 Input Microinstructions - A member of the Input Microinstruction group completes the basic DATI or Read-Input operation. WDIN is asserted during PH4 following the removal of address information from the bus. This signal is interfaced to the system bus as BDIN L, which informs the addressed device to place its data on the bus for input to the microprocessor. In order for the input microinstruction to be executed to completion, WDINH and REPLY H must be asserted during PH3. Since only the A port register is capable of writing register contents, input word microinstructions require 2 microcycles to execute.

In some instances it is desirable to have the input operations execute to completion independent of the state of the REPLY H signal. The input status microinstructions execute immediately without being preceded by a read operation and without checking the REPLY H signal.

It should be noted that of all the possible Input Microinstructions, only the Input Word (IW) microinstructions can load the translation register (when the B register field contains the proper code). The B field argument of the Input Byte (IB) and Input Word (IW) microinstruction can also specify a Read-Modify-Write operation. All of the input microinstructions are available in a flag-affecting version.

The microinstruction mnemonics are shown below:

IB	Input Byte
IBF	Input Byte (Flags)
IW	Input Word
IWF	Input Word (Flags)
ISB	Input Status Byte
ISBF	Input Status Byte (Flags)
ISW	Input Status Word
ISWF	Input Status Word (Flags)

4.3.2.3 Write Microinstructions - The Write Microinstructions are the first part of the Write-Output sequence which implements the DATO (Data-Output) transactions. Because the A and B register ports may be read simultaneously, Write Microinstructions place both the upper and lower byte of the device address on $BDAL_L$ <15:00> at the same time. All Write Microinstructions, except for those which increment an address word, execute in one microcycle. However, before execution is initiated, the Control Chip checks REPLY H and BUSY H at PH3 to verify that no other system bus operations are in progress. Because of the requirement to check REPLY H and BUSY H during PH3, the address information cannot be placed on the bus until the following microcycle. The address information is placed on BDAL <15:00> during PH1 and is cancelled at the end of PH4, thus making the address valid for one microcycle. All Write Microinstructions assert WSYNC during PH2 of the microcycle in which the address is valid. The Write Acknowledge (WA) microinstruction asserts WIACK H simultaneously with WSYNC. The microinstruction mnemonics are shown below:

W	Write
WIB1	Write And Increment Byte By 1
WIB2	Write And Increment Byte By 2
WIW1	Write And Increment Word By 1
WIW2	Write And Increment Word By 2
WA Has no use to microprogrammer -	Write Acknowledge

4.3.2.4 Output Microinstructions - An Output Microinstruction is used to complete the basic DATO or Write/Output operation, or to complete a Read-Modify-Write operation (DATIO or DATIOB). WDOUT is asserted by an Output Microinstruction during PH1 following the removal of address information from the bus. This signal is interfaced to the system bus as BDOUT L, which causes the addressed device to accept the data placed on the bus by the microprocessor Data chip. In order for an Output Microinstruction to be completed, REPLY H must be asserted by the addressed device during PH3. Both register A and B ports provide for reading or accessing register contents, so for Output Word and Output Status Word microinstructions all 16-bits can be output during one microcycle. In some instances it is desirable to complete an output operation completion independent of the state of the REPLY H signal. The Output Status microinstructions execute immediately without being preceded by a write microinstruction and without checking the REPLY H signal. The microinstruction mnemonics are shown below:

OB	Output Byte
OW	Output Word
OSW	Output Status Word

4.3.3 Microprogram Control Microinstructons

The Microprogram Control Microinstructions control or modify the microinstruction flow. The only available control methods not covered by this group are (1) the RSVC bit (Read Next Instruction/Interrupt and Trap Service) and (2) the microprocessor Control chip RESET line.

4.3.3.1 Conditional / Unconditional Jump and Return Micro- - instructions

This group contains all microinstructions which alter the microinstruction flow (both conditionally and unconditionally). The Modify Microinstruction (MM) - Jump (JMP) sequence allows the microprogrammer to determine the jump microaddress from data in one of the micromachine registers. The microinstruction mnemonics are shown below:

JMP	Jump (Unconditionally)
RFS	Return From Subroutine
JZB0	Jump if ZB=0
JZB1	Jump if ZB=1
JNB0	Jump if NB=0
JNB1	Jump if NB=1
JIC0	Jump if Indirect Condition Code=0
JIC1	Jump if Indirect Condition Code=1
JC80	Jump if C8=0
JC81	Jump if C8=1
JN0	Jump if N=0
JN1	Jump if N=1
JZ0	Jump if Z=0
JZ1	Jump if Z=1
JV0	Jump if V=0
JV1	Jump if V=1
JC0	Jump if C=0
JC1	Jump if C=1
MM	Modify Microinstruction

1 ≡ True
Ø ≡ False

MM ≡ MI

4.3.3.2 Compare and Test Microinstructions - This group consists of five arithmetic compare and five logical test microinstructions. The microinstruction mnemonics are shown below:

CL	Compare Literal
CB	Compare Byte
CBF	Compare Byte (Flags)
CW	Compare Word
CWF	Compare Word (Flags)
TL	Test Literal
TB	Test Byte
TBF	Test Byte (Flags)
TW	Test Word
TWF	Test Word (Flags)

4.3.3.3 Miscellaneous Control Microinstructions - This group of microinstructions affects control of (1) the interrupt register contents, (2) the ALU flag register contents, (3) the Translation Register contents, and (4) the Translation State Register (TSR) contents. Also included here is the No Operation (NOP) microinstruction. The micoinstruction mnemonics are shown below:

RI		Reset Interrupts
SI		Set Interrupts
CRF	CCF	Copy Status And Condition Flag Register
LFR	LCF	Load Status And Condition Flag Register
LTR	LCT	Load Translation Register
RTSR		Reset Translation State Register
NOP		No Operation

CHAPTER 5

MICROPROGRAMMING LSI-11 SYSTEM BUS TRANSACTIONS

5.1 GENERAL

The LSI-11 processor module is designed using a microprogramable microprocessor chip set to emulate a PDP-11. Since the LSI-11 bus is more powerful than the microprocessor I/O structure, some additional logic is used to implement this added capability. Consequently, even though LSI-11 bus I/O operations are easy to microprogram, the actual implementation must be examined. The schematic diagram of the LSI-11 CPU should be referenced for the discussion to follow.

5.2 LSI-11 SYSTEM BUS INTERFACE LOGIC OVERVIEW

The LSI-11 processor module contains hardware logic circuitry which interfaces the microprocessor to the LSI-11 system bus. This logic circuitry accomplishes basically two functions: (1) it modifies the sequencing and timing of control signals which appear at the Data chip, and (2) it provides electrical buffering between the MOS LSI microprocessor and the TTL system bus. This discussion relates directly to the material in The Microcomputer Handbook. The reference provides two useful logic drawings. Bus I/O Control Serial Logic" and "Interrupt Control and Reset Logic". An additional Handbook reference is the timing diagrams for all basic system bus transations.

The format of this overview is to list each control signal first in its original form and then in its final form. For example, the microprocessor originates WSYNC H which is processed by hardware logic circuitry and becomes the system bus signal BSYNC L. The overview discusses 7 microprocesor-system bus interface signals and one control signal which is only input to the control chip.

5.2.1 WSYNC H - BYSNC L

WSYNC H is asserted by the control chip as part of the execution of any Read or Write microinstruction. In all cases, this signal appears at the beginning of PH2 following the microcycle in which the microinstruction was executed. The interface logic circuitry provides the following 3 functions:

-DELAYS The appearance of WSYNC H as BSYNC L until after PH3, by means of the SYNC Flip Flop which is clocked by the trailing edge of PH3L.

-MAINTAINS BSYNC L in the asserted state until after BRPLY L has been terminated by the slave device at the end of the transaction. This serves to keep the device recognition signal on the device itself stable until the device has completed its role in the bus transaction.

-INHIBITS the appearance of WSYNC H as BSYNC L during interrupt operations. The two acknowledge microinstructios RA and WA cause the control chip to assert WSYNC H simultaneously with WIAK H. The WIAK H signal forces the SYNC F/F to the reset state which inhibits BSYNC L. This is used to input an interrupt vector from an interrupting device.

5.2.2 WWB H - BWTBT L

WWB H is asserted by the control chip during the microcycle in which address information is placed on the bus. This signal appears immediately as BWTBT L, delayed only by the inverting buffer propagation delay.

5.2.3 WDIN H - BDIN L

WDIN H is asserted by the control chip during the microcycle which follows the transmission of address information. All Read microinstructions cause this signal to be asserted at the beginning of PH2. This signal appears immmediately as BDIN L, delayed only by two inverters and the inverting buffer propagation delay.

5.2.4 WDOUT H - BDOUT L

WDOUT H is asserted by the control chip at the beginning of the microcycle which follows transmission of address information. All output microinstructions cause this signal to be asserted at the beginning of

PH1. WDOUT H is also asserted as part of the DATIO bus transaction. The interface logic circuitry provides the following 4 functions:

-DELAYS the appearance of WDOUT H as BDOUT L until after PH2, by means of the DOUT F/F which is clocked by the leading edge of PH3.

-INHIBITS the appearance of BSYNC L until BRPLY has been passive long enough for the REPLY F/F to have been reset. The reset state of the REPLY F/F must be ANDed with WDOUT H in order for the DOUT F/F to be set. This mechanism assures that the system bus is clear of all slave device action before the addressed device is told (via BDOUT L) that the content of BDAL<15:00> is valid data.

-CANCELS BDOUT L as soon as the REPLY F/F has been set, in response to the receipt of BRPLY L. Since BRPLY L indicates that the data output by the processor has been accepted by the addressed device, there is no need to continue BDOUT L.

-INHIBITS the setting of the DOUT F/F when WSYNC H is passive.

5.2.5 BRPLY L - REPLY H

BRPLY L is asserted by the addressed (slave) device to indicate that the required slave action has been completed. This action is the placing of data on the bus in the case of an input operation or the acceptance of data from the bus in the case of an output operation. Due to the propagation delays along the system bus and to response delays in the slave device, BRPLY L appears asynchronously at the processor module. The interface logic circuitry provides the following 3 functions:

-DELAYS the effective appearance of the reply signal for use internal to the logic circuitry until the beginning of each new microcycle. This is accomplished by clocking the REPLY F/F with PH1.

-ENABLES the set state of the REPLY F/F to appear as REPLY H for input to the control chip only when a valid I/O operation is in progress (BSYNC L is active).

-CANCELS the BDOUT L signal by negating the signal which was ANDED with WDOUT H to set the DOUT F/F. This mechanism accelerates release of the system bus by the addressed device in the case of an output transaction.

5.2.6 BRPLY L - BUSY H

BRPLY L is more directly linked to the BUSY H input of the Control chip than it is to REPLY H. BUSY H is asserted immediately when the REPLY F/F is set and is not dependent on any enabling action. The BUSY H input is checked by the control chip during PH3 of both read and write microinstruction execution. If BUSY H is asserted, execution is suspended and the pending bus transfer is delayed until the system bus becomes free. BUSY H is also asserted by the DMA arbitration circuitry when another master device has control of the LSI-11 bus.

5.2.7 WIAK H - BIACK H

WIACK H is asserted by the control chip as part of the execution of either acknowledge microinstruction, RA or WA. This signal is asserted at the beginning of PH2, simultaneously with WSYNC. Because BSYNC L does not take part in LSI-11 system bus interrupt transactions, the WSYNC H - BSYNC L interface logic inhibits BSYNC L. The interface logic associated with the control chip WIAK H output provides the following 2 functions:

-DISABLES the INT ACK F/F (interrupt acknowledge) until the appearance of WDIN H.

-DELAYS the appearance of WIAK H, as BIACK H until one microcycle after PH2 of the assertion of BDIN L.

These two functions are necessary to assure that BDIN L arrives at the interrupting device before BIACK H. The transition of BDIN L from its passive to active state is used by the interrupting device to latch the state of the two interrupt acknowledge flip flops, thus arbitrating their interrupt priorities. This mechanism has the consequence that the write Acknowledge microinstrucitons will never produce a BIACK H signal because WDIN H is not asserted as part of a write operation.

5.3 THE DATA-INPUT (DATI) OPERATION

5.3.1 DATI Operation, Minimum Execution Time

Figures 5.1-1 and 5.2-2 illustrate the control signal action relevant to a DATI operation which executes with minimum possible delay. The microinstruction sequence presented is READ, INPUT WORD, Next Microinstruction:

```
R       B,A                 ; ADDRESS DEVICE, SIGNAL DIN
```

```
IW      B,A                   ; INPUT LOW AND HIGH BYTE
Next Microinstruction         ;
```

The individual events which occur in the DATI operation are discussed with the aid of Figures 5.1-1 and 5.2-2, which contain the Microcycle reference numbers.

microcycle 1 Read

The execution of the Read microinstruction in the first microcycle causes no change in the Control chip. The microinstruction executes to completion because REPLY H and BUSY H are not asserted during PH3.

Microcycle 2 Input Word (Wait)

Because the Read microinstruction executed to completion during Microcycle 1, the address information is placed on BDAL<15:00> at the beginning of PH1. WSYNC H is asserted at PH2, causing BYSNC L to be asserted at the beginning of PH4. Since the addressed device can only responsd to BDIN L, which has not yet been asserted, the INPUT WORD microinstruction check of REPLY H during PH3 fails and initiates the WAIT state.

Microcycle 3 Input Word (Wait)

Because the Read microinstruction executed to completion during Microcycle 1, WDIN H is asserted at the beginning of PH2. The assertion of BDIN L follows immediately. In response to BDIN L, the addressed device places data on BDAL<15:00> and asserts BRPLY L to the processor. However, the REPLY F/F is not clocked until PH_1 and REPLY H remains passive which maintains the WAIT state.

To assure minimum possible delay in the execution of the DATI operation illustrated here, BRPLY L must be received by the processor in time to set the REPLY F/F at the beginning of Microcycle 4. Otherwise, the WAIT state will continue throughout Microcycle 4 and the next opportunity to set the REPLY F/F will not occur until the beginning of Microcycle 5.

Microcycle 4 Input Word (Input Low Byte)

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H is now active, the WAIT state is terminated during the PH3 REPLY H check and the Data chip stores the low byte in its designated register.

Microcycle 5 Input Word (Input High Byte)

Since the PH3 check of REPLY H is still true the data chip stores the high byte in its designated register (determined by complementing the low-order bit of the A register field). No control signals are changed by either the processor or the addressed device.

Microcycle 6 Next Microinstruction (Possible WAIT)

Because the Input Word microinstruction executed to completion during Microcycle 5, both WSYNC H and WDIN H are made passive at the end of PH1. Since WDIN H must be asserted to enable REPLY H during a DATI operation, REPLY H also goes passive. BDIN L goes passive immediately, which informs the addressed device to remove its data from BDAL<15:00>. This data is removed about the same time that BRPLY goes passive at the processor. Since the SYNC F/F is locked into its set state due to the set state of the REPLY F/F, BSYNC L remains asserted.

If the NEXT MICROINSTRUCTION belongs either to the Read or Write group, the state of BUSY H is checked and a new WAIT state will be initiated which delays the beginning of the next I/O operation. Any other microinstruction will execute. To assure minimum possible delay in the execution of the DATI operation illustrated here, BRPLY L must go passive at the processor in time to reset the REPLY F/F at the beginning of Microcycle 7. Otherwise the next opportunity to set the REPLY F/F will not occur until the beginning of Microcycle 8, and a possible WAIT state initiated during Microcycle 6 will continue at least into Microcycle 8.

Microcycle 7 Next Microinstruction

Since BRPLY L was negated by the addressed device during Microcycle 6, the REPLY F/F is clocked to the reset state at the beginning of PH1, which cancels BUSY H. The reset state of the REPLY F/F also allows the SYNC F/F to be clocked to the reset state at the end of PH3, which makes BSYNC L passive.

FIGURE 5.1-1
DATI MIN 1

FIGURE 5.1-2
DATI MIN2

5.3.2 DATI Operation, Delayed Execution Time

Figures 5.2-1 and 5.2-2 illustrate the control signal action relevant to a DATI operation in which execution is delayed at 3 points. The delays occur during the execution of the Read microinstruction and in the response of the addressed device to both the assertion and negation of BDIN L. The microinstruction sequence presented here is Read, Input word, Next Microinstruction:

```
R       B,A                      ; ADDRESS DEVICE, SIGNAL DIN
IW      B,A                      ; INPUT LOW AND HIGH BYTE
Next Micrinstruction             ;
```

The individual events which occur in the DATI Operation are discussed with the aid of Figure 5.2-1 and 5.2-2, which contain the Microcycle reference numbers.

Microcycle 1 Read (Wait)

The BUSY H input to the Control chip is asserted during the entire microcycle which causes the PH3 check performed by the READ microinstruction to fail. The WAIT state is initiated during PH3, For complete accuracy, other control signal actions should be apparent during microcycle 1 which relate to the conclusion of a previous I/O transaction. These actions have been omitted from the figures.

Microcycle 2 Read

The BUSY H signal goes passive prior to the PH3 check, Since the BUSY H check is passed, the Wait state is terminated and the Read microinstruction executes to completion.

Microcycle 3 Input word (Wait)

Because the READ microinstruction executed to completion during Microcycle 2, the address information is placed on BDAL<15:0> at the beginning of PH1. WSYNC H is asserted at PH2, causing BSYNC L to be asserted at the beginning of PH4. Since the addressed device can only respond to BDIN L, which has not yet been asserted, the Input Word microinstruction check of REPLY H during PH3 fails and initiates the wait state.

Microcycle 4 Input word (Wait)

Because the Read microinstruction executed to completion during Microcycle 2, WDIN H is asserted at the beginning of PH2. The assertion of BDIN L follows immediately. Since there is no change in the state of REPY H, the WAIT state is maintained.

Microcycle 5 Input word (Wait)

In response to BDIN L asserted during Microcycle 4, the addressed device asserts BRPLY L and places data on BDAL<15:00>. Since there is no change in the state of REPLY H, the wait state is maintained.

If BRPLY L had been asserted during Microcycle 4, REPLY H would have become active during Microcycle 5. As illustrated, the additional delay in device response adds one full microcycle to the execution of the DATI operation.

Microcycle 6 Input Word (Input Low Byte)

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the WAIT state is terminated during the PH3 REPLY H check and the Data Chip stores the low byte in the designated register.

Microcycle 7 Input Word (Input High Byte)

Since the PH3 check of REPLY H is again passed, the data chip stores the high byte in its designated register (determined by complementing the low order bit of the A register field). No other control signals are changed by either the processor or the addressed device.

Microcycle 8 Next Microinstruction (Possible WAIT)

Because the Input Word microinstruction executed to completion during Microcycle 7, both WSYNC H and WDIN H are made passive at the end of PH1. Since WDIN H must be asserted to enable REPLY H during a DATI operation, REPLY H also goes passive. BDIN L goes passive immediately, which informs the addressed device to remove its data from BDAL<15:00>. Since the SYNC F/F is locked into its set state due to the set state of the REPLY F/F, BSYNC L remains asserted.

If the next microinstruction belongs either to the read or write group, the state of REPLY H and BUSY H is checked and a new WAIT state will be initiated which delays the beginning of the next I/O operation. Any other microinstruction will execute.

Microcycle 9 Next Microinstruction (Possible WAIT)

In response to BDIN L negated during Microcycle 8, the addressed device negates BRPLY L and removes its data from BDAL<15:00>. Since there is no change in the state of BUSY H the possible WAIT state would be maintained.

Microcycle 10 Next Microinstruction

Since BRPLY L was negated by the addressed device during Microcycle 9, the REPLY F/F is clocked to the reset state at the beginning of PH1, which cancells BUSY H. The reset state of the REPLY F/F also allows the SYNC F/F to be clocked to the reset state at the end of PH3, which makes BSYNC L passive.

Device Response Note

If BRPLY L had been negated during Microcycle 8, BUSY H would have become passive during Microcycle 9. As illustrated, the additional delay in device response adds one full microcycle to the execution of the DATI operation.

FIGURE 5.2-1
DATI DELAYED 1

FIGURE 5.2-2
DATI DELAYED 2

5.3.3 DATI Microprogramming Summary

The DATI operation, as implemented by the READ-INPUT microinstruction sequence, allows for variable delays in addressed device resonse to both the assertion and negation of BDIN L. In any conventional machine configuration the device responses will likely not be the minimums discussed in section 5.3.1. To make better use of what would otherwise be wait-induced microprocessor idle time, data manipulation microinstructions may be inserted between the Read and Input microinstructions. The time required to execute the inserted data manipulation microintructions should optimatlly be close to, but less than, the worst case maximum device response to the assertion of BDIN L. In this way the microprogram will execute with minimized or even zero overall delay. No difficulty is encountered if addressed device response causes REPLY H to be asserted before completion of the inserted data manipulation microinstruction sequence is completed. This condition merely delays, for a different reason, the eventual completion of the DATI operation.

Once the INPUT microinstruction has executed to completion, the addressed device is expected to remove data and negate BRPLY L. The microproessor may be best used at this point by executing microinstrucitons which do not check the status of either REPLY H or BUSY H.

To summarize, the delay in execution of a microprogram containing a DATI operation may be minimized if:

1. The microinstruction immediately preceding the READ microinstruction does not cause BUSY H or REPLY H to be asserted.

2. The addressed device responds to the assertion of BDINL in minimum time.

3. The addressed device responds to the negation of BDIN L in minimum time.

4. The microinstruction immediately following the INPUT microinstruction does not check REPLY H or BUSY H.

As an alternative to condition 2 above, unavoidable delays may be utilized by data manipulation microintructions.

5.4 THE DATA-OUTPUT (DATO) OPERATION

5.4.1 DATO Operation, Minimum Execution Time

Figure 5.3 illustrates the control signal action relevant to a DATO operation which executes with minimum possible delay. The microinstruction sequence presented is Write, Output Word, Next Microinstruction:

```
W       B,A                 ; ADDRESS DEVICE, SIGNAL DOUT
OW      B,A                 ; OUTPUT WORD
Next Microinstruction
```

The individual events which occur in the DATO Operations are discussed with the aid of Figure 5.3, which contains the Microcycle reference numbers.

Microcycle 1 Write

The execution of the Write microinstruction in the first microcycle causes no change in the control lines. The microinstruction executes to completion because REPLY H and BUSY H are unasserted during PH3.

Microcycle 2 Output Word (Wait)

Because the Write microinstruction executed to completion during Microcycle 1, the address information is placed on BDAL<15:00> at the beginning of PH1. The WWB H signal is asserted at the beginning of PH1 which indicates a write operation. The assertion of BWTBT L follows immediately. WSYNC H is asserted at PH2, causing BSYNC L to be asserted at the beginning of PH4. Since the addressed device can only respond to BDOUT L, which has not yet been asserted, the Output Word microinstruction check of REPLY H during PH3 fails and initiates the WAIT state.

Microcycle 3 Output Word (Wait)

Because the Output Word microinstruction executed up to the failing check of REPLY H during Microcycle 2, WDOUT H is asserted at the beginning of PH1. The REPLY F/F is in the reset state during this microcycle, which allows the asserted WDOUT H to set the DOUT F/F at the beginning of PH3. When the DOUT F/F is set, BDOUT L is asserted. Since this example contains the Output Word microinstruction, WWBH and thus BWTBT L are negated at the beginning of PH1. BWTBT L remains asserted past this point only in the case of a byte output Operation. Also during PH1, the Data chip simultaneously places two bytes (a full word) of data on BDAL<15:00>. In response to the assertion of BDOUT L, the addressed device accepts the data output by the processor and returns BRPLY L. However, the REPLY F/F is only clocked at PH1 and REPLY H remains passive which maintains the WAIT state.

To assure minimum possible delay in the execution of the DATO operation illustrated here, BRPLY L must be received by the processor in time to set the REPLY F/F at the beginning of Microcycle 4. Otherwise, the wait state will continue throughout Microcycle 4 and the next opportunity to set the REPLY F/F will not occur until the beginning of Microcycle 5.

Microcycle 4 Output Word

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the wait state is terminated at the PH3 REPLY H check and the Output Word microinstruction executes to completion. The set state of the REPLY F/F also negates the DOUT F/F input, and it is clocked to the reset state at the beginning of PH3 which negates BDOUT L and also cancels REPLY H. Note that the Control chip determines the state of REPLY H at the beginning of PH3, thus recognizing the addressed device response before REPLY H is canceled via the resetting of the DOUT F/F. In response to the negation of BDOUT L, the addressed device negates BRPLY L.

To assure minimum possible delay in the execution of the DATO operation illustrated, BRPLY L must go passive at the processor in time to reset the REPLY F/F at the beginning of Microcycle 5. Otherwise the next opportunity to reset the REPLY F/F will not occur until the beginning of Microcycle 6.

Microcycle 5 Next Instruction

Since BRPLY L was negated during Microcycle 4, the REPLY F/F is clocked to the reset state at the beginning of PH1, thus cancelling BUSY H. Because the Output Word microinstruction executed to completion during Microcycle 4, WSYNC H is made passive at the beginning of PH2 and WDOUT H is made passive at the beginning of PH4. The Data chip removes the output word from BDAL<15:00> at the end of PH4. SYNC L is clocked to the passive state at the end of PH3.

Any type of microinstruction may be executed during Microcycle 5, since the REPLY F/F was reset at the beginning of PH1, thus negating BUSY H.

FIGURE 5.5
DATO MIN 1

5.4.2 DATO Delayed Execution Time

Figures 5.4-1 and 5.4-2 illustrate the control signal action relevant to a DATO operation in which execution is delayed at 3 points. The delays occur during the execution of the Write microinstruciton and in the response of the addressed device to both the assertion and negation of BDOUT L. The microinstruction sequence presented here is Write, Output Word, Next Microinstruction:

```
W       B,A                   ; ADDRESS DEVICE, SIGNAL DOUT
OW      B,A                   ; OUTPUT WORD
Next Microinstruction
```

The individual events which occur in the DATI Operation are discussed with the aid of Figures 5.4-1 and 5.4-2 which contain the microcycle reference numbers.

Microcycle 1   Write (Wait)

The BUSY H input to the Control chip (not shown in the timing diagram) is asserted during the entire microcycle which causes the PH3 check performed by the Write microinstruction to fail. The WAIT state is initiated during PH3. For complete accuracy, other control signal actions should be apparant during Microcycle 1 which relate to the conclusion of a previous I/O transaction. These actions have been omitted from the figures.

Microcycle 2   Write

The BUSY H signal goes passive prior to the PH3 check. Since the BUSY H check is passed, the Wait state is terminated and the Write microinstruction executes to completion.

Microcycle 3   Output Word (Wait)

Because the Write microinstruction executed to completion during Microcycle 2, the address information is placed on BDAL<15:00> at the beginning of PH1.

The WWB H signal is asserted at the beginning of PH1 which indicates a Write operation. The assertion of BWTBT L follows immediately.

WSYNC H is asserted at PH2, causing BSYNC L to be asserted at the beginning of PH4. Since the addressed device can only respond to BDOUT L, which has not yet been asserted, the Output Word microinstruction check of REPLY H during PH3 fails and initiates the WAIT state.

Microcycle 4   Output Word (Wait)

Because the Output Word microinstruction executed until the failing check of REPLY H during microcycle 3; WDOUT H is asserted at the beginning of PH1. The REPLY F/F is in the reset state during this microcycle which allows the asserted WDOUT H to set the DOUT F/F at the beginning of PH3. When the DOUT F/F is set, BDOUT L is asserted. Since this example contains the Output Word microinstruction, WWBH and thus BWTBT L are negated at the beginning of PH1. BWTBT L remain asserted past this point only in the case of a byte output operation. Also during PH1, the Data chip simultaneously places two bytes (a full word) of data on BDAL<15:00>. Since REPLY H is unchanged in this microcycle, the WAIT state is maintained.

Microcycle 5 Output Word (wait)

In response to BDOUT L, the addressed device accepts the data on BDAL<15:00> and returns BRPLY L to the processor. However, the REPLY F/F is only clocked at PH1 and REPLY H remains passive which maintains the WAIT state.

If BRPLY L had been asserted during Microcycle 4, REPLY H would have become active during Microcycle 5. As illustrated, the additional delay in device response adds one full microcycle to the execution of the DATO operation.

Microcycle 6 Output Word

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the WAIT state is terminated at the PH3 REPLY H check and the Output Word microinstruction executes to completion. The set state of the REPLY F/F also negates the DOUT F/F input, and it is clocked to the reset state at the beginning of PH3 which negates BDOUT L and also cancels REPLY H. Note that the Control chip determines the state of REPLY H at the beginning of PH3, thus recognizing the response of the addressed device before REPLY H is cancelled via the resetting of the DOUT F/F.

Microcycle 7 Next Microinstruction (Possible WAIT)

Because the Output Word microinstruction executed to completion during Microcycle 6, WSYNC H is made passive at the beginning of PH2 and WDOUT H is made passive at the beginning of PH4. The Data chip removes the output word from BDAL<15:00> at the end of PH4. In response to the negation of BDOUT L, the addressed device negates BRPLY L. However, since the REPLY F/F is clocked at PH1, it remains in the set state which asserts BUSY H.

If the Next Microinstruction belongs either to the read or write group, the state of BUSY H is checked and a new wait state will be initiated which delays the beginning of the next I/O operation. Any other microinstruction will execute immediately.

If BRPLY L had been negated during Microcycle 6, the REPLY F/F would have been reset at the beginning of Microcycle 7. As illustrated, the additional delay in device response adds one full microcycle to the execution of the DATO operation.

Microcycle 8 Next Microinstruction

Since BRPLY L was negated during Microcycle 7, the REPLY F/F is clocked to the reset state at the beginning of PH1, thus cancelling BUSY H. The reset state of the REPLY F/F also allows the SYNC F/F to be clocked to the reset state which negates BSYNC L.

If the microinstruction in Microcycle 7 caused the microprocessor to initiate the WAIT state, the microinstruction will now execute. Otherwise microinstruction execution proceeds normally.

FIGURE 5.4-1
DATO DELAYED 1

FIGURE 5.4-2
DATO DELAYED 2

5.4.3 DATO Microprogramming Summary

The DATO operation, as implemented by the Write-Output microinstruction sequence, allows for variable delays in addressed device response to both the assertion and negation of BDIN L. However, there is an important difference between the DATI and DATO operations regarding delay handling. To optimize microprocessor performance in the DATO context, the Output microinstruction must immediately follow the WRITE microinstruction. This is necessary because BDOUT L is a function of the Output microinstruction. In contrast, BDIN L is a function of a Read microinstruction. Therefore, there is no opportunity in the DATO context, to make use of idle time caused by the delay of the addressed device to respond to the assertion of BDOUT L.

Once BRPLY L is received, indicating that the addressed device has stored the output data, the LSI-11 system bus interface logic proceeds immediately to negate BDOUT L. This action accelerates the negation of BRPLY L by the addressed device. The microinstructions which follow the OUTPUT microinstruction should not be of the type which check REPLY H or BUSY H. Otherwise the resulting WAIT state will cause microprocessor idle time.

To summarize, the delay in the execution of a microprogram containing a DATO operation may be minimized if:

1. The microinstruction immedately proceeding the Write microinstruction does not cause BUSY H or REPLY H to be asserted.

2. The Write microinstruction is immedately followed by an Output microinstructio n.

3. The addressed device responds to the assertion of BDOUT L in minimum time.

4. The addressed device responds to the negation of BDOUT L in minimum time.

5. The microinstruction immediately following the Output microinstruction does not check REPLY H or BUSY H.

5.5 THE DATA-INPUT-OUTPUT (DATIO) OPERATION

5.5.1 DATIO Minimum Execution Time

Figures 5.5-1 and 5.5-2 illustrate the control signal action relevant to a DATIO operation which executes with minimum possible delay. The microinstruction sequence presented is Read, Input Word, Modify the data, Output Word, Next Microinstruction:

```
R       B,A                  ; ADDRESS DEVICE, SIGNAL DIN
Iw      B,A                  ; INPUT LOW AND HIGH BYTE.
Modify the data              ; MODIFY DATA
Ow      B,A                  ; SIGNAL DOUT, OUTPUT WORD
Next Microinstruction        ;
```

The individual events which occur in the DATIO operation are discussed with the aid of Figures 5.5-1 and 5.5-2, which contain the Microcycle reference numbers.

Microcycle 1   Read

The execution of the Read microinstruction in the first microcycle causes no change in the control lines. The microinstruction executes to completion because REPLY H and BUSY H (not shown) are unasserted at PH3.

Microcycle 2   Input Word (Wait)

The Control chip determines that a Read-Modify-Write operation is to be performed by means of the argument contained in the B register field of the Input Word microinstruction. It stores this information internally and will not conclude the DATIO operation until an output microinstruction is successfully completed.

Because the Read microinstruction executed to completion during Microcycle 1, the address information is placed on BDAL<15:00> at the beginning of PH1. WSYNCH is asserted at PH2, causing BSYNC L to be asserted at the beginning of PH4. Since the addressed device can only respond to BDIN L (during the input portion), which has not yet been asserted, the Input Word microinstruction check of REPLY H at PH3 fails and the wait state is initiated.

Microcycle 3   Input Word (Wait)

Because the Read microinstruction executed to completion during Microcycle 3, WDIN H (not shown) is asserted at the beginning of PH2. The assertion of BDIN L follows immediately. In response to BDIN L, the addressed device places data on BDAL <15:00> and returns BRPLY L to the processor. However, the REPLY F/F is not clocked until PH1 and REPLY H remains passive which maintains the wait state.

Microcycle 4   Input Word (Input Low Byte)

The asserted state of BRPLY sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the wait state is terminated during the PH3 REPLY H check and the data chip stores the low byte in a designated register.

Microcycle 5   Input Word (Input High Byte)

Since the PH3 check of REPLY H is again passed, the Data chip stores the high byte in a designated register (determined by complementing the low-order bit of the A register field). No control signals are changed by either processor or the addressed device.

To assure the minimum possible delay in the execution of the input portion of the DATIO operation, BRPLY L must be received at the processor in time to set the REPLY F/F at the beginning of Microcycle 4. Otherwise the WAIT state will continue throughout Microcycle 4 and the next opportunity to set the REPLY F/F will not occur until the beginning of Microcycle 5.

Microcycle 6 Modify Data

Because the Input Word microinstruction executed to completion during Microcycle 5, WDIN H (not shown) and BDIN L are negated at the end of PH1. The negation of WDIN H also cancels REPLY H. The Read-Modify-Write operation maintains WSYNC H in the asserted state. The addressed device responds to the negation of BDIN L by removing data from BDAL<15:00> and negating BRPLY L.

This example of the DATIO operation allows one microcycle for modification of the data retrieved by the Input Word microinstruction. Normal usage would probably require manipulation of the entire data word which would necessitate at least 2 microcycles. Since none of the data manipulation microinstructions check the REPLY H or BUSY H signals, execution can proceed without delay.

Microcycle 7 Output Word (Wait)

Since BRPLY L was negated by the addressed device during Microcycle 6, the REPLY F/F is clocked into the reset state at the beginning of PH1, which cancels BUSY H (not shown). Execution of the Output Word microinstruction proceeds up to the PH3 check of REPLY H. REPLY H has been cancelled and the WAIT state is initiated.

Microcycle 8 Output Word (Wait)

Because the Output WORD microinstruction executed up to the failing check of REPLY H during Microcycle 7, WDOUT H is asserted at the beginning of PH1. BDOUT L is asserted at the beginning of PH3.

Since this example contains the Output Word microinstruction, WWBH and BWTBT L are not asserted. In the case of an Output Byte microinstruction, the assertion of BWTBT L would begin at PH1 of Microcycle 8 and end at PH1 of Microcycle 9. Also, as a result of the partial execution of the Output Word microinstruction, the Data chip simultaneously places two bytes (a full word) of data on BDAL<15:00>.

In response to the assertion of BDOUT L, the addressed device accepts the data output by the processor, and asserts BRPLY L. However, the REPLY F/F is not clocked until PH1 and REPLY H remains passive which maintains the WAIT state.

Microcycle 9 Output Word

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the wait state is terminated at the PH3 REPLY H check and the Output Word microinstruction executes to completion. The set state of the REPLY F/F also negates the DOUT F/F input and it is clocked to the reset state at the beginning of PH3, which negates BDOUT L and also cancels REPLY H. Note that the Control chip determines the state of REPLY H at the beginning of PH3, thus recognizing the addressed device response before REPLY H is cancelled via the resetting of the DOUT F/F.

Microcycle 10 Next Microinstruction

The negation of BRPLY L causes the REPLY F/F to be reset at the beginning of PH1 which negates BUSY H (not shown). Because the Output Word microinstruction executed to completion during Microcycle 9, WSYNC H is made passive at the beginning of PH2 and WDOUT H is made passive at the beginning of PH4. The Data chip removes the output word from BDAL<15:00> at the end of PH4. BSYNC L is clocked to the passive state at the end of PH3.

Any type of microinstruction may be executed during Microcycle 10, since the REPLY F/F was reset at the beginning of PH1, thus negating BUSY H.

In this example of a DATIO operation, both REPLY H and BUSY H (not shown) are passive during Microcycle 10. As a result, any type of microinstruction may be executed. However, if the negation of BRPLY L received by the processor is delayed past the beginning of PH1 of Microcycle 10, BUSY H would remain active, thus delaying the start of any new I/O operations.

FIGURE 5.5-1
DATIO MIN 1

FIGURE 5.5-2
DATIO MIN PLUS MOD 2

5.5.2 DATIO Microprogramming Summary

The DATIO operation, as implemented by the Read-Input-Modify-Output microinstruction sequence, allows for variable delays in addressed device response to both the assertion and negation of BDIN L and also to both the assertion and negation of BDOUT L. The optimizing considerations relevant here are a combination of those discussed in the DATI and DATO contexts.

To summarize, the delay in the execution of a microprogram containing a DATIO operation may be minimized if:

1. The microinstruction immediately preceding the Read microinstruction does not cause BUSY H or REPLY H to be asserted.

2. The addressed device responds to the assertion of BDIN L in minimum time.

3. The addressed device responds to the negation of BDINL in minimum time.

4. The addressed device responds to the assertion of BDOUT L in minimum time.

5. The addressed device responds to the negation of BDOUT L in minimum time.

6. The microinstruction immediately following the Output microinstruction does not check REPLY H or BUSY H.

As an alternative to condition 2 above, unavoidable delays may be utilized by inserting data manipulation microinstructions.

5.6 THE INTERRUPT OPERATION

Figures 5.6-1 and 5.6-2 illustrate the control signal action relevant to an interrupt operation. The microinstruction sequence presented is Read Acknowledge, Input Word, Next Microinstruction.

```
RA      B,A                     ; SIGNAL DIN, IACK
IW      B,A                     ; INPUT VECTOR LOW AND HIGH BYTE
Next Microinstruction
```

The individual events which occur in the interrupt transaction are discussed with the aid of Figures 5.6-1 and 5.6-2 which contain the Microcycle reference numbers.

Microcycle 1   Read Acknowledge

The execution of the Read Acknowledge microinstruction in the first microcycle causes no change in the control lines. The microinstruction executes to completion because REPLY H and BUSY H are unasserted during PH3.

Microcycle 2 Input Word (Wait)

Because the Read Acknowledge microinstruction executed to completion during Microcycle 1, the contents of the designated registers are placed on BDAL<15:00> at the beginning of PH1. Since no address information is required in the LSI-11 system bus interrupt transaction, any register(s) may be designated. WSYNC H and WIACK H are asserted at the beginning of PH2. The assertion of WIACK H holds the SYNC F/F in the reset state thus blocking the assertion of BSYNC L, as would otherwise occur at the beginning of PH4. BSYNC L is inhibited because it is not required in the LSI-11 system bus interrupt transaction.

The interrupting device only returns BRPLY L in response to BIACK H, which has not yet been asserted. Therefore, the wait state is initiated at the beginning of PH3.

Microcycle 3 Input Word (Wait)

Because the Read Acknowledge microinstruction executed to completion during Microcycle 1, WDIN H is asserted at the beginning of PH2. The assertion of BDINL follows immediately and is used to stablize the priorities in the interrupting device. Since REPLY H remains unasserted, the wait state is maintained.

Microcycle 4 Input Word (Wait)

With the assertion of WDIN H during Microcycle 3, the INT ACK F/F is no longer locked in the reset state therefore the INT ACK F/F is clocked to the set state on the trailing edge of PH1 thus asserting BIACK H. In response to the assertion of BIACK H, the interrupting device places its device vector on BDAL<15:00> and asserts BRPLY L. However, the REPLY F/F is not clocked until PH1. REPLY H remains passive which maintains the wait state.

Microcycle 5 Input Word (Input Low Byte)

The asserted state of BRPLY L sets the REPLY F/F at the beginning of PH1. Since REPLY H becomes active, the Wait state is terminated at the PH3 REPLY H check and the Data chip stores the low byte of the device vector in the designated register pair.

To assure minimum possible delay in the execution of the interrupt operation illustrated here, BRPLY L must be asserted at the processor in time to set the REPLY F/F at the beginning of Microcycle 5. Otherwise the wait state will continue throughout Microcycle 5 and the next opportunity to set the REPLY F/F will not occur until the beginning of Microcycle 6.

Microcycle 6 Input Word (Input High Byte)

Since the PH3 check of REPLY H is again passed, the Data chip stores the high byte of the device vector in a designated register (determined by complementing the low-order bit of the A register field). No control signals are changed by either the processor or the interrupting device.

Microcycle 7 Next Microinstruction (Possible wait)

Because the Input Word microinstruction executed to completion during Microcycle 6, both WSYNC H and WDIN H are made passive at the end of PH1. Since WDIN H must be asserted to enable REPLY H during an interrupt operation, REPLY H also goes passive. BDIN L goes passive immediately, which informs the interrupting device to remove its vector from BDAL<15:00>. The vector is removed about the same time BRPLY L goes passive at the processor.

If the Next Microinstruction belongs either to the Read or Write group, the state of BUSY H (not shown) is checked and a new WAIT state will be initiated which delays the beginning of the next I/O operation. Any other microinstruction will execute immediately.

Microcycle 8 Next Microinstruction

Since BRPLY L was negated by the interrupting device during Microcycle 7, the REPLY F/F is clocked to the reset state at the beginning of PH1, which cancels BUSY H (not shown). A WAIT state which may have been initiated during Microcycle 7 will be terminated at the PH3 check of BUSY H.

To assure minimum possible delay in the execution of the interrupt operation illustrated here, BRPLY L must go passive at the processor in time to reset the REPLY F/F at the beginning of Microcycle 8. Otherwise the next opportunity to reset the REPLY F/F will not occur until the beginning of Microcycle 9; and a possible wait state initiated during Microcycle 7 will continue at least into Microcycle 9.

FIGURE 5.6-1
INTERRUPT TRANSATION 1

FIGURE 5.6-2
INTERRUPT TRANSACTION 2

5.6.1 Interrupt Operation Microprogramming Summary

The interrupt operation, as implemented by the Read Acknowledge-Input microinstruction sequence, allows for variable delays in addressed device response to both the assertion and negation of BIACK H. BDIN L also takes part in the interrupt operation. The timing relationships between BIACK H and BDIN L is established by the LSI-11 system bus interface logic. The return of BRPLY L and the device vector is under control of BIACK H, not BDIN L, as is normally the case. Because WDIN H must be asserted before the INT ACK F/F can be set, the Write Acknowledge microinstruction does not assert BIACK H.

To summarize, the delay in the execution of a microprogram containing an interrupt operation may be minimized if:

1. The microinstruction immediately preceding the Read Acknowledge microinstruction does not cause REPLY H or BUSY H to be asserted.

2. The interrupting device responds to the assertion of BIACK H in minimum time.

3. The interrupting device responds to the negation of BIACK H in minimum time.

4. The microprogram immediately following the Input microinstruction does not check BUSY H or REPLY H.

As an alternative to 2 above, unavoidable delays may be utilized by inserting data manipulation microinstructions. However, due to the largely unpredictable nature of the interrupt operation, it is not expected that the freedom to perform data manipulation between execution of the Read Acknowledge and Input microinstructions would be of any general use.

CHAPTER 6

THE LSI-11 WRITEABLE CONTROL STORE

6.1 GENERAL

The KUV11-AA LSI-11 writeable Control Store is contained on a single 8.5x10 inch printed circuit board. The WCS module (M8018) contains the circuitry to support user microprogramming with the LSI-11 CPU.

Under normal circumstances the user need not be concerned with the hardware details of WCS module structure and operation. The micprogram trace facility is made available as a software tool under MODT.

The WCS module interfaces to the LSI-11 machine at two points; the LSI-11 system bus and the microinstruction bus. This configuration is illustrated in Figure 6.1.

LSI-11 SYSTEM BUS

LSI-11 PROCESSOR - WRITEABLE CONTROL STORE INTERCONNECTION

FIGURE 6.1

6.2 The Writeable Control Store Memory

The memory component of the Writeable Control Store module consists of 1024 24-bit memory words. This memory is implemented by high speed static semiconductor memory devices. Since the memory is static, no refreshing is required, but the the semiconductor memory is volatile, so the control store must be reloaded after each power up.

6.2.1 Control Store Microword Organization

All 24 bits of the microword are stored identically. The microword organization is determined by the functioning of individual bit fields and by the access timing relative to the micromachine cycle (microcycle). As shown in Figure 6.2, the 24 bit WCS microword is composed of the standard 22-bit microinstruction, MI<21:0>, and 2 additional bits, MI<23:22>, which function as extended TTL control bits.

6.2.1.1 Standard 22-Bit Microword MI<21:0> - The WCS memory provides storage for the standard 22-bit microwords. This microword is composed of 4 functionally distinct and independent fields, as illustrated in Figure 6.2. The three lower fields, MI<17>, MI<16>, MI<15:0> are delivered to the microprocessor for execution during the microfetch operation. The upper field, MI<21:18> is also accessed during a microfetch, but is delivered directly to the processor module. These 4 bits are decoded by the Special Control Function Logic on the LSI-11 CPU and provide TTL compatible control signals which are synchronized with the vertical microinstruction.

Regarding the standard 22-bit microword, the writeable control store memory is functionally identical to the MICROMS.

STANDARD 22-BIT MICROWORD PLUS EXTENDED CONTROL BITS

FIGURE 6.2

6.2.1.2 Extend TTL Control Bits MI<23:22> - The WCS memory provides storage for two additional control bits, called the Extended TTL control bits, that are not found in an LSI-11 MICROM. Both bits, MI<23:22>, appear at the WCS module fingers for user access via backplane connection. The highest bit, MI<23>, is used by the microaddress trace circuitry on the WCS module. These extended TTL control bits provide the user with additional control which is synchronized with the micromachine operations.

6.2.2 Writeable Control Store Page Organization

The 1024 Writeable Control Store microwords are organized as two 512 microword pages. The page organization provides flexibility in control addressing as well as a control store diagnostic capability. The page organization is illustrated in Figure 6.3.

WRITEABLE CONTROL STORE PAGE ORGANIZATION

FIGURE 6.3

6.2.2.1 Control Store Microaddressing Modes - The WCS module contains 8 DIP switches which determine the relationship of the WCS memory to the LSI-11 micromachine control store microaddress space. There are four address modes which are of general use and the switch positions for each mode are illustrated below:

	SW1	SW2	SW3	SW4	SW5	SW6	SW7	SW8
MODE 1	ON	OFF	OFF	ON	OFF	ON	OFF	-
MODE 2	OFF	ON	OFF	ON	OFF	OFF	ON	-
MODE 3	OFF	OFF	ON	ON	OFF	ON	OFF	-
MODE 4	ON	OFF	OFF	OFF	ON	ON	OFF	-

The address modes are explained as follows:

MODE 1 The WCS memory responds to microaddresses 2000-3777 (octal). This is the normal mode because MICROMS 0 and 1 contain the LSI-11 emulation and console ODT microcode and respond to microaddresses 0000-1777 (octal).

MODE 2 This is the Paging mode in which both 512 microword pages respond to control store addresses in the 2000-3777 (octal) range. The page from which the control store microword is ultimately accessed is determined by the WCS page logic. This logic is initialized by asserting the reset bit in the control/status register, CSR<15>. Once reset, all microaddresses point to WCS page 0. The WCS pages are swapped when the the page control F/F is toggled by placing a 07 (octal) code in MI<21:18>. The pages are swapped immediately after the microinstruction containing MI<21:18> = 07 is accessed in control store and subseuquent control store microwords are be accessed from PAGE 1. A second occurance of 07 MI<21:18> toggles the page control F/F back, and microwords are again fetched from PAGE 0.

MODE 3 The WCS memory responds to microaddresses 2000-3777 (octal). This is similar to MODE 1 except that WCS memory pages are permanently swapped. MODE 3, therefore, provides the diagnostic capability of executing a given microprogram out of different physical memory locations.

MODE 4 The WCS memory responds to microaddresses 0000-1777 (octal). This mode would not normally be used because MICROMS 0 and 1 respond to this same region.

In all of the modes described above, the WCS memory is loaded from the LSI-11 system bus via WCS RAM addresses 0000-1777 (octal). Figure 6.4 illustrates the relationship between the WCS RAM addresses and the responding microaddresses for each of the four modes. Note that page 0 is always accessed via the system bus in the WCS RAM address range 0000-0777 (octal) and similarly that Page 1 is always accessed in the

WCS RAM address range 1000-1777 (octal) range.

	WCS RAM <0000-0777>	WCS RAM <1000-1777>
MODE1	MIB<2000-2777>	MIB<3000-3777>
MODE2	MIB<3000-3777> (INITIAL PAGE)	MIB<3000-3777> (TOGGLED PAGE)
MODE3	MIB<2000-2777>	MIB<3000-3777>
MODE4	MIB<0000-0777>	MIB<1000-1777>

WCS RAM ADDRESS - MICROINSTRUCTION ADDRESS

FIGURE 6.4

6.3 WRITEABLE CONTROL STORE MEMORY ACCESS

The memory contained on the WCS module may be accessed in two ways: (1) by the microrocessor chip set via the microinstruction bus and (2) by the LSI-11 processor via the LSI-11 system bus (using normal PDP-11 instructions).

6.3.1 Microinstruction Bus Access

Figures 6.5-1 and 6.5-2 illustrate the LSI-11 micromachine configuration which contains both writeable Control Store and MICROMs for microprogram storage. The first hald of the illustration, Figure 6.5-1, appeared earilier in Chapter 3 as Figure 3.3 (Micromachine Inter-Chip Wiring Detail). The second half of the illustration, Figure 6.5-2, shows the interconnections to the WCS module. In addition to bus interconnections at both the machine and micromachine level, the WCS module also requires connections to 2 of the processor module clock phases, PH2 H and PH4 H. The wiring for these two connections is contained within the microinstruciton bus interconnect cable.

A Writeable Control Store memory access by the microinstruction bus is Read Only. There are no microinstructions which alter control store memory via the microinstruction bus.

WCS MICROINSTRUCTION BUS AND SYSTEM BUS INTERCONNECTION

FIGURE 6.5-1

WCS MICROINSTRUCTION BUS AND SYSTEM BUS INTERCONNECTION

FIGURE 6.5-2

The function of each of the 24 signal paths in the microinstruction bus interconnect cable is listed in Figure 6.6. MIB<10:00> carries the microaddress ,MADR<10:00>, to the WCS during the control store addressing phase of the microcycle. The WCS then responds, during the control store access phase, by asserting the microinstruction word on MIB<21:0>. Note that the MIB<10:0> lines are time-multiplexed for address and data information. For a 2 cycle microinstruction, during the first microcycle, a control store disable function (CSD) is performed with MIB<16>. If MIB<16> is asserted by the control chip during PH3, WCS response during the next microcycle is disabled. The highest 2 bits of the WCS microword, the Extended TTL control bits, do not appear in Figure 6.6 since they are not used by the LSI-11 CPU module and consequently are not carried by the MIB interconnection cable.

SIGNAL	ADDRESS CYCLE	ACCESS CYCLE
MIB<01>	MADR <1>	MI<01>
MIB<02>	MADR<02>	MI<02>
MIB<03>	MADR<03>	MI<03>
MIB<04	MADR<04	MI<04>
MIB<05>	MADR<05>	MI<05>
MIB<06>	MADR<06>	MI<06>
MIB<07>	MADR<07>	MI<07>
MIB<08	MADR<08	MI<08>
MIB<09>	MADR<09>	MI<09>
MIB<10>	MADR<10>	MI<10>
MIB<11>		MI<11>
MIB<12>		MI<12>
MIB<13>		MI<13>
MIB<14>		MI<14>
MIB<15>		MI<15>
MIB<16>	CSD <16>	MI<16>
MIB<17>		MI<17>
MIB<18>		MI<18>
MIB<19>		MI<19>
MIB<20>		MI<20>
MIB<21>		MI<21>
PH2 H		WCS TIMING WCS TIMING
PH4 H		

MICROINSTRUCTION BUS ACCESS FUNCTIONS

FIGURE 6.6

6.3.1.1 Control Store Access Timing - The control store access timing relevant to the WCS module is the same as the MICROM access timing. This timing appeared earlier as Figure 3.17 (MICROM ACCESS MICROINSTRUCTION BUS CYCLE) and is repeated here as Figure 6.7. As noted in the figure, the WCS module does not perform precharging of any microinstruction bus lines. This function is provided by every MICROM connected to the microinstruction bus. It is important also to note that active TTL control bits, MI<21:18>, remains asserted (active low) from the beginning of PH1 through the end of PH3 whereas the lower bits of the control store, MI<17:00> remain asserted during PH1 only.

During the first microcycle of a two microcycle operation, MIB<16> is asserted by the control chip to disable the control store response during the following microcycle. The relevant timing appeared earlier as Figure 3.18 (Microm Access Microinstruction Bus Cycle (Disabled)) and is repeated here as Figure 6.8. Again, the WCS module performs exactly as do the MICROMS, with the exception that only the MICROMS precharge tne MIB.

CONTROL STORE ACCESS MICROINSTRUCTION BUS CYCLE

FIGURE 6.7

Note: The MIB precharge function is only provided by MICROM control store.

6.3.1.2 Extended TTL Control Bit Timing - The timing diagrams illustrated in Figure 6.7 and 6.8 also show the timing relevant to the extended TTL control bits M1<23:22>. There are two operational characteristics which differentiate extended from standard control bit timing:

1. Both extended TTL control bits are latched for an entire microcycle, from the beginning of PH1 to the end of PH4.

2. Since the extended TTL control bits are not output to the microinstruction bus, they are not affected by control store disabling. Consequently, they appear at the system backplane during the microcycle immediately following the transmitted microaddress whether or not the WCS module was enabled.

The consequence of (1) above, is that an output signal can be produced that is maintained for multiple microcycles. For example, this would be achieved by assembling a 1 into successively executing MI<23> or MI<22> locations. The consequence of (2) above is that the extended TTL control bits may be asserted one microcycle, before a microinstruction that has MI<23> or MI<22> set to a "1". This occurs when a 2 cycle microinstruction immediately precedes (in WCS RAM) the microinstruction that has MI<23> or MI<22> set to a "1" (whether or not that microinstruction is executed) since during the second (or disabled) cycle the WCS still accesses the next sequential microinstruction.

6.3.2 LSI-11 System Bus Access

The Writeable Control Store RAM memory access by the LSI-11 system bus is Read/Write. The WCS module system bus interface logic supports only Programmed I/O data transfers via the system bus interface registers. The configuration and operation of these interface registers are discussed in section 6.5.

6.4 THE MICROADDRESS TRACE FACILITY

The writeable control store option has a 16-word microaddress trace facility which is an aid to microprogram debugging. The trace hardware is normally controlled by the operator under MODT.

6.4.1 Microaddress Trace Buffer Operation

The hardware portion of the microaddress trace facility consists of a 16-word recirculating buffer which is continuously loaded with the last microaddress asserted on the microinstruction bus by the microprocessor control chip. The buffer therefore contains the last 16 microaddresses presented to the Control store (either the WCS or MICROM including the disabled cycle of a 2 cycle microinstruction). When it is desired to freeze a microaddress trace for examination, the operator, via MODT, sets MI<23> (one of the extended TTL bits) of the microword that is to be the LAST microinstruction in trace buffer. When this microinstruction is fetched by the microprocessor Control chip, the WCS hardware inhibits the microaddress trace buffer clock. Once the clock is inhibited, the trace buffer contents remain unchanged. The operator may then access the buffer to read the traced microaddresses in the order of the actual execution. (See Section 6.5.3).

The trace buffer is implemented with three 16x4-bit random access memories addressed by a 4-bit counter. The counter is normally clocked continuously, thus providing a recirculating, increasing address sequence (module 16) to the buffer memories. When the microprogram trace has been halted, each of the stored microaddresses can be accessed (See Section 6.5.3.1).

The format of a trace buffer word is shown in figure 6.9.

6.4.2 WCS ENABLE BIT

In addition to providing a readout of the last 16 microaddresses on the MIB, the trace buffer also indicates whether the WCS responded to the microaddress. The buffer stores an extra bit (bit <11>), the WCS Enable Bit, along with each microaddress. This bit is cleared to 0 when the WCS was disabled during the microfetch and set to 1 when the WCS was enabled. This is useful in determining microprogram execution delays incurred during data access (system bus I/O) operations.

MICROADDRESS TRACE BUFFER WORD FORMAT

FIGURE 6.9

6.5 LSI-11 SYSTEM BUS INTERFACE

The WCS interface to the LSI-11 bus is similar to that of other I/O devices which support only Programmed I/O transactions. The device address format is a modification of the general format presented earlier in section 2.2.3.1. The primary features of the WCS inteface registers are that there is a single control/status register and that two other registers are required to provide the 24-bit access path for the WCS microword. In addition, one register is multiplexed to provide access to the microaddress trace buffer. All WCS interface register formats are illustrated in Figure 6.10. The register addresses on the LSI-11 system bus are 177540, 177542 and 177544.

WCS INTERFACE REGISTER BIT ASSIGNMENTS

FIGURE 6.10

6.5.1 WCS Control/Status Register

The specific functions of the WCS control/status register are explained according to the related bit-fields.

CSR <9;Ø> Read/Write

When CSR<12> is a "Ø" (WCS module is not enabled) this field is the 1Ø-bit WCS RAM address and is Read/Write (CSR <Ø> is the low-order bit and CSR<9> is the high-order bit).

When CSR<12> is a "1" (WCS module is enabled), this field is Read only and contains the last microaddress prior to the actual bus cycle of the PDP-11 instruction that is used to access the control/status register. Normal usage of this field is the former case.

CSR<11;1Ø> Read Only

These bits are unused and always read as "Ø".

CSR<12> Read/Write

This bit functions as the Writeable Control Store Enable. After it is set to "Ø", the WCS module makes no response to the microinstruction bus, and the WCS RAM can then be accessed by the LSI-11 bus. When it is set to a "1", the WCS module responds to microaddresses in its switch-selected range, and the WCS RAM cannot be altered by the LSI-11 bus.

CSR<13> Read/Write

This bit functions as the Examine Toggle for the trace buffer and is toggled to examine sequential trace words. Its use is further explained in Section 6.5.3.1.

CSR<14> Read/Write

This bit functions as the Trace Mode Examine. It should be set to Ø for normal control store operation. When it is set to 1, the interface register located at 177542 may be read to access the trace word, as shown in Figure 6.11. Note that the trace word appears as Read Only information (see Section 6.5.3.1).

CSR<15> Read/Write

This bit functions as the WCS Reset. When set to a "1", the control store memory paging logic is reset to PAGE Ø and the Trace Buffer address is set to a "Ø". Subsequently clearing this bit to a "Ø" enables the paging logic and the counter that supplies the Trace Buffer address.

6.5.2 WCS Memory Access Registers

When CSR<12> is a "0", the WCS module is not enabled to respond to the MIB. The WCS module device regsters, which function as WCS RAM data access registers, are 177542 and 177544, as shown in Figure 6.10. The lower 16 bits of the microword, MI<15:0>, are accessed via location 177542. The higher 8 bits, MI<23:16> are accessed via location 177544. Note that bits <15:8> of location 1/7544 are unused and are read as "0". The address of the WCS RAM microword accessed via these registers is determined by CSR<9:0>.

Both WCS memory access registers support Read/Write access only when the WCS is disabled (CSR<12>=0). The Write mode, implemented by a DATO operation, alows WCS memory to be loaded. The Read mode, implemented by a DATI operation allows the WCS memory contents to be examined. Because of the 24 bit length a single microword must be either written or read as two operations.

When the WCS module is enabled (CSR<12>=1), both WCS register 177542 and 177544 are READ only, and the data which is read will depend on the type instruction (or console ODT) that is used to access the register. This is due to the fact that, when enabled, the WCS module is being addressed only by the MIB, and the microaddress present on the MIB determines the WCS RAM location that is accessed.

Proper disabling of the WCS module in preparation for system bus access operations proceeds as follows:

STEP 1 DISABLE WCS:
(177540) <- 000000

STEP 2 WRITE DESIRED ADDRESS (WITH CSR<15>=0) INTO THE CSR MEMORY ADDRESS REGISTER
(CSR<9:0>); (177540) <- 000777

Following step 2, read or write operations may be performed at LSI-11 locations 177542 and 177544 to access both parts of the stored microword.

6.5.3 Microaddress Trace Register

When bit <14> of the control/status register is set to 1, location 177542 no longer functions as a memory access register, but as a trace buffer access register, as shown in Figure 6.10. The specifications of this register are explained according to the related bit fields:

MIR <10:0>

This bit field contains an 11-bit microaddress MIR<00> being the low-order microaddress bit and MTR<10> being the high order microaddress bit. The value read from MTR<10:00> is the value that appeared on the microinstruction bus while the microprogram was being traced.

MTR<11>

This bit is the value of the WCS Enable Bit which is stored in the trace buffer along with each microaddress. A value of 0 indicates that WCS response was disabled for the microcycle.

MTR<12:15>

This bit field contains the 4-bit (modulo 16) address of the trace buffer memory. The counter value is made available in the trace word to aid in dumping the buffer contents in the proper order of occurrence.

6.5.3.1 Microaddress Trace Buffer Dump Algorithm - Proper operation of the microaddress trace buffer hardware requires the following steps:

1. The trace buffer is initialized by asserting CSR<15>.

2. The last microinstruction to be traced contains a value of 1 in MI<23>.

After the trace buffer is initialized, the trace buffer is continually updated to contain the 16 most recent microaddresses placed on the microinstruction bus by the microprocessor Control chip. The buffer contents are frozen when the microinstruction containing MI<23>=1 is accessed. The buffer contents may be subsequently dumped by placing the WCS module in Examine Trace mode and then subsequently accessing the trace buffer contents. The algorithm used is illustrated in Figure 6.11. Note that the buffer memory address must be saved before the toggle/microaddress store loop is entered to provide a reference for any subsequent tests. Also note that every CSR write operation must contain a 1 in bit <14> to maintain the Trace Examine Enable. The final 2 events in the flow chart reset the Page Logic and the Trace Buffer and re-enable the WCS memory response.

MICROADDRESS TRACE BUFFER DUMP ALGORITHM

FIGURE 6.11

6.6 WRITEABLE CONTROL STORE MODULE CIRCUIT DESCRIPTION

The basic functional blocks of the WCS module are illustrated in Figure 6.12. This circuit description is intended to aid understanding of the schematic diagrams supplied with the WCS module.

6.6.1 Clock Generation

The clock generation circuit receives as input 2 of the 4 TTL microcycle clock phases (PH2 H, and PH4 H) from the LSI-11 processor module. These clock signals are connected to the WCS module via the microinstruciton bus interconnect cable. Note that only processor modules of etch revision F (CS Rev Y) or later have the necessary clock signals available at the empty MICROM Socket, E32 (normally used for the KEV-11 option).

The clock generation circuit then deriveadditional 2 separate signals from its inputs, namely, PH1 H and PH23 H. PH23 H is high continuously during microcycle phases 2 and 3 and is used to enable the standard TTL control bits. PH1 H enables the output of MI<17:0>.

WCS MODULE FUNCTIONAL BLOCK DIAGRAM

FIGURE 6.12

6.6.2 Control Store Memory and Microaddress Multiplexer

The control store memory is implemented with 24 random access memory integrated circuits each having a 1-bit by 1024 organization. The control store page organization is established by the microaddress multiplexer in conjunction with the paging logic. Note that the microinstruction bus input to the microaddress multiplexer is 11 bits wide (1-of-2048) and the multiplexer output is 10 bits wide (1-of-1024).

6.6.3 LSI-11 System Bus Interface

The system bus inteface is implemented with integrated bus transceivers (DC005) and a protocol logic circuit (DC004). The read portion of the 4 possible register formats is determined by the read back multiplexer. The extended TTL control bits are available on the WCS module rear fingers at locations which are normally Spare. Note that bus address 177546 is blocked (since this is the address assigned to the KWV-11 option).

6.6.4 Microinstruction Bus Interface

The microinstruction bus interface is implemented with special integrated circuits which interface the MOS logic levels to the TTL levels on the WCS module. Only 12 receivers are implemented (MIB<10:0> and MIB<16>), and since 22 bits of the stored microword are returned via the microinstruction bus, 22 drivers are implemented. The phases during which the control store output is asserted are determined by the output enable circuitry. Control store output is enabled only when:

1. An appropriate microaddress has been received.

2. The WCS module is enabled (CSR<12>=1).

3. And when the control store disable bit (MIB<16>) was not asserted by the Control Chip on the previous cycle (PH3).

6.6.5 Microaddress Trace Buffer and Counter

The microaddress trace facility is implemented by three 4x16 bit random access memories, one 4-bit counter which address the memories, and associated clearing and clocking logic. Assertion of CSR <15> clears the counter and restores the clock input (PH2) to the counter. The 16 microaddress buffer location produced by this configuration are loaded from the output of the microaddress multiplexer, MADR<9:0> bit <11> is taken directly from MIB<11>]. In this way the stored microaddress reflects the control store address placed on the MIB by the microprocessor Control chip. The buffer contents are examined from the system bus interface via the read back multiplexer.

6.7 WRITEABLE CONTROL STORE HARDWARE SPECIFICATIONS

The items contained in this section provide a quick reference for module hardware details.

6.7.1 Dimensions

The KUV11-AA LSI-11 Writeable Control Store option consists of (1) a standard quad height, 8.5 by 10 inch-multilayer printed circuit board (M8018) with signal etch on both sides and 2 inner layers (VCC and GND) and (2) a Microinstruction Bus Interconnect cable.

6.7.2 Power Requirements

The only power supply voltage required by the WCS module is +5 volts, Connection to the +5 volt supply as well as ground return is established through the module finger/backplane interconnection, The supply voltage tolerance is ?15% and the current drawn from the +5 volt supply is 3.A typical (7.34A worst case).

6.7.3 LSI-11 System Bus Backplane Pin Assignment

The figure below lists the WCS module (M8018) backplane pin assignments:

AE1	MI<22> (SROM 4 H)
AF1	MI<23> (SROM 5 H)
AJ1	GND
AM1	GND
AT1	GND
AA2	+5V
AC2	GND
AE2	BDOUT L
AF2	BRPLY L
AH2	BDIN L
AJ2	BSYNC L
AK2	BWTBT L
AM2	BIAKI L (JUMPERED TO AN2)
AN2	BIAKO L
AP2	BBS7 L
AR2	BDMGI L (JUMPERED TO AS2)
AS2	BDMGO L
AU2	BDAL00 L
AV2	BDAL01 L
BA1	BDCOK H
BJ1	GND
BM1	GND
BT1	GND
BA2	+5V
BE2	BDAL02 L
BF2	BDAL03 L
BH2	BDAL04 L
BJ2	BDAL05 L
BK2	BDAL06 L
BL2	BDAL07 L
BM2	BDAL08 L
BN2	BDAL09 L
BP2	BDAL10 L
BR2	BDAL11 L
BS2	BDAL12 L
BT2	BDAL13 L
BU2	BDAL14 L
BV2	BDAL15 L
CJ1	GND
CM1	GND
CT1	GND
CA2	+5V
CC2	+5V
CM2	BIAKI L (JUMPERED TO CN2)
CN2	BIAKO L
CR2	BDMGI L (JUMPERED TO CS2)
CS2	BDMGO L
DJ1	GND
DM1	GND
DT1	GND
DA2	+5V
DC2	GND

LSI-11 SYSTEM BUS BACKPLANE PIN ASSIGNMENT

FIGURE 6.13

6.7.4 Microinstruction Bus Connector Pin Assignment

Figure 6.14 contains a table which lists the pin assignments for the microinstruction bus interconnect cable. All unlisted numbers have no connection at either end of the cable assembly. The pin assignments are the same at both the processor module end and the WCS module end of the cable. However, because of two series matching registers (pins 24,25) on the CPU end of the cable, (inside the cable assembly), the cable plugs are NOT interchangeable and the processor module end carries a special designation. A continuity check of each signal path will produce a low resistence reading (less than 1 ohm) except for pins 24 and 25. These paths carry the microcycle clock phases and contain a 100 OHM series resistence on each path.

CABLE PLUS PIN NUMBER (BOTH ENDS)	FUNCTION
7	MIB<15>
8	MIB<14>
9	MIB<13>
10	MIB<12>
11	MIB<16>
12	MIB<17>
13	MIB<18>
14	MIB<19>
15	MIB<20>
16	MIB<21>
19	GND
24	PH4 H (100 OHM SERIES RESISTANCE)
25	PH2 H (100 OHM SERIES RESISTANCE)
26	MIB<11>
27	MIB<10>
28	MIB<9>
29	MIB<8>
30	MIB<7>
31	MIB<6>
32	MIB<5>
33	MIB<4>
34	MIB<3>
35	MIB<2>
37	MIB<1>
38	MIB<0>

MICROINSTRUCTION BUS INTERCONNECT CABLE PIN ASSIGNMENT

FIGURE 6.14

6.7.5 TTL Control Bit Summary

The TTL control bits which may be used by the microprogrammer to issue control pulses which are synchronized with micromachine operation are stored in the top 6 bits of the control store microword, MI<23:18>. The standard TTL control bits, MI<21:18> directly drive the special control function logic on the LSI-11 processor module. The bit field values which are not decoded by this logic may be employed by the microprogrammer/interface designer to implement synchronized functions. The extended TTL control bits, MI<23:22>, are a feature of the WCS Module. Figure 6.15 lists the backplane pin assignments of all TTL control bits; only PH3 of the microcycle is available at the system backplane which may be used to enable synchronized decoding of user-defined control states. The following timing details should be recalled from sections 5.3.1.1 and 5.3.1.2.

1. MI<21:8> are valid from the beginning of PH1 to the end of PH3.

2. MI<23:22> are valid throughout the microcycle and may be used to create a signal which is asserted continuously over multiple microcycles.

3. Only MI<21:8> are subject to control store disabling and MI<23:22> appear regardless of the disable signal (MIB<16>).

It should further be recalled from section 5.5 that MI<23> is shared with the microaddress trace buffer functions. However, if the trace feature is not utilized, MI<23> can be used in the same fashion as MI<22>.

Figure 6.16 contains a table which lists the special control functions which are decoded by the LSI-11 processor logic. Figure 6.15 also indicates that octal code 07 is used to swap WCS memory pages. Swapping is effected only when MODE 2 control store memory addressing has been selected via the WCS module option switches (section 5.2.2).

MODULE	SIGNAL	DEFINITION	PIN
PROCESSOR	SROM Ø H	MI<18>	DD1
PROCESSOR	SROM 1 H	MI<19>	DE1
PROCESSOR	SROM 2 H	MI<2Ø>	DF1
PROCESSOR	SROM 3 H	MI<21>	DH1
WCS	SROM 4 H	MI<22>	AE1
WCS	SROM 5 H	MI<23>	AF2
PROCESSOR	SPH3 H	MICROCYCLE PHASE 3	DC1

TTL CONTROL PIN BACKPLANE PIN ASSIGNMENT

FIGURE 6.15

BINARY	OCTAL	FUNCTION
1000	10	RESERVED
1001	11	IFCLR+ SRUN L
1010	12	TFCLR L
1011	13	RFSET L
1100	14	INITIALIZE SET
1101	15	FAST DIN
1110	16	PFCLR L
1111	17	EFCLRL
		#
0000	00	Available
0001	01	Available
0010	02	Available
0011	03	Available
0100	04	Available
0101	05	Available
0110	06	Available

STANDARD TTL CONTROL BIT FUNCTIONS

FIGURE 6.16

CHAPTER 7

LSI-11 MICROPROCESSOR SET (MPS)

7.1 GENERAL

A set of MACRO-11 macros has been defined to assemble LSI-11 microcode using the MACRO-11 assembler. This manual describes the symbolic format and special features contained in the macro parameter file.

Each 22 -bit WD microword is assembled into 2 PDP 11 16 -bit words starting at PDP-11 word locations 1000 ,1002 (octal). For example:

WD ADDRESS:	PDP-11 ADDRESS:
0	1000 ,1002
2000	11000 ,11002
3000	15000 ,15002

7.2 SYMBOLIC INSTRUCTION FORMAT

The basic instruction line format is:

label: opcode operands, translation, state ; comment

The label, translation, state, and comment are optional. The format of the operand field depends on the opcode. Since the instructions fall into a number of distinct format classes, each class will be dealt with as a group.

Each microword in the ROM holds an additional 6 bits. One bit (LRR) controls loading of the return register with the incremented LC; four bits specify a "state code" for use by the chip set environment (e.g., - LSI11 BUS interface), and one bit (RSVC) invokes a translation.

This state field can be specified for all instructions. If omitted, the field will be assembled with a value of 0. The following predefined symbols are available for use in this field and may be or'd (!) or added (+) together as necessary.

MNEMONIC	OCTAL VALUE (BITS 21:16)	MEANING
RSVC (MIB17)	2	Service Translation (called RNI in WD spec)
LRR (MIB16)	1	Load Return Register
SØ (MIB2Ø:18)	Ø	State code Ø
S1 (MIB2Ø:18)	4	State code 1
S2 (MIB2Ø:18)	1Ø	State code 2
S3 (MIB2Ø:18)	14	State code 3
S4 (MIB2Ø:18)	2Ø	State code 4
S5 (MIB2Ø:18)	24	State code 5
S6 (MIB2Ø:18)	3Ø	State code 6
S7 (MIB2Ø:18)	34	State code 7
CØ (MIB21)	4Ø	LSI11 BUS CØ control signal

Furthermore, a ROM address may have a translation associated with it, While the assembler will not "assemble" the translation information into the PTA, it does allow specification of a symbolic translation identifier at a particular microlocation for documentation purposes.

For all instructions involving registers, the following predefined symbols are available for use in the register field(s):

MNEMONIC	VALUE	MEANING FOR A PDP11 MACHINE
RBA	2	bus address
RBAL	2	bus address lower byte
RBAH	3	bus address upper byte
RSRC	4	source operand
RSRCL	4	source operand lower byte
RSRCH	5	source operand upper byte
RDST	6	destination operand
RDSTL	6	destination operand lower byte
RDSTH	7	destination operand upper byte
RIR	10	instruction register
RIRL	10	instruction register lower byte
RIRH	11	instruction register upper byte
RPSW	12	program status word
RPSWL	12	program status lower byte
RPSWH	13	program status upper byte
SP	14	stack pointer
SPL	14	stack pointer lower byte
SPH	15	stack pointer upper byte
PC	16	program counter
PCL	16	program counter lower byte
PCH	17	program counter upper byte
G	0	indirect through G register
GL	0	lower byte indirect through G
GH	1	upper byte indirect through G

7.3 OPCODE FORMATS

In the source formats explained in this section, a "t" is used to represent the translation symbol and an "x" is used to represent the state code field.

7.4 JUMP Instruction

Format:

```
        label:   JMP address, t,x    ; comment
```

3.2 Conditional JUMP Instructions

Format:

```
      label:     opcode  address, t,x     ; comment
```

Opcodes:	jump if	jump condition code
JZBF	zb=0	0
JZBT	zb=1	1
JC8F	c8=0	2
JC8T	c8=1	3
JIF	ics=0	4
JIT	ics=1	5
JNBF	nb=0	6
JNBT	nb=1	7
JZF	z=0	10
JZT	z=1	11
JCF	c=0	12
JCT	c=1	13
JVF	v=0	14
JVT	v=1	15
JNF	n=0	16

JNT	n=1	17

7.4.1 Literal Instructions

Format:

```
        label:     opcode  literal, register, t,x ;comment
```

Opcodes:

MNEMONIC	OPCODE VALUE	MEANING
AL	2	add literal
CL	3	compare literal
NL	4	and literal
TL	5	test literal
LL	6	load literal

7.4.2 Two Register Instructions

Format:

```
        label:   opcode bregister, aregister,t,x ;comment
```

When it is desired to affect the macro level flags (ie., - C, V, Z, N), an "F" is appended to the opcode mnemonic (of course, only for those opcodes that have the capability of affecting the macro flags)

Opcodes:

MNEMONIC	OPCODE VALUE	MEANING
MB	200	move byte
MBF	201	move byte and set flags
MW	202	move word
MWF	203	move word and set flags
CMB	204	conditionally move byte
CMBF	205	conditionally move byte & set flags
CMW	206	conditionally move word
CMWF	207	
TCB	230	two's complement byte
TCBF	231	
TCW	232	two's complement word
TCWF	233	
OCB	234	one's complement byte
OCBF	235	
OCW	236	one's complement word
OCWF	237	

ICB1	220	increment by 1
ICB1F	221	
ICW1	222	
ICW1F	223	
ICB2	224	increment by 2
ICB2F	225	
ICW2	226	
ICW2F	227	
DB1	274	decrement by 1
DB1F	275	
DW1	276	
DW1F	277	
AB	240	add
ABF	241	
AW	242	
AWF	243	
CAB	244	conditionally add
CABF	245	
CAW	246	
CAWF	247	
ABC	250	add with carry
ABCF	251	
AWC	252	
AWCF	253	
CAI	256	conditionally add word on ICS
CAIF	257	
SB	260	subtract
SBF	261	
SW	262	
SWF	263	
SBC	270	subtract with carry
SBCF	271	
SWC	272	
SWCF	273	
CB	264	compare
CBF	265	
CW	266	
CWF	267	
ORB	310	inclusive or
ORBF	311	
ORW	312	
ORWF	313	
XB	314	exclusive or
XBF	315	
XW	316	
XWF	317	
NB	300	and
NBF	301	
NW	302	
NWF	303	
TB	304	test
TBF	305	
TW	306	

TWF	307	
SLB	214	shift left
SLBF	215	
SLW	216	
SLWF	217	
SLBC	210	shift left with carry
SLBCF	211	
SLWC	212	
SLWCF	213	
SRB	334	shift right
SRBF	335	
SRW	336	
SRWF	337	
SRBC	330	shift right with carry
SRBCF	331	
SRWC	332	
SRWCF	333	
NCB	320	and complement
NCBF	321	
NCW	322	
NCWF	323	
MI	354	modify instruction
LTR	356	load translation register
CAD	254	conditionally add digits
RIB1	360	read & increment byte by 1
WIB1	361	write & increment byte by 1
RIW1	362	read & increment word by 1
WIW1	363	write & increment word by 1
RIB2	364	read & increment byte by 2
WIB2	365	write & increment byte by 2
RIW2	366	read & increment word by 2
WIW2	367	write & increment word by 2
R	370	read
W	371	write
RA	372	read acknowledge
WA	373	write acknowledge
OB	374	output byte
OW	375	output word
OS	376	output status

7.4.3 Isingle Register Instructions

Format:

```
        label:   opcode  aregister,t,x  ;comment
```

Opcodes:

MNEMONIC	OPCODE VALUE	MEANING
CCF	162	Copy condition flags
LGL	165	Load g register low
CIB	166	Conditionally increment byte
CDB	167	Conditionally decrement byte

The following symbols have been added for referring to the status bits read by the CCF instruction:

MNEMONIC	VALUE	MEANING
C8	20	Carry into bit 8
C4	40	Carry into bit 4
Z8	100	Zero
N8	200	Negative

These symbols may be or'd (!) or added (+) together as necessary.

7.4.4 Other Instructions

7.4.4.1 Reset and Set Flags Instructions (RF and SF) -

Format:

```
        label:   opcode   flags,t,x   ; comment
```

Opcodes:

MNEMONIC	OPCODE VALUE	MEANING
RF(RI)	160	reset (clear) flags
SF(SI)	161	set flags

The following predefined symbols are available for use in the flags field:

MNEMONIC	VALUE	MEANING
I4	1	internal interrupt flag 4
I5	2	internal interrupt flag 5
I6	4	internal interrupt flag 6

These symbols may be or'd (!) or added (+) together as neccessay.

7.4.4.2 Input Instructions -
Format:

```
        label:   opcode  accesscode, register,t,x      ; comment
```

Opcodes:

MNEMONIC	OPCODE VALUE	MEANING
IB	340	Input byte
IBF	341	Input byte and set flags
IW	342	Input word
IWF	343	Input word and set flags
ISB	344	Input status byte
ISBF	345	Input status byte and set flags
ISW	346	Input status word
ISWF	347	Input status word and set flags

The following predefined symbols are available for use in the access code field:

MNEMONIC	ACCESS CODE VALUE	MEANING
UB	0	upper byte
UBC	1	upper byte conditionally
LB	2	lower byte
LBC	3	lower byte conditionally
RMW	4	read/modify/write
TG6(only IW)	1	load TR from DAL 15:0>, load G from DAL6:4>
TG8(only IW)	2	load TR from DAL15:0>, load G from DAL 8:6>

These symbols may be or'd (!) or added (+) together as necessary.

7.4.4.3 No-Operation Instruction (NOP) -
Format:

```
        label:   NOP   t,x   ; comment
```

7.4.5 Load Condition Flags Instruction (LCF)

Format:

```
        label:   LCF flagenables, register,t,x      ; comment
```

The following predefined symbols are available for use in the flag enable field:

mnemonic	value	meaning
C	1	carry
V	2	overflow
Z	4	zero
N	10	negative

These symbols may be or'd (!) or added (+) together as necessary.
h13 Return From Subroutine Instruction (RFS)
Format:

```
        label:    RFS    t,x     ; comment
```

(Note that an RFS instruction assembles as a JMP 4000)

7.4.5.1 Reset TSR Instruction (RTSR) -

Format:

```
        label:    RTSR     t,x      ;comment
```

7.5 SPECIAL DIRECTIVES SUMMARY

In addition to the normal MACRO-11 directives, several directives have been made available via the RT-11 macro file to make programming easier. They are as follows:

7.6 NXT Directive

The NXT (next) directive is used as an aid in placing or locating code branched to by a translation. Eight directives are available

```
     NXT2
     NXT4
     NXT10
     NXT20
     NXT40
     NXT100
     NXT200
     NXT400
```

which advance the LC to the next address evenly divisible by the specified power of two. If the current LC satisfies that condition, it is unchanged.

7.6.1 Normal MACRO-11 Directives

All features of MACRO-11 are available to the programmer and other MACRO-11 directives that have meaning can be used. An .ASECT Directive is done in the microassembler and the microlocation counter is set to 0. To change the microlocation counter in an assemply, a directive of the form: .=N can be used, where N is the new location desired.

7.6.2 LOC Directive

The LOC(Micro location) directive can also be used to establish a particular micro address in a micro assembly. The form of the directive is:

```
        LOC N
```

where N is the desired Micro address. This directive was included for consistancy with the System 10 microassembler (U11 .MAC).

7.7 SPECIAL ERROR MESSAGES

In addition to the normal error checking done by MACRO-11, several error checking features have been added via the parameter file.

7.7.1 Off-page Conditional Jump References

Since the LSI-11 microprocessor allows conditional jumps only within a fixed (not relative) 256 decimal (400 octal) word page, the assembler will flag violations at assembly time with a "P" (questionable syntax) error. MACRO-11 also uses the error symbol "P" as a catch-all error message.

An idiosyncrasy of the microprocessor occurs when a conditional jump is assembled at the last location of a 256 decimal word page. Instead of remaining in the existing fixed page, jumps can only be made to the next successive page. For example, if a conditional jump is at location 377 octal, the jump can only reference locations 400 through 777 octal. The microassembler will also flag these violations in the same fashion as above.

7.7.2 Overwritten ROM Locations

Because of the nature of the beast, the microprogrammer must live with address constraints. Thus when an attempt is made to assemble a ROM location that was already assembled,the assembler will not flag the occurrence.

7.7.3 Oversized Literals

Any literal that exceeds the literal width of 8 bits will also be flagged with a "P" error.

7.8 OPERATING PROCEDURES and EXAMPLES

A macro parameter file, UCASM.MAC, must be used along with your source file to assemble WD microcode. The source code should be a garden variety MACRO-11 source file which can be created by using any system compatable editor (such as EDIT).

To assemble WD microcode on the PDP-11 , run Macro as follows:

```
     R MACRO
```

and when MACRO responds with a "*", type the command string

```
     FILNAM,FILNAM=UCASM,FILNAM
```

where FILNAM is the name of your source file (if extension is .MAC, it can be eliminated). The output will be a .OBJ (object module) file and a .LST listing file with the filenames the same as the source file. when MACRO is finished, type "Control C" to get back to monitor level. Then the listing can be printed by using the PIP utility program

A useful aid during program development is a cross reference listing which is appended to the normal assembler listing. To obtain one, type the following command to MACRO:

```
     FILNAM,FILNAM/C=UCASM,FILNAM
```

An object module that is produced by MACRO must then be passed through the RT-11 Linker program to produce the .SAV file on the system device (or any device for that matter) that is required by the WCS LOADER.

The following example illustrates a session at a terminal: (the source file has already been created)

```
     .R MACRO
```

```
        *EXAMPL,EXAMPL/C=UCASM,EXAMPL
        ERRORS DETECTED:  Ø
        FREE CORE:  7745.  WORDS

        *^C

        .R PIP
        *LP:=EXAMPL.LST
        *^C

        .R LINK
        *EXAMPL=EXAMPL

        *^C
```

A SAMPLE PROGRAM IS SHOWN BELOW THAT ILLUSTRATES A TYPICAL SOURCE FORMAT. THE SYMBOL TABLE IS NOT SHOWN.

LSI-11 MICROPROCESSOR SET (MPS)

```
                                .TITLE PUSHR AND POPR INSTRUCTIONS
                      ;         THIS IS THE MICROCODE FOR THE PUSHR AND POPR INSTRUCTI
                      ;         PUSHR=Ø767ØØ
                      ;         POPR=Ø767Ø1
                      ;
                      ;         SINCE WE DON'T HAVE AUTO POWERUP OR MICRO INTERRUPTS:
         ØØØØØØ                 POWRUP=Ø
         ØØØØØ3                 INTSER=3
                      ;FIRST DO TRANSLATION BRANCHES AND THE DECODE OF Ø76XXX OPCODE
   ØØØØØ                        LOC         3ØØØ
                                                                              12X
   Ø3ØØØ                        JMP         Ø                     ;TRAP ILLEGAL OPCODES
   Ø3ØØ1  * NORMAL ENTRY        JMP         DECODE                ;COME HERE FOR OPCODES
   Ø3ØØ2                        JMP         POWRUP                ;POWER UP ENTRY IF NEE
   Ø3ØØ3                        JMP         Ø        75040-75777  ;TRAP ILLEGAL OPCODES
   Ø3ØØ4              INTABI:   JMP         INTSER                ;COME HERE FOR MICRO I
                      ;                                                         to 076777
   Ø3ØØ5              DECODE:   CL          175,RIRL              ;ZB IF OPCODE IS Ø764Ø
   Ø3ØØ6                        JZBT        DECON                 ;JUMP IF TRUE
   Ø3ØØ7              DECTRP:   JMP         Ø                     ;TRAP ILLEGAL CODE
   Ø3Ø1Ø              DECON:    CL          3Ø1,RIRH              ;ZB IF OPCODE IS Ø767Ø
   Ø3Ø11                        JZBT        POPR                  ;JUMP IF TRUE
   Ø3Ø12                        CL          3ØØ,RIRH              ;ZB IF OPCODE IS Ø767Ø
   Ø3Ø13                        JZBF        DECTRP                ;JUMP IF FALSE
                      ;            this is byte swapped due to PLA
                      ;THIS IS THE ALGORITHM FOR THE PUSHR INSTRUCTION:
   Ø3Ø14              PUSHR:    LL          5,RIRL                ;RIRL=FIRST G CONSTANT
   Ø3Ø15              PUSHLP:   LGL         RIRL                  ;SET UP G REGISTER
   Ø3Ø16                        AL          376,SPL               ;DECREMENT STACK BY 2
   Ø3Ø17                        CDB         SPH
   Ø3Ø2Ø                        W           SPH,SPL               ;PUSH REGISTER ONTO ST
   Ø3Ø21                        DB1         RIRL,RIRL             ;DECREMENT G CONSTANT
   Ø3Ø22                        OW          GH,GL
   Ø3Ø23                        JNBF        PUSHLP                ;JUMP IF NOT DONE
   Ø3Ø24                        JMP         1Ø34                  ;JUMP TO DO FLAGS AND
                      ;NOW HERE IS THE POPR ALGORITHM:
   Ø3Ø25              POPR:     LL          Ø,RIRL                ;RIRL=FIRST G CONSTANT
   Ø3Ø26              POPLP:    LGL         RIRL                  ;SET UP G REGISTER
   Ø3Ø27                        RIW2        SPH,SPL               ;INPUT REGISTER AND IN
   Ø3Ø3Ø                        ICB1        RIRL,RIRL             ;INCREMENT G CONSTANT
   Ø3Ø31                        IW          ,G
   Ø3Ø32                        CL          6,RIRL                ;ZB IF DONE
   Ø3Ø33                        JZBF        POPLP                 ;JUMP IF NOT DONE
   Ø3Ø34                        JMP         1Ø34                  ;JUMP TO DO FLAGS AND
                      ;
                      ;
                      ;IN THE MAIN CODE THE ENDING CODE USED IS AS FOLLOWS:
                      ;         LOC         1Ø34
                      ;TSTØ:    SBF         RIRL,RIRL,,RSVC       ;CLEAR C FLAG
                      ;         MWF         G,G                   ;EFFECT FLAGS
                      ;
```

```
53 03035                    END
```

7.9 PREDEFINED SYMBOLS

Care must be exercised not to redefine any of the symbols in the attached list, as this can result in an erroneous assembly. These symbols are the ones used by the parameter file, UCASM.MAC.

C	000001	RIR	000010
C0	000040	RIRH	000011
C4	000040	RIRL	000010
C8	000020	RMW	000004
DOP		RPSW	000012
DOP1		RPSWH	000013
DOPCOD		RPSWL	000012
G	000000	RSRC	000004
GH	000001	RSRCH	000005
GL	000000	RSRCL	000004
I4	000001	RSVC	000002
I5	000002	S0	000000
I6	000004	S2	0004
JOP		S3	000006
JOP1		S4	000010
JOPCOD		S5	000012
LB	000001	S6	000014
LBC	000003	S7	000016
LITUP		SOP	
LOP1		SOP1	
LOPCOD		SOPCOD	
LRR	000001	SP	000014
N	000010	SPH	000015
NB	000200	SPL	000014
PC	000016	TG6	000001
PCH	000017	TG8	000001
PCL	000016	UB	000000
RBA	000002	UBC	000002
RBAH	000003	V	000002
RBAL	000002	Z	000004
RDST	000006	Z8	000100
RDSTH	000007		
RDSTL	000006		

7.9.1 Dummy Directives

THE FOLLOWING DIRECTIVES ARE ACCEPTED BY THE MICROASSEMBLER TO KEEP COMPATIBLE WITH SOURCE FILES PREVIOUSLY PREPARED FOR THE SYSTEM 10 MICROASSEMBLER(U11 .MAC). These directives do no produce any code,or have any effect on the assembly.

EMPTY0

EMPTY1

EMPTY2

EMPTY3

TRAN

7.10 OPCODE SUMMARY IN OCTAL AND HEX RADICES

The octal value is listed first and then the hex value is given in parenthesis. Add one to the stated value for the opcode that affects the flags.

00(0)=JMP
04(01)=RFS
1(1)=JXX
2(2)=AL
3(3)=CL
4(4)=NL
5(5)=IL
6(6)=LL

160(70)=RF
161(71)=SF
162(72)=CCF
163(73)=LCF
164(74)=RTSR
165(75)=LGL
166(76)=CIB
167(77)=CDB

170(78)=
171(79)=
172(7A)=
173(7B)=
174(7C)=
175(7D)=
176(7E)=
177(7F)=

200(80)=MB
202(82)=MW
204(84)=CMB
206(86)=CMW
210(88)=SLBC
212(8A)=SLWC
214(8C)=SLB
216(8E)=SLW

220(90)=ICB1
222(92)=ICW1
224(94)=ICB2
226(96)=ICW2
230(98)=ICB
232(9A)=ICW
234(9C)=OCB
236(9E)=OCW

240(A0)=AB
242(A2)=AW
244(A4)=CAB
246(A6)=CAW
250(A8)=ABC
252(AA)=AWC
254(AC)=CAD
256(AE)=CAI(CAWI)

260(B0)=SB
262(B2)=SW
264(B4)=CB
266(B6)=CW
270(B8)=SBC
272(BA)=SWC
274(BC)=DB1
276(BE)=DW1

300(C0)=NB
302(C2)=NW
304(C4)=TB
306(C6)=TW
310(C8)=ORB
312(CA)=ORW
314(CC)=XB
316(CE)=XW

320(D0)=NCB
322(D2)=NCW
324(D4)=
326(D6)=
330(D8)=SRBC
332(DA)=SRWC
334(DC)=SRB
336(DE)=SRW

```
340(EØ)=IB             360(FØ)=RIB1           370(F8)=R
342(E2)=IW             361(F1)=WIB1           371(F9)=W
344(E4)=ISB            362(F2)=RIW1           372(FA)=RA
346(E6)=ISW            363(F3)=WIW1           373(FB)=WA
350(E8)=               364(F4)=RIB2           374(FC)=OB
352(EA)=               365(F5)=WIB2           375(FD)=OW
354(EC)=MI             366(F6)=RIW2           376(FE)=OS
356(EE)=LTR            367(F7)=WIW2           377(FF)=NOP
```

CHAPTER 8

MICROPROGRAMMING TECHNIQUES

8.1 GENERAL

8.2 FUNDAMENTAL MICROPROGRAM CHARACTERISTICS

8.2.1 Entering User Microcode

8.2.2 Decoding the User Machine Instructions

8.2.3 Passing Arguments

8.2.4 Executing the User Machine Instructions

8.2.5 Updating the Processor Status Word

8.2.6 Rejoining the Machine Operating Cycle

8.3 USER MICROPROGRAMMING ENTRY AREA

THE USER MICROPROGRAMMING ENTRY AREA IS LOCATED AT CONTROL STORE MICROADDRESSES 3ØØØ-3ØØ4 INCLUSIVE. CONTROL OF THE MICROMACHINE IS TRANSFERRED TO THE MICROWORD STORE AT THESE LOCATIONS UNDER DIFFERENT CONDITIONS, AS EXPLAINED IN THIS SECTION. IN THE NORMAL APPLICATION, NOT ALL ENTRY AREA LOCATIONS WILL BE UTILIZED. THESE LOCATIONS SHOULD BE LOADED WITH THE ALL-ZERO MICROWORD, WHICH THE MICRPROCESSOR INTERPREIS AS A JMP Ø. WITH THIS PREPARATION, ANY INADVERTENT ATTEMPT TO ENTER THE USER CONTROL AREA WILL BE HANDLED AS A RESERVED INSTRUCTION, WHICH CAUSES A TRAP TO LSI-11 VECTOR LOCATION 1Ø.

NOTE THAT WHEN USING ADDRESSING MODE 1, THE ENTRY AREA LOCATIONS ARE AT THE BEGINNING OF WRITEABLE CONTROL STORE PAGE 1. THE EIS/FIS OPTION NORMALLY OCCUPIES MICROADDRESSES 2ØØØ-2777 BUT IS REMOVED IN A WCS EQUIPPED SYSTEM. THE MICROPROGRAMMER CAN MAKE USE OF THIS VACATED SPACE WITH CERTAIN RESTRICTIONS (SEE SECTION 8.12.1). WRITEABLE CONTROL STORE PAGE Ø.

THERE ARE THREE GENERAL EVENTS WHICH CAUSE CONTROL TO BE TRANSFERRED TO THE ENTRY AREA. THE FIRST TYPE OF EVENT OCCURS WHEN AN APPROPRIATE MACHINE INSTRUCTION IS FETCHED AND LOADED INTO THE INSTRUCTION REGISTER, RIR. THESE MACHINE INSTRUCTION VALUES CAN AFFECT A TRANSFER TO THREE OF THE MICROADDRESSES (3ØØØ,3ØØ1,3ØØ3). THE SECOND TYPE OF EVENT OCCURS AT POWER-UP (3ØØ2). THE THIRD TYPE OF EVENT OCCURS IN THE IMPLEMENTATION OF THE MICROPROGRAMMED EXTERNAL INTERRUPT TEST (3ØØ4).

8.3.1 Entry Microaddress 3ØØØ

CONTROL IS TRANSFERRED TO MICROADDRESS 3ØØØ WHENEVER A MACHINE INSTRUCTION IN THE RANGE ØØØ22Ø TO ØØØ227 (OCTAL) IS FETCHED AND DECODED BY THE TRANSLATION ARRAY. NOTE THAT THESE MACHINE INSTRUCTIONS ARE RESERVED BY DEC. A JMP Ø MICROINSTRUCTION MUST ALWAYS BE ASSEMBLED INTO MICROLOCATION 3ØØØ TO CAUSE A TRAP TO LSI-11 VECTOR LOCATION Ø.

8.3.2 Entry Microaddress 3ØØ1

MICROADDRESS 3ØØ1 IS THE MICROCODE ENTRY POINT FOR USER-MICROPROGRAMMING. CONTROL IS TRANSFERRED TO MICROADDRESS 3ØØ1 WHENEVER A MACHINE INSTRUCTION HAVING A VALUE IN THE RANGE Ø76ØØØ TO Ø76777 IS FETCHED AND DECODED BY THE TRANSLATION ARRAY. NOTE THAT THE ONLY LEGAL CUSTOMER OPCODES ARE IN THE RANGE OF Ø767ØØ TO Ø76777. IT IS THE RESPONSIBILITY OF THE USER-MICROPROGRAMMER TO CAUSE A JMP Ø MI-

CROINSTRUCTION TO OCCUR FOR ANY OPCODE IN THE RANGE 076000 TO 076677 SINCE THESE ARE RESERVED BY DEC. SEE SECTION 6.4.

8.3.3 Entry Microaddress 3002

CONTROL IS TRANSFERRED TO MICROADDRESS 3002 AT POWER-UP WHEN POWER-UP MODE 3 IS SELECTED. MODE SELECTION IS MADE VIA JUMPER OPTION ON THE LSI-11 PROCESSOR MODULE (SEE THE MICROCOMPUTER HANDBOOK) THE MICROINSTRUCTION LOCATED AT 3002 SHOULD JUMP TO A MICROROUTINE WHICH EXECUTES SPECIAL START-UP OPERATIONS, OR POSSIBLY A SYSTEM BOOTSTRAP.

SINCE THE WCS MEMORY IS VOLATILE, THE USER CANNOT EXPECT A MICROCODED BOOTSTRAP TO BE EXECUTED OUT OF USER CONTROL STORE IMMEDIATELY AFTER A POWER REMOVAL.

8.3.4 Entry Microaddress 3003

CONTROL IS TRANSFERRED TO MICROADDRESS 3003 WHENEVER A MACHINE INSTRUCTION HAVING A VALUE IN THE RANGE 075040-075777 IS FETCHED AND DECODED BY THE TRANSLATION ARRAY. NOTE THAT THESE MACHINE INSTRUCTIONS ARE RESERVED BY DEC AND A JMP 0 MICROINSTRUCTION MUST ALWAYS BE ASSEMBLED INTO MICROLOCATION 3003 TO CAUSE A TRAP TO LSI-11 VECTOR LOCATION 10.

8.3.5 Microaddress 3004

MICROADDRESS 3004 IS USED TO IMPLEMENT THE MICROMACHINE EXTERNAL INTERRUPT TEST. THIS TEST MAY BE PERFORMED DURING THE EXECUTION OF LONG MICROROUTINES WHEN AN EXTERNAL INTERRUPT OR THE EVENT LINE INTERRUPT IS EXPECTED AND MUST BE SERVICED. THE MACHINE-MICROMACHINE CONTROL FLOW PATHS RELEVANT TO THE EXTERNAL INTERRUPT TESTS ARE VISABLE IN FIGURE 2.16-1 AND 2.16-2. THESE ILLUSTRATIONS ARE REPRODUCED HERE AS FIGURE 8.1-1 AND 8.1-2. NOTE THAT INTERRUPT 6, (I6), HAS THE HIGHEST PRIORITY OF ANY DECISION IN THE INTERRUPT/TRAP DECISION CHAIN. NOTE ALSO THAT ONCE I6 IS SET, CONTROL ALWAYS RETURNS TO USER CONTROL STORE. IF EITHER AN EVENT (I2) OR A DEVICE (I3) INTERRUPT IS ACTIVE, CONTROL IS TRANSFERRED TO MICROADDRESS 3004. IF I6 HAS BEEN SET AND NEITHER I2 OR I3 IS ACTIVE, THE MICROPROGRAM COUNTER IS NOT MODIFIED AND MICROINSTRUCTION EXECUTION CONTINUES IN SEQUENTIAL FASHION. IN ORDER TO ENTER THE INTERRUPT/TRAP DECISION CHAIN, THE RSVC BIT (MI<17>) MUST BE SET. THE MICROPROGRAM WHICH IMPLEMENTS THE EXTERNAL INTERRUPT TEST IS SHOWN BELOW.

```
EXITST: SI        I6              ; SET I6
        XXX       8,A,, RSVC      ; CONTROL TO INTERRUPT TEST
        RI        I6              ; RESET I6
CONT:             ...             ;
```

The Set Interrupt (SI) microinstruction is used to set I6, I6 being specified in the argument field. The microinstruction at EXITST1+1 may be anything except a jump (unconditional or conditional). The RSVL argument assembles as a 1 at MI<17>. Immediately following is a Reset Interrupt (RI) microinstruction which restores I6 to its normal state (cleared). The reset operation must be included here to restore normal interrupt control flow. If neither of the two possible external interrupts is active and execution continues at CONT. This interrupt test will not respond to REFRESH (I0), TRACE (I4), or PFHALT (I1). Additional information on interrupt testing is contained in Section 8.6.7.

8.3.6 Entry Microaddress Summary

Because the microaddress entry points are positioned in sequential locations, the microinstructions located in the entry area 3000 to 3003 must all be unconditional jumps. Each JMP microinstruction then transfers control to a microroutine which implements the appropriate operations. The source code for the entry area normally appears as follows:

```
        LOC     3ØØØ            ; SET MICROPROGRAM COUNTER TO 3ØØØ
3ØØØ    JMP     Ø               ; TRAP RESERVED OPCODES ØØØ22X
3ØØ1    JMP     DECODE          ; ENTER HERE FOR OPCODES Ø76XXX
3ØØ2    JMP     PWRUP           ; ENTER HERE FOR POWER-UP
3ØØ3    JMP     Ø               ; TRAP RESERVED OPCODES Ø75ØØØ-Ø75777
3ØØ4    XXX     B,A             ; ENTER HERE IF I2 OR I3 PENDING
                                ; AND START INTERRUPT SERVICE

DECODE:                         ; START OF USER OPCODE DECODE

PWRUP:                          ; START OF POWER UP ROUTINE
```

8.4 MACHINE INSTRUCTION DECODING TECHNIQUES

The microprocessor Control chip will transfer control to user entry point (3001) in response to any machine instruction in the range 076000-076777. The microprogrammer's first task is to determine which of his special machine instructions has been fetched. Note that opcoded 076700 to 076777 are the only legal customer opcodes and opcodes 076000 to 076677 must cause a JMP microinstruction to occur.

8.4.1 No Decoding

In certain cases it is not necessary to perform any instruction decoding. Such a case occurs during microprogram development when WCS memory is only loaded with a single microprogram. Control therefore transferred directly to the starting microaddress where execution begins. When microprogram development is completed, the verified routine may become part of a larger WCS load module. The final load module would contain an instruction decoding mechanism along with the multiple execution routines. Note that all unused machine codes must perform a JMP Ø microinstruction to trap to LSI-11 vector 10.

8.4.2 Successive Comparison Decoding

One technique for determining where micromachine control should go next is the successive comparison technique. Principle in this technique is the ability to maintain in control store a 16-bit constant with which the machine instruction is to be compared. This ability is supplied by the literal class of microinstructions. Two successive LOAD LITERAL (LL) microinstructions allow the reference value to be placed in any designated register. Once the reference value is established, the Compare WORD microinstruction is used, followed by a conditional jump (JZ1). If the two words are equal, control will be transferred to the microaddress contained in the conditional jump field. Note that the comparison must be performed with the flag-affecting microinstruction, CWF. The jump is made contingent upon a condition code as opposed to a status bit because the latter is updated for each byte result, and may not reflect the results of the word comparison. Note also that the microprogrammer must document all possible flag results for the special machine instruction. In the case discussed here, the flag manipulation is for an intermediate purpose, and flag values will probably be modified later to reflect a different result.

The microprogram used to implement the above technique is presented below.

```
DECODE: LL      30,RDSTL        ; USE RDST FOR REFERENCE WORD
                                ; LOAD LOW BYTE.
        LL      175,RDSTH       ; LOAD HIGH BYTE

        CWF     RIR,RDST        ; COMPARE, AFFECT FLAGS
        JZ1     OP00            ; IF EQUAL, GO TO ROUTINE FOR 076700
        ICR1    RDSTH           ; INCREMENT REFERENCE WORD
        CWF     RIR,RDST        ; COMPARE AFFECT FLAGS
        JZ1     OP01            ; IF EQUAL, GO TO ROUTINE FOR 076701
        More Comparisons
        .
        .
        .
        jmp 0                   ; if none of them, trap to vector 10.
```

This microprogram uses two LOAD LITERAL microinstructions to establish the reference value of 076000 in the destination operand register. If the machine instruction has the same value, the jump condition will be satisfied. Otherwise additional comparisons are necessary. In preparation for the next comparison, the Increment Byte by 1 ICB1 microinstruction is used to increment the comparison constant.

The implementation of the successive comparison technique described above is highly generalized in that it can decode or recognize all of the 64 possible user opcodes. However, it is not likely that such a large number of special opcodes would be simultaneously used. An alternative implementation of the successive comparison technique employs a single Compare Literal (CL) microinstruction for each expected user opcode. Use of the Compare Literal microinstruction also has the advantage of not requiring any other registers to perform the operation. The microprogram which implements this technique appears as follows:

```
DECODE: CL          300,RIRL          ; IS IT 076700?
        JZ81        OP0               ; YES, GO EXECUTE
        CL          301,RIRL          ; IS IT 076701?
        JZ81        OP01              ; YES, GO EXECUTE
        CL          301,RIRL          ; IS IT 076702?
        JZ81        OP02              ; YES, GO EXECUTE
        .
        .
        .
        more comparisons
        .
        .
        .
        jmp         0                 ; if none of them, trap to vector 10.
```

The advantage of this microprogram is that it does not use any additional registers as scratch space, and it executes faster because it does not need to modify (increment) the reference value with a separate microinstruction.

8.4.3 Modified Jump Decoding

The microinstruction modification technique allows user opcodes to be decoded much faster, especially when a large number of decisions are necessary. This technique effectively uses the lower byte of the machine opcode as part of the microaddress portion of an unconditional jump microinstruction. The operation of the modify microinstruction (MM) microinstruction is discussed in Chapter 4. The two microinstruction sequence:

```
        MM          RDSIH,RDSIL       ; OR IN THE DISPATCH OFFSET
        JMP         0                 ; AND JUMP TO THE ROUTINE
```

produces an unconditional jump microaddress formed by the logical OR of the contents of RDSI with, in this case, 0. However, the argument used with the JMP instruction is not restricted to 0.

The final implementation of this technique uses only the low byte of the machine instruction register, which is stored in RIRH and the And literal (A_L) microinstruction masks off all but the lower six bits of the opcode in RIRH. Note that RPSWL contains all zeroes upon entry of the control store (see Section 8.6.5.7). The argument of the JMP instruction is changed so that it establishes a base address within the microaddress range. If the user opcodes to be decoded are sequential, a jump or dispatch table is required to transfer control to the appropriate routines. These requirements are illustrated in the following example:

```
DECODE: CL       175,RIRL       ; IS IT A LEGAL USER OPCODE?
        JZB      DC1            ; MAYBE
DC0:    JMP      0              ; IF NOT, TRAP TO VECTOR 10
DC1:    MB       RIRH,RIRL      ; COPY THE HI BYTE
        AL       300,RIRL       ; MASK OFF ALL BUT 2 HI BITS
        CL       300,RIRL       ; IS IT A LEGAL OPCODE
        JZB0     DC0            ; NO, GO TRAP
        AL       77,RIRH        ; MASK ALL BUT LOWER 6 BITS
        MM       RPSWL,RIRH     ; LO BYTE OF IR




3200:   JMP      OP00           ; 1ST USER OP
        JMP      OP01           ; 2ND USER OP
        JMP      OP02           ; 3RD USER OP
```

In this example, the starting microaddress of the jump table is 3200 octal. Micromachine control is transferred here when a machine opcode of 076700 is decoded. Control is subsequently transferred to microaddress OP00 where the microprogram appropriate to the execution of the 076700 machine opcode is stored. Note that there is a check for reserved opcodes performed first (see Section 8.4).

8.5 PASSING OPERANDS TO USER MACHINE INSTRUCTIONS

A universal attribute of all LSI-11 data manipulation machine instructions is that they allow very flexible operand addressing. In the design of new machine instructions, the microprogrammer must provide some means for delivering the operands to the micromachine for processing. Since one of the microprogrammer's goals is to optimize a specific target function, a total generality in operand addressing is usually not desired. Furthermore, generalized operand addressing would use microprogram space which may be more profitably dedicated to execution microcode.

8.5.1 Predefined Operand Addressing

The general addressing information can be fixed at the time of microprogram assembly. In this case the user machine instruction would expect to find its input operands at a predefined location and would deliver the processed results or output operand to a predefined location. Predefined input and output operands may be located either in the LSI-11 processor registers (R0-R7) or in LSI-11 machine memory. Since the WCS memory contains only "control store" information, it may not contain either input (source) or output (destination) operands. Predefined machine memory address information is stored within the microprogram via the Load Literal (LL) microinstruction.

8.5.2 Register Operand Addressing

The format of user-defined machine instructions is fixed except for the lower 6 bits of the opcode. If addressing modes similar to the LSI-11 machine instructions are desired, the instruction first must have an additional word (in the next successive memory location) to contain the information after the initial instruction fetch, this successive word can be fetched with the automatic loading of the G register by bits <8:6>. Reference to the description of the Input Word (IW) microinstruction, will show that G register loading is one of its specificable options. The machine instruction fetch operand which is part of the base microcode specifies that the source register address information (bits 6-8) be loaded into the G register. Figure 8.3 illustrates the role played by bits <8:6> in forming the source operand for a double operand instruction.

As further noted in Figure 8.3, the lowest 3 bits of the user machine opcode, <2:0> (corresponding to a destination register), may also be loaded into the G register, but only by execution of the Load G (Register) Low (LGL) microinstruction. The availability of this and the previous register operand addressing technique will facilitate operand addressing which is similar to that used by the LSI-11 machine instructions.

8.5.3 Complex Operand Addressing

The microprogrammer has the option of implementing whatever operand addressing modes are desired, but total generality is generally not feasible used for general addressing mode calculations. This is because the base microcode is integrally designed with the Control chip translation array and the resulting microinstruction sequences are not available for user-microprogramming.

8.6 MICROPROGRAMMING THE USER MACHINE INSTRUCTION

8.6.1 Defining The Instruction

The first step in creating a new machine instruction is to define as precisely as possible exactly what functions the instruction is supposed to accomplish. One approach which is an aid to definition is to program the desired functions in the LSI-11 assembly language. This approach usually will also reveal the suitability of micoprogramming to accomplish a desired collection of functions.

8.6.2 Documenting the Instruction

The requirements for instruction documentation will vary with the nature of the functions provided. Data manipulation instructions which only process data and perform no micromachine I/O are probably the simplest to document. Such instructions would accept as input, data which is already located in processor registers. The output data would also be deposited in a processor register as opposed to a memory or device location.

At the opposite extreme are user machine instructions which are intended for a real time control and I/O environment. These instructions would make use of data access microinstructions and also the control capability of the two TTL control bit fields, MI<21:18> and MI<23:22>. In this context, timing and sequencing information would be essential.

It should be remembered that the designer (or microprogrammer) of the new machine instruction may not always be the user. In this case, documentation for the new instruction should be as complete as possible. Such documentation includes information on input and output operands, resultant PSW condition code flags for all possible situations, and execution timing for all possible situations.

8.6.3 Temporary Flag Use

During the execution of user-designed machine instructions it is often necessary to monitor ALU results. These results are available via the status bit and condition code flag register and may be used to effect microprogram control via the conditional jump microinstructions. However, when the machine instruction's operations have been completed, additional microinstructions may need to be used to update the Processor Status Word Conditional Code flag, (N,Z,C,V). It must be remembered that the PSW resides in a microprocessor register and that it is partially separated from the ALU flag register. The microprogrammer must decide what PSW flag states accurately represent the operation's executed and provide for their implementation.

8.6.4 Executing Machine-Level I/O Operations

The microprogrammer has complete freedom in implementing any of the 5 possible machine-level I/O operations (DATI, DATO, DATOB, DATIO, DATIOB) as part of a new machine instruction. In designing the I/O portion of the instruction,the operator should make detailed reference to Chapter 5. All I/O operations other than Input Status, and Output Status, require a response from a system bus device and effectively transfer control outside the processor. Under normal conditions, a non-responding bus device will cause a bus error trap. The microprogrammer must recognize the possibility of non-responding bus devices and not validate any system conventions (e.g., register designations).

8.6.4.1 Bus Error Trap Control - The source operand scratch register (RSRC) is used in base microcode to hold vector values when trap or interrupt operations are performed. The microprogrammer can create special trap vectors.

8.6.5 S

CRATCH REGISTER USAGE

Many processes require scratch register space in which to store intermediate results. This section discusses the function of each register contained in the microprocessor data chip and the conditions under which it may be used by the user-microprogrammer.

8.6.5.1 Source Operand Register - The source operand register may be used for scratch store during microprogram execution. It may be left in any state at the conclusion of the user microprogram.

8.6.5.2 Destination Operand Register (RDST) - The destination operand register may be used for scratch storage during microprogram execution. It may be left in any state at the conclusion of the user microprogram.

8.6.5.3 Instruction Register (RIR) - Every machine instruction, whether standard or user-designed, is loaded into the instruction register as part of the machine instruction fetch operation. It should be remembered that the low byte of the machine instruction is loaded into the high byte of the instruction register, (RIRH). Similarly, the high byte of the machine instruction is loaded into the low byte of the instruction register, (RIRL). This byte-swapped loading is performed to facilitate translation operation operations in the LSI-11 base microcode.

It can happen that the user opcode transmits no information to the micromachine. In this case, its only purpose is to transfer micromachine control to the appropriate microprogram. The microprogram may then use the upper and lower byte of RIR for scratch storage. However, RIR does have specific uses when communicating with ODT microcode, as explained in Section 8.7.

8.6.5.4 Bus Address Register (RBA) - The microprocessor bus address register is generally used to store the system bus address required in most data access operation. This register has no auxilliary function of importance and can be used for scratch space during microprogram execution and it can be left in any state at the conclusion of the user microprogram.

8.6.6 Scratch Register Usage

Many processes require scratch register space in which to store intermediate results. This section discusses the function of each register contained in the microprocessor data chip and the conditions under which it may be used by the microprogramm.

8.6.6.1 LSI-11 Processor Registers (R0-R5) - The LSI-11 processor general purpose registers R0 through R5 should not be used for scratch storage. The main reason is that arguments relating to other processes are normally stored here and should not be modified or destroyed by execution of a user machine instruction.

If the new instruction uses the general purpose registers for operand transmission, there may be opportunity for scratch space at an intermediate stage of execution. For example, the register which will eventually contain the output operand may be used for scratch purposes as long as such use does not destroy needed data.

Another reason for avoiding R0 through R5 as scratch space is that they can only be addressed indirectly. Consequently, the indirect addressing G register must be loaded with appropriate values, requiring extra microinstructions. Directly addressed microprocessor register do not require this execution overhead.

8.6.6.2 LSI-11 Stack Pointer (R6) and Program Counter (R7) - The stack pointer and program counter should never be used as scratch registers.

8.6.6.3 Processor Status Word Register (RPSW) - The microprocessor register called RPSW actually contributes only 4 bits to the PSW that is visable to the conventional machine-level programmer. PSW <7:4> are stored in RPSWH<7:4>. The lower 4 bits of RPSWH are replaced by the four condition codes of the ALU flag register when a MFPS machine instruction is executed (see Section 8.8). The lower byte of RPSW is dedicated to the use of the bus error flag register. Each bit signifies a specific bus error context and RPSWL is inspected when a bus error occurs to determine appropriate recovery measures.

RPSWL may be used as temporary scratch storage only if no data access operations are being executed. However, before a data access operation is initiated, RPSWL must be loaded with all zeros. Note that this register will always contain all zeros upon entry to the user control store.

8.6.7 Creation and Use of Arbitrary Flags

Any of the available scratch registers may be used as an arbitrary flag register. The meaning of each of the 8 bits in a byte register will be determined by the microprogrammer. As a first step in arbitrary flag register usage, the register should either be cleared or set to an initial value by a Load Literal microinstruction.

8.6.7.1 Setting an Arbitrary Flag - An important requirement in setting a flag is usually to not disturb any other flags in the register. This may be accomplished by using the Or Byte (ORB) microinstruction. Since OR Byte is a register format microinstruction, the proper constant must be prepared in the B-field register.

8.6.7.2 Clearing an Arbitrary Flag - Just as the OR function allows a single flag to be set without disturbing the other flags in a register, the And Complement function allows a single flag to be cleared. This is accomplished by the And Complement Byte (NCB) microinstruction in conjunction with a prepared B-field register. The B-field register contains a 1 in the bit location to be cleared. The NCB microinstruction forms the complement of the B-field register and ANDs this with the flag register contents. The flag register is left with a zero in the desired postiion and all other positions unmodified.

8.6.7.3 Checking an Arbitrary Flag - The most direct means of checking a single arbitrary flag in a byte flag register is via a literal microinstruction. The flag to be tested for is assembled into the literal field of the Test Literal microinstruction. The logical AND of the arbitrary flag register and the literal field is used to update the ZB and NB status bit flags. If the tested flag is present, ZB will be cleared. If the tested flag is absent, ZB will be set. Appropriate microprogram control is achieved by a Test Literal - Conditional Jump microinstruction sequence.

8.6.8 Interrupt Considerations

The overriding fact regarding interrupt service is that all interrupts are completely ignored while control stays within the micromachine, executing user microcode. If the programming environment does not involve real time interrupts for which service latency is an important consideration, lengthy microcode routines may not pose any problem. If interrupts are expected, however, they may be tested by the technique described in Section 8.3.5.

8.6.8.1 Aborting a Microprogram - Using the external interrupt test technique, an active Event or Device interrupt will cause control to be transferred to microaddress 3004. This micro location will usually contain the beginning of a microprogrammed abort routine which restores the process under execution to its beginning state (e.g., decrement the PC by 2). This procedure of course requires that the input arguments are still available, or that they may be quickly recovered an placed in their original locations. If the beginning state of the process is not recoverable, the process may simply be uninterruptable.

8.6.8.2 Modifying the Program Counter - In the event of an aborted routine, the program counter must be decremented by 2. In this way, the user machine instruction will again be fetched when interrupt service has been completed. The following microinstruction sequence may be used to decrement the program counter by one word location.

```
        AL      376, PCL        ; BUMP BACK PCL
        CDB     PCH             ; BORROW, IF NECESSARY
```

Once the program counter is bumped back, control is returned to the normal machine operating cycle where the pending external interrupt will receive service.

8.7 USING CONSOLE ODT MICROCODE

The microcoded routines which impelement LSI-11 console ODT are usually highly specialized, and as such, are not usable by the microprogrammer. However, two functions may be attractive in that they can provide communication between the micromachine and the console TTY. Due to the complexities of dynamic memory refreshing, it is recommended that possible refresh requirements be satisfied by either DMA refreshing or by self-refreshing memory.

8.7.1 Dynamic Memory Refresh Requirement

If neither DMA refreshing or self-refreshing memory is available, two console ODT microcode routines may still be used. However, some additional preparation is required to support processor-controlled refresh. This support involves setting a special flag which allows memory refresh to interrupt console ODT microcode and subsequently return to the interrupted location. Due to the delays inherent in driving the console TTY, the need for memory refresh can almost certainly be expected to occur while testing the Done bit in either the receive or transmit CSR for the terminal device.

8.7.2 Printing a Character on the Console Device

8.7.2.1 Passing the Character -

8.7.2.2 Setting the Print Routine Control Register -

8.7.3 Reading a Character From the Console Device

8.7.3.1 Receiving the Character -

8.7.3.2 Setting the Read Routine Control Register -

8.7.4 Processor Priority

8.7.5 Trace Bit

8.7.6 ALU Condition Codes

8.8 MICROPROGRAMMING USER-DEFINED TRAP VECTORS

New User-defined trap operations may be implemented by the microprogrammer. This is efficiently accomplished by setting up a vector address and transferring control to the base microcode routine which executes all other LSI-11 trap operations.

8.8.1 Creating the Vector Addresses

The vector address of the special user trap is inserted into the source operand scratch register, RSRC. Usually this is done with two consecutive Load Literal microinstructions. RSRC$_H$ receives the high byte of the vector address and RSRCL the low byte.

8.8.2 Joining the Base Microcode

Once the special vector address has been loaded into both bytes of RSRC, an unconditional jump microinstruction transfers control to the bas microcode routine which executes the trap operation. The microinstruction sequence is as follows:

```
        LL      VEC LOW BYTE, RSRCL      ; LOAD VECTOR LOW BYTE
        LL      VEC HIGH BYTE, RSRCH     ; LOAD VECTOR HIGH BYTE
        JMP     1402                      ; JUMP TO TRAP ROUTINE
```

8.9 MICROPROGRAMMING SYNCHRONIZED CONTROL SIGNALS

The synchronization of TTL control signals with microprogram execution is determined at the time of microprogram assembly. The control function code appears as the final argument in the microprogram source code. The detailed construction of microassembly source language is presented in Chapter 7 for each microinstruction format. The microprogrammer has the option of pre-calculating the octal equivalent of the combined standard and extended TTL control bit codes, or the individual codes can be OR'd together by the microassembler.

8.10 Standard TTL Control Bits

The standard TTL control bit codes are assembled into MI<21:18>. Of the 16 possible codes, 8 are employed by the LSI-11 interface circuitry. One additional code, 07, is used to swap user control store pages for the WCS PAGING mode. The remaining 7 codes, 00 through 06, may be used by the microprogrammer. These codes must be decoded by special decoding hardware which is connected to the system backplane. Section 6.7.5 presented information on standard TTL control function codes as well as hardware connection details relevant to the design of decoding circuitry.

As explained in Section 6.7.5 the standard TTL control bits are valid on the microinstruction bus from the begining of PH1 to the end of PH3. Decoding of the 8 functions used by the LSI-11 module is enabled by PH3 of the microcycle, and this signal, SPR3 H, is also available at the system backplane.

Due to the pipelining techniques used within the micromachine, the TTL control bits can be enabled before the microinstruction with which they are assembled is actually executed. This occurs for any 2 cycle microinstructions, in particular, conditional or unconditional jump microinstructions. Because the microinstruction sequence is modified, all jump microinstructions require 2 microcycles to execute. Microaddress modification does not occur until the second microcycle, and the control store contents fetched as a result of normal microprogram counter sequencing must be discarded. The microprocessor Control and Data chips recognize the jump format and simply ignore the control store information on the microinstruction bus. However, MI<21:18> is decoded by combinational logic. In the following example, the control code of the microinstruction following the JMP is decoded and enabled, even through the microinstruction flow does not include the following microinstruction.

```
3100    LL     0, RSRCL       ;
3101    JMP    ALTLOC         ;
3102    CL     10, RDSTH,,01  ;
```

Note: 01 decoded and enabled during second cycle of JMP microinstruction.

8.10.1 Extended TTL Control Bits

The extended TTL control bits are assembled into the same source field as the standard TTL control bits. The extended control bits are fetched at the beginning of PH1 and are valid through the end of PH4. As such, they are not subject to enabling via any microcycle phases. However, extended control signals may appear erroneously when contained in a microinstruction following a 2 cycle microinstruction in the same fashion as the standard control signals.

The assembly value corresponding to MI<22> is 20 and that corresponding to MI<23> is 40. These values may be ORd with the standard TTL control bit code, as shown below:

```
          LL        0,RSRCL,,, 01|20|40 ; CONTROL CODE 01
                                        ; PLUS MI<22> PLUS MI<23>
```

Recall that MI<23> is shared with the microaddress trace buffer logic. This has no effect on MI<23> but it may complicate microprogram debugging.

8.11 CONTROLLING THE MICROINSTRUCTION FLOW

8.11.1 Unconditional Jump and Return Microinstruction

8.11.2 Conditional Jump Microinstructions

8.11.3 Multi-Level Subroutine Techniques

8.12 USER CONTROL STORE MEMORY MAP

8.12.1 Reserved Microaddress Locations

8.12.2 Implementing Paged Control Store

8.13 MICRO OUT FUNCTIONS

8.13.1 Supporting the START Pointer

8.13.2 Measuring I/O Device Response Time

8.14 BUILDING THE MICROPROGRAM - A PROGRESSIVE SUMMARY

8.14.1 Machine Operation With User Control Disabled

8.14.2 User Control Store Enabled - JMP Ø Microwords

8.14.3 Single JMP and Exit

8.14.4 Multiple JMP and Exit

8.14.5 Printing a Character, a Word

8,14,5,1 Console TTY Driver -

8,14,5,2 Microcode ODT Print Subroutine -

8,14,6 Multiple User Opcodes - Multiple Printed Words

8,14,6,1 Successive Comparison Decoding -

8,14,6,2 Modified Jump Decoding -

8,14,6,3 Modified Jump Decoding Without Jump Table -

8,14,7 Selective Microprogram Execution with MODT

8,14,7,1 Microprogram Tracing with MODT -

8,15 EXAMPLE

8,16 EXAMPLE

8,17 EXAMPLE

8.18 A SUMMARY OF MICROPROGRAMMING TECHNIQUES

CHAPTER 9

WRITEABLE CONTROL STORE INSTALLATION

9.1 GENERAL

This chapter provides a guide to the installation of the Writeable Control Store module in the LSI-11, PDP-11/03 system. It is recommended that this chapter be read completely before beginning any phase of the installation procedure. Module installation should procede according to the Installation Check List contained in section 9.7.

9.2 SYSTEM CONFIGURATION

The LSI-11 Writeable Control Store option places very few restrictions on the PDP-11/03 system configuration.

9.2.1 Module Location

Because of its direct connection to the microinstruction bus, the WCS module must be located in the backplane slot immediately adjacent to the LSI-11 processor module. Backplane configurations are illustrated in Figure 9.1.

WCS MODULE LOCATION

FIGURE 9.1

9.2.2 Memory Requirement

The operation of the WCS module does not place any special demands on system memory space. However, the WCS Development software (QVK40-YY) requires 16K words of memory.

9.2.2.1 Memory Refresh Restrictions - It is recommended that all memory modules used in a LSI-11 WCS system contain self-refresh capability such as the MSV11-C or MSV11-D memory modules. The self-refresh feature frees the microprogrammer from the requirement to allow ample opportunity for CPU microcoded refresh to execute.

If processor-controlled memory refreshing must be used, the microprogrammer must give special attention to the techniques described in section 8.6.1 if lengthy microinstructions are to be implemented.

An alternative to processor controlled refreshing is the use of DMA-controlled refreshing, as supplied by the REV11-A an d REV11-C modules. This approach also frees the microprogrammer from the real-time restrictions, while allowing system memory to be used which does not have the self-refresh feature.

9.3 MICROINSTRUCTION BUS INTERCONNECT CABLE

The microinstruction bus interconnect cable-plug assembly extends the microinstruction bus from the the LSI-11 micromachine on the processor module to the WCS module. Extreme caution should be exercised when either installing or removing the cable. It should be emphasized that a half-installed cable provides a direct electrical connection to the MOS-level circuitry on either the processor module or the WCS module. The cable's open end should be guarded against inadvertent contact with any electrical potential.

The cable is connected between the third MICROM socket E32, (normally used for the KEV11 option) on the LSI-11 processor module (Etch Rev, F CS Rev Y or greater) and the single 40-pin socket on the WCS module. The cable should be inserted into both sockets before the 2 modules are inserted into the backplane/card guide assembly. The processor module should be installed last, taking care to avoid contact with the cable-plug assembly already in the WCS socket.

9.4 WCS Module Switch Options

The DIP switch on the WCS module is used to configure the microaddress range to which the WCS will respond. They should be set up to the desired address mode as shown in Section 6.2.2.1.

9.5 Checkout with LSI-11 Console ODT

T.B.D.

9.6 Checkout with MODT Under RT-11 V3.0

T.B.D.

9.7 Installation Checklist

T.B.D.