

**KD11-K processor
detailed technical
description manual**

digital equipment corporation • maynard, massachusetts

1st Edition, November 1977

Copyright © 1977 by Digital Equipment Corporation

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

Digital Equipment Corporation assumes no responsibility for any errors which may appear in this manual.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts:

DEC	DECtape	PDP
DECCOMM	DECUS	RSTS
DECsystem-10	DIGITAL	TYPESET-8
DECSYSTEM-20	MASSBUS	TYPESET-11
		UNIBUS

CONTENTS

Page

CHAPTER 1 BUS CONTROL DETAILED LOGIC DESCRIPTION

1.1	SCOPE	1-1
1.2	INTRODUCTION.....	1-1
1.3	TIMING PULSE GENERATION	1-1
1.3.1	Pulse Suppression	1-8
1.3.2	Stop Clocks	1-10
1.3.3	Single Clock	1-10
1.4	INTERNAL ADDRESS REGISTERS.....	1-15
1.5	I/O PAGE EXPANSION	1-15
1.6	DATA TRANSFERS	1-19
1.6.1	Bus Command Generation	1-19
1.6.2	Bus Command Flow Diagrams	1-22
1.6.3	DATI Flow	1-22
1.6.4	DATIP Flow	1-38
1.6.5	DATO/DATOB Flow	1-39
1.7	SERVICE	1-40
1.7.1	Unibus Interrupt Flow.....	1-41
1.7.2	NPR Flow	1-49
1.7.3	Jam	1-57

CHAPTER 2 KT CACHE MEMORY DETAILED LOGIC DESCRIPTION

2.1	CACHE MEMORY CONCEPTS	2-1
2.1.1	Scope	2-1
2.1.2	Introduction.....	2-1
2.1.3	Cache Memory Organization.....	2-1
2.1.4	Program Locality	2-2
2.1.5	Direct Mapping Cache	2-3
2.1.6	Write-Through and Write-Back.....	2-4
2.2	PDP-11/60 CACHE.....	2-5
2.2.1	Scope	2-5
2.2.2	PDP-11/60 Cache.....	2-5
2.2.3	Cache Data and Address Paths	2-9
2.2.3.1	Physical Address Bits 17:11	2-9
2.2.3.2	Cache HIT.....	2-9
2.2.3.3	Physical Address Bits 10:01	2-13
2.2.3.4	Physical Address Outputs to Bus Control.....	2-13
2.2.3.5	Store Outputs	2-13
2.2.3.6	Store Inputs.....	2-14
2.2.3.7	Cache Control Register (CCR)	2-16

CONTENTS (Cont)

	Page
2.2.4	Memory Management2-16
2.2.4.1	Memory Relocation.....2-16
2.2.4.2	Program Relocation2-19
2.2.4.3	Virtual to Physical Address Conversion2-20
2.2.4.4	PAR/PDR Registers2-23
2.2.4.5	Memory Management Register.....2-28
2.2.5	Stack Limit Register2-29
2.2.5.1	Functional Description.....2-32
CHAPTER 3 DATA PATHS DETAILED LOGIC DESCRIPTION	
3.1	SCOPE3-1
3.2	THE ARITHMETIC AND LOGIC UNIT3-1
3.2.1	The Carry-In Multiplexer3-5
3.2.2	The D(C) Multiplexer.....3-5
3.3	THE D REGISTER3-6
3.4	THE SHIFT REGISTER (SR).....3-6
3.4.1	Residual Control Register.....3-9
3.4.2	The XMUX.....3-9
3.5	THE SHIFT TREE3-10
3.5.1	The AMUX.....3-11
3.5.1.1	The Counter Register.....3-11
3.5.2	The BMUX3-11
3.5.3	The CMUX3-11
3.5.4	The SENDMUX3-14
3.6	A AND B SCRATCHPADS3-14
3.6.1	Scratchpad Address Selection3-17
3.6.1.1	Chip Enable.....3-22
3.6.1.2	Read/Write Input.....3-22
3.6.2	BUS AIN and BUS BIN3-25
3.7	THE CONSTANT SCRATCHPAD (CSP)3-25
3.8	THE BUS ADDRESS (BA) REGISTER3-27
CHAPTER 4 PROCESSOR CONTROL DETAILED LOGIC DESCRIPTION	
4.1	SCOPE4-1
4.2	INTRODUCTION.....4-1
4.3	MICROWORD CONTROL STORE4-3
4.4	SEQUENCING4-3
4.5	THE MICROBRANCH MULTIPLEXER.....4-8
4.5.1	BUT (CASE).....4-8
4.5.2	BUT (COUNT).....4-12
4.5.3	NUA and Return Registers.....4-14
4.5.4	NUA and Return Address Control4-15
4.6	INSTRUCTION DECODING4-16
4.6.1	Instruction Register4-16
4.6.2	Preliminary IR Decoding (K304)4-21

CONTENTS (CONT)

		Page
4.6.3	Microbranch Conditions (K305).....	4-21
4.6.3.1	Instruction 1 Branch Conditions.....	4-21
4.6.3.2	Instruction 5 Branch Conditions.....	4-21
4.6.3.3	Condition Code Control (V and C)	4-34
4.6.3.4	Byte Decode	4-38
4.6.3.5	Instruction Branch ROM.....	4-38
4.7	OVERLAP DECODE.....	4-38
4.8	PROCESSOR STATUS WORD (PS, PSW).....	4-46
4.9	FLOATING POINT STATUS (LOW BYTE) - (FPS)	4-46
4.10	KT DECODE AND CONTROL	4-50
4.11	MICROADDRESS MATCH AND MICROBREAK (U BREAK).....	4-51
4.12	SHORT AND LONG TERM FLAGS.....	4-55
4.13	EMIT REGISTER.....	4-57
4.14	CURRENT MICROADDRESS REGISTER (CUA).....	4-58
4.15	BUS DIN MULTIPLEXER	4-59
4.16	MICROCONTROL (UCON) INTERFACE.....	4-60

FIGURES

Figure No.	Title	Page
1-1	Bus Control Circuits.....	1-2
1-2	Microcycle Definition	1-4
1-3	Clock Pulse Generation (K603)	1-5
1-4	Clock Pulse Timing	1-6
1-5	ECL to TTL Conversion	1-7
1-6	M7876 Board Showing S1 and S2 Placement.....	1-11
1-7	Single Step Clock Operation	1-12
1-8	HALT/S1 Switch (Single Step Jam Routine)	1-14
1-9	I/O Page Expansion, NPR Transfer Control Latch.....	1-18
1-10	DATI Flow	1-24
1-11	DATIP Flow	1-30
1-12	DATO/DATOB Flow.....	1-33
1-13	Service Requests.....	1-42
1-14	Bus Interrupt Flow	1-44
1-15	NPR Flow.....	1-51
1-16	NPR Invalidation.....	1-56
1-17	Jam Logic.....	1-59
2-1	Relationship of Cache to Main Memory and the CPU	2-2
2-2	Direct Mapping Cache Memory System	2-3
2-3	18-Bit Byte Address Breakdown (4 Words per Block, 64 Blocks).....	2-4
2-4	Cache Tag and Data Fields.....	2-6
2-5	Cache Addressing (Simplified).....	2-7
2-6	Direct Mapping in the PDP-11/60 Memory System.....	2-8

FIGURES (CONT)

Figure No	Title	Page
2-7	Cache Addressing and Data Paths	2-10
2-8	Cache Control Register (CCR)	2-11
2-9	Data Multiplexer Inputs	2-12
2-10	Cache Data Interface.....	2-15
2-11	Simplified Memory Relocation Example	2-19
2-12	Relocation of a 32K Word Program Into 124K Word Physical Memory	2-20
2-13	Cache Memory Relocation (PAR).....	2-21
2-14	Internal Address Relocation	2-22
2-15	Page Status and Control (PDR).....	2-25
2-16	Example of an Upward Expandable Page	2-26
2-17	Example of a Downward Expandable Page.....	2-27
2-18	Memory Management Registers	2-30
2-19	Stack Limit Register (SLR).....	2-31
3-1	Data Flow Loops	3-2
3-2	ALU Operation.....	3-3
3-3	Shift Register (SR) and XMUX.....	3-8
3-4	Microcycle Shift Tree Operations	3-10
3-5	Counter Register and Shift Tree.....	3-13
3-6	Shift End Multiplexer Operation (Example: Shift Right Seven).....	3-15
3-7	A and B Scratchpads	3-16
3-8	Scratchpad Assignments.....	3-17
3-9	Scratchpad Addressing (ASP).....	3-18
3-10	A Scratchpad (ASP) Timing	3-20
3-11	C Scratchpad (CSP).....	3-26
3-12	BA Register.....	3-28
4-1	Processor Control Block Diagram	4-2
4-2	ROM Control Store Addressing (Sequencing)	4-5
4-3	Microaddress Modification	4-7
4-4	ROM Control Store Addressing [Example: BUT (CASE)].....	4-11
4-5	ROM Control Store Addressing [Example: BUT (COUNT)].....	4-13
4-6	Counter Register Timing [Example: BUT (COUNT) Used to Count Four Passes Through a Loop]	4-14
4-7	Return Address Control (Simplified)	4-15
4-8	Instruction Decode.....	4-17
4-9	IR Clock Enable (Cache DATI).....	4-18
4-10	IR Clock Enable Timing (Cache DATI Hit).....	4-19
4-11	IR Clock Timing (Cache DATI Miss – Unibus Cycle).....	4-20
4-12	A Class Instruction Decoding	4-23
4-13	B Class Instruction Decoding	4-24
4-14	C Class Instruction Decoding	4-25
4-15	D Class Instruction Decoding.....	4-26
4-16	E Class Instruction Decoding	4-27
4-17	F Class Instruction Decoding	4-28
4-18	G Class Instruction Decoding.....	4-29
4-19	V and C Condition Code Control	4-35
4-20	Overlapped and Nonoverlapped Fetches	4-41
4-21	Overlap and Prefetch Applications	4-44
4-22	Fetch Overlap.....	4-45

FIGURES (CONT)

Figure No	Title	Page
4-23	Processor Status Word	4-48
4-24	FPS (Low Byte).....	4-49
4-25	FPS Timing (Low Byte).....	4-50
4-26	Microaddress Match and Microbreak.....	4-54
4-27	Long and Short Term Flags.....	4-56
4-28	Emit Register	4-57
4-29	CUA Register Timing.....	4-58
4-30	Log CUA Register.....	4-59
4-31	BUS DIN Multiplexer	4-61
4-32	UCON Interface.....	4-62
4-33	UCON Control Word	4-63
4-34	UCON Signal Generation	4-64
4-35	UCON Interface (Processor Control and KT Cache)	4-67
4-36	UCON Interface (Bus Control).....	4-68

TABLES

Table No.	Title	Page
1-1	Bus Control Circuits.....	1-3
1-2	Pulse Suppress Flip-Flop (K604)	1-9
1-3	Internal Address Selection (E58 and E49)	1-16
1-4	Internal Address (E59, E70, E77).....	1-17
1-5	Read and Write Data Transfers	1-20
1-6	Bus Command ROM (E44)	1-21
1-7	Interrupt Sequence	1-43
1-8	NPR Sequence	1-50
1-9	Jam Condition Descriptions	1-58
2-1	Cache Write Signals.....	2-17
2-2	UCON KT/Cache Functions	2-18
2-3	PAR/PDR Address Assignments	2-23
2-4	ACF Keys	2-26
2-5	Comparison of Address and SLR	2-32
2-6	Detecting Type of Violation.....	2-33
3-1	ALU Control Multiplexer	3-4
3-2	CINMUX Input Selection.....	3-5
3-3	COUT Field Definition of Data Carry [D(C)]	3-5
3-4	Shift Register (74S194)	3-7
3-5	XMUX (74S257).....	3-9
3-6	Shift Tree Selection Control.....	3-12
3-7	SENDMUX Output.....	3-14
3-8	ASP Address Sources	3-19
3-9	BSP Address Sources.....	3-21
3-10	AEN and BEN Field Supplies.....	3-23

TABLES (CONT)

Table No.	Title	Page
3-11	A and B Scratchpad ROM Patterns (K405).....	3-24
3-12	BUS AIN and BUS BIN Connections.....	3-25
4-1	Processor Control Circuits	4-1
4-2	Microword Bits	4-4
4-3	Branch Microprogram Tests (BUTs)	4-9
4-4	Active BUT ROM (K207)	4-10
4-5	Preliminary IR Decode (K304)	4-22
4-6	Instruction 5 Branch ROM (E78).....	4-30
4-7	Instruction 5 Branch ROM (E88).....	4-31
4-8	Floating Point Branch ROM (E86)	4-34
4-9	Overflow-Carry ROM (E85).....	4-36
4-10	Overflow-Carry ROM (E62).....	4-37
4-11	Byte Decode ROM (E60).....	4-39
4-12	Instruction Branch ROM (E83)	4-40
4-13	Overlap Decode ROM (E80)	4-43
4-14	PSW Bit Assignments	4-47
4-15	KT Decode ROM (E47)	4-52
4-16	User Mode Multiplexer (E28)	4-53
4-17	UCON Signals	4-66

CHAPTER 1

BUS CONTROL DETAILED LOGIC DESCRIPTION

1.1 SCOPE

This chapter describes the operation and functions of the bus control circuits in the PDP-11/60 processor. These circuits are contained on two hex boards, M7876 Timing and M7877 Status, which are described together in the following paragraphs.

1.2 INTRODUCTION

9 The major bus control circuits are shown in Figure 1-1; their functions are listed in Table 1-1. These functions can be consolidated into two purposes for bus control: timing pulse generation and interface control. The basic timing pulses used throughout the base machine originate in the bus control. Interfacing of the base machine with external devices, i.e., peripherals and memory on the Unibus and the operator's console, is controlled by bus control circuits.

1.3 TIMING PULSE GENERATION

Synchronous operations in the PDP-11/60 are under the timing control of three clock pulse groups: P1, P2, and P3. A fourth pulse, UP3, a derivation of P3, defines the beginning of each microcycle (Figure 1-2). These pulses occur every 170 ns unless suspended by some machine operation (Figure 1-3).

The timing pulse generator, shown on drawing K603, is a circuit that consists of both ECL and TTL logic components (Figure 1-4). ECL components are used to achieve higher switching speeds than could be obtained from a circuit that consisted entirely of TTL components. Conversion of ECL logic levels to TTL logic levels, an example of which is shown in Figure 1-5, is accomplished by six 3639D (PNP type) transistors, Q3 through Q8. All ICs on this drawing (K603) are ECL with the exception of the inverters and gates connected from the collectors of Q3 through Q8.

P1, P2, and P3 are developed from a 60 MHz oscillator output (15 ns cycle time). The oscillator, whose principle component is a type 10216 differential amplifier, produces two square wave pulse trains: K603 CLK H and its mirror image, K603 CLK L. This output is continuous unless the oscillator enable input, K603 ENB OSCILLATOR L, is negated.

The module test gates allow the oscillator output to be inhibited and then replaced by an externally generated clock signal with a different frequency, for maintenance testing. MT ENB L, when low, inhibits K603 CK H and K603 CK L. These signals may then be replaced by MT CLK L.

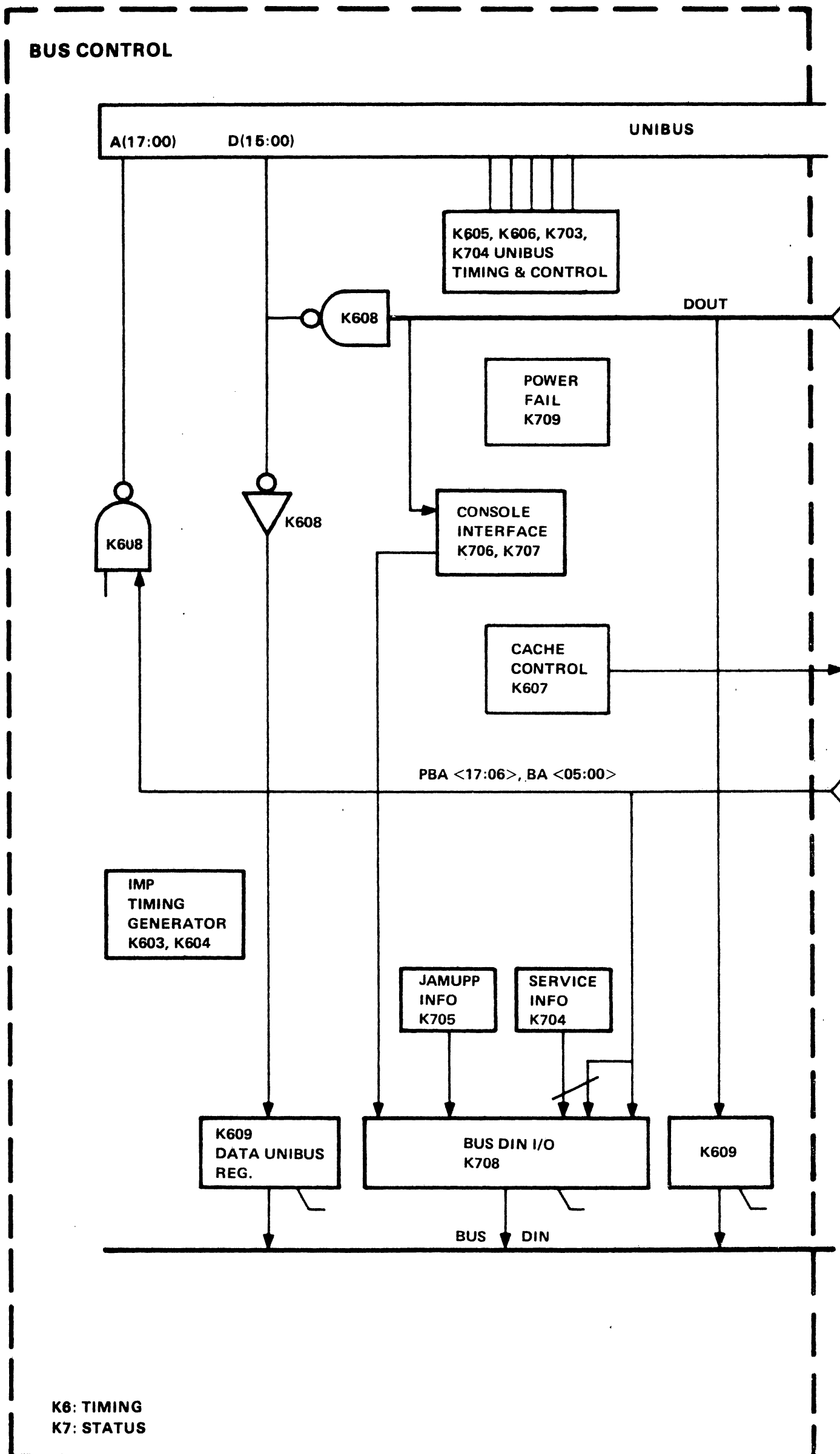


Figure 1-1 Bus Control Circuits

11-5525

Table 1-1 Bus Control Circuits

Major Circuits	Functions
Timing Generation	Produces three basic timing pulses for use throughout the base machine.
Data Transfer	Provides for data transfers (DATI, DATIP, DATO, and DATOB) to and from the Unibus and cache memory. Temporary register storage of the data.
Bus Request Arbitration	Decodes the priority of bus requests from peripheral devices on the Unibus and, based on current processor state, determines if a request is to be honored.
Interrupt Handling	Carries out the processor's part of the handshaking with the interrupting device.
Non-Processor Requests (NPRs)	Invalidates the cache location if the data at the NPR address is resident in cache and controls the granting of the NPRs.
Error and Service Handling	Decodes the conditions that cause the microprogram to go to a jam or service routine.
Console Interface	Console request control and interface. The path through which the console operator can interface with the microprogram.
Power Fail	Controls Unibus and processor actions so that power ups and power downs can occur without the loss of data.
Cache Control	Contains the logic for cache arbitration and controls the cache data flow.

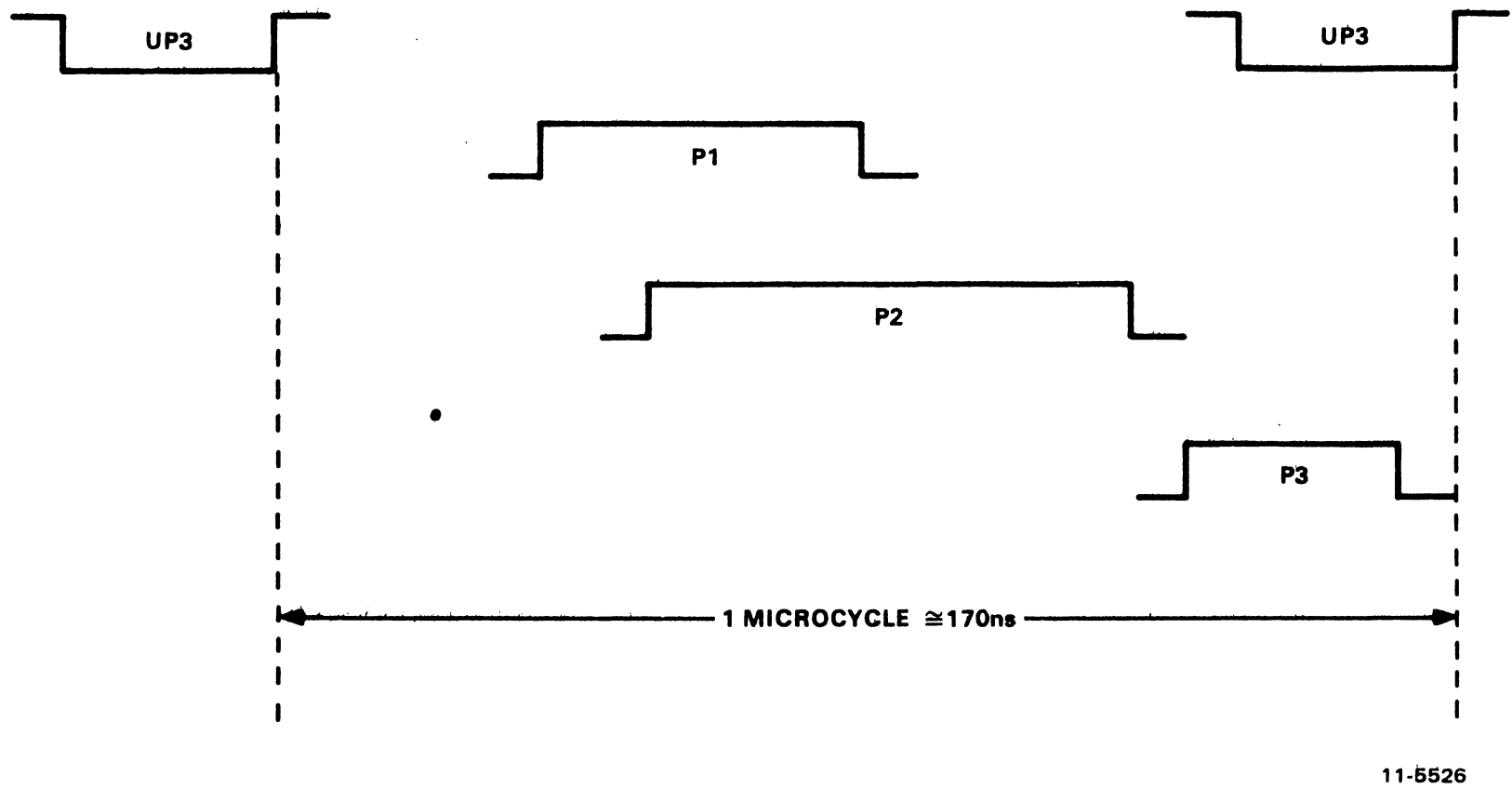
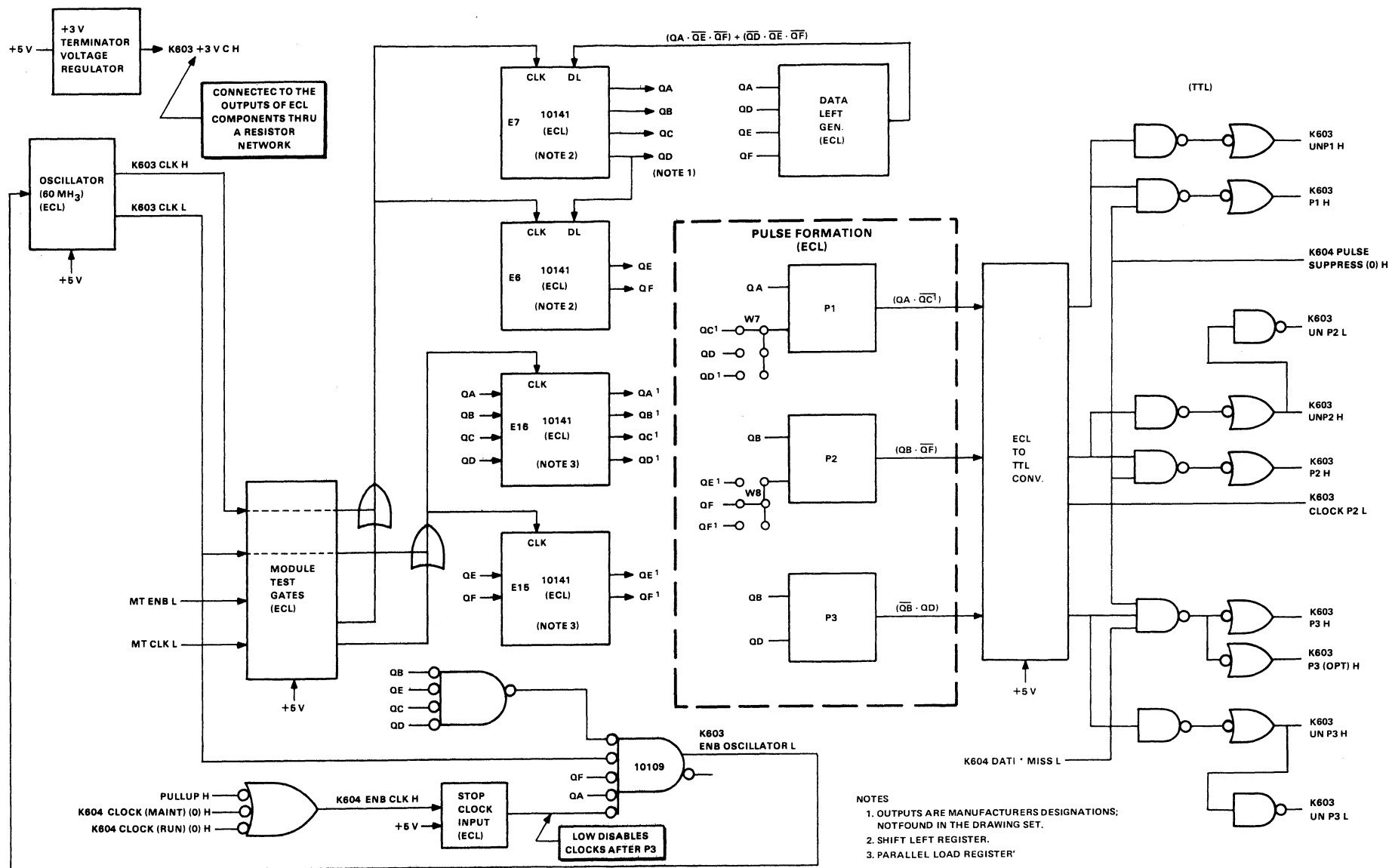


Figure 1-2 Microcycle Definition



11-5527

Figure 1-3 Clock Pulse Generation (K603)

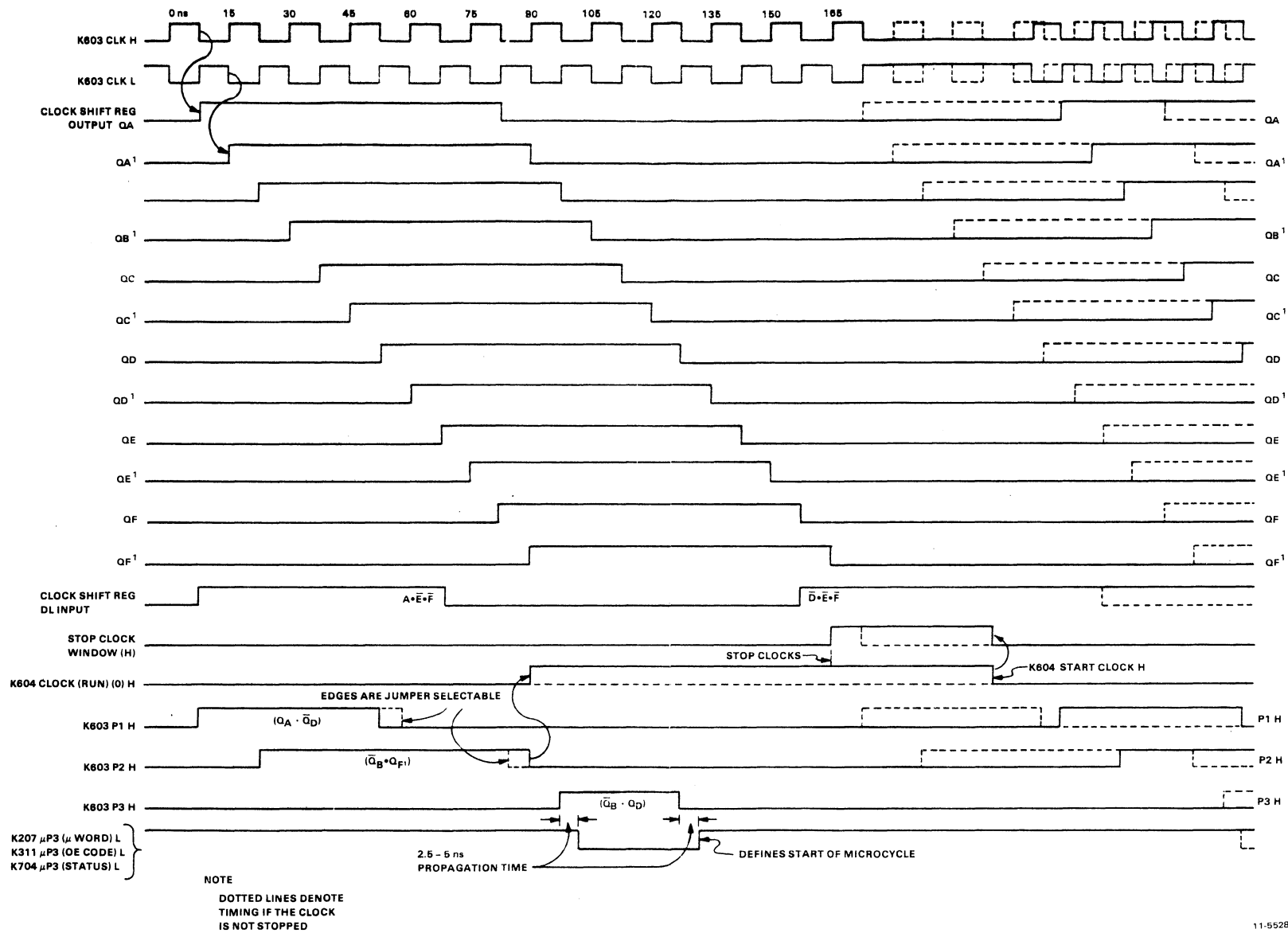


Figure 1-4 Clock Pulse Timing

The clock shift register consists of four, type 10141 shift register chips (ECL logic). The register output is a series of sequentially overlapping 7.5 ns pulses at outputs QA through QF and QA¹ through QF¹ (Figure 1-3). (Outputs QA, etc., are manufacturer's designations; they are not used in the drawing set.) The two chips with outputs QA through QF are in a permanent shift left mode, with the data input to the least significant bit (LSB) obtained from the data left (DL) generating circuit. Consequently, with the DL input high, each succeeding output (QA, QB, QC, etc.) goes high on the trailing edge of K603 CLK H, i.e., every 15 ns. The other two shift register chips, with outputs QA¹ through QF¹, are in a permanent parallel load mode (inputs are from QA through QF). The QA¹ through QF¹ outputs are clocked high on the leading edge of K603 CLK L. Therefore, the prime outputs (QA¹, QB¹, etc.) go high one half cycle (7.5 ns) after their respective inputs go high.

The clock shift register outputs serve a dual purpose: in the formation of P1, P2, and P3, and in the assertion of K603 ENB OSCILLATOR L. An example of the former is shown in Figure 1-5. QA (high), through the NOR outputs of a type 10101 OR/NOR gate, provides one input to a type 10105 OR/NOR gate (reconfigured as an AND/NAND gate). The other qualifying input to the 10105 is QD low (or QD). With these conditions (QA * QD), the high output at pin 6 of the 10105 is applied to the base of Q3; Q3 is cut off, its collector goes low, and K603 P1 H is asserted. Note that Q4 conducts at this time because pin 7 of the 10105 is low.

The trailing edge of P1 can be adjusted by connecting a different pair of W7 terminals. (An engineering test procedure exists for making adjustments to the trailing edge of P1.) W8 provides a similar fine adjustment for the trailing edge of P2.

P2 and P3 are the result of other combinations of clock shift register outputs (Figure 1-4), while UP3 is produced separately on the microword, decode, and status boards. Occurring slightly after K603 P3 H (Figure 1-2), UP3 defines the beginning of each microcycle. A new microword is clocked into the different microword registers each time UP3 is asserted. An example of this is shown in Figure 1-4. The arithmetic logic unit (ALU) field bits, BUS U (47:44) H, are clocked into a 74S175 register by K311 UP3 (DECODE) L. These four microword bits select the arithmetic or logic operation performed in the ALU on the macrodata circulating in the data path.

1.3.1 Pulse Suppression

Information transfers during intramachine operations are within and between cache, processor control, data path, and bus control. These events occur synchronously under the control of the three clock pulses. If an external (Unibus) data transfer is required, the operation is asynchronous; transactions on the Unibus are not controlled by P1, P2, and P3. At this time, these pulses are temporarily suppressed (i.e., halted) in cache, processor control, and data paths – that part of the base machine that does not participate in Unibus transfers. [An exception to this is the instruction register (IR). It is clocked at P2 time during Unibus operations.]

Clock suppression takes place when the Pulse Suppress flip-flop (drawing K604) is set. K604 PULSE SUPPRESS (0) H, low at this time, inhibits the assertion of K603 P1 H, K603 P2 H, and K603 P3 H (Figure 1-4). UP3, dependent on the assertion of K603 P3 H, is also inhibited. In addition to these pulses, there are "unsuppressed" clocks that are not gated by K604 PULSE SUPPRESS (0) H. However, these clocks (K604 UN P1 H, K604 UN P2 H, and K604 UN P3 H) are sent only to the two bus control boards and to the IR (K303).

Table 1-2 lists those signals that set and clear the Pulse Suppress flip-flop.

Table 1-2 Pulse Suppress Flip-Flop (K604)

Set By	Explanation
K 707 ALLOW BG (0) H	Set under UCON control at the end of instructions to allow the servicing of BRs.
K 607 ABORT XFER L K 606 DATA XFER (1) H	Red zone or MEM MGT error or odd address error has occurred. Set Pulse Suppress flip-flop to allow jam routine.
K 605 I/O PAGE L K 606 DATA XFER (1) H	I/O page to be accessed. I/O page not in cache; Unibus cycle necessary.
K 605 DATIP (LOCK) L K 606 DATA XFER (1) H	DATIP decoded. Followed by DATO to backing store; two Unibus cycles necessary.
K 605 DATO/DATOB (1) H K 606 DATA XFER (1) H	DATO or DATOB to backing store; Unibus cycle necessary.
K 607 CACHE*NPR L K 606 DATA XFER (1) H	Suppress clocks for an NPR DATO to see if that location is in cache and, therefore, requires invalidating.
K 604 DATI*MISS L	Cache DATI miss; must be followed by a Unibus DATI.
K 311 UBREAK (1) H	UBREAK occurs. Set Pulse Suppress to allow jam routine.
KXCS EXTRA JAM L	Parity error in WCS or RQST from DCS. Set Pulse Suppress to allow jam routine.
K 707 BUS RESET(UCON) (0) H	Invoked by reset instruction or MED Unibus INIT. Initializes the bus.
Cleared By	Explanation
K 605 SSYN DELAYED (1) H K 705 MEM PAR ERR (0) H	SSYN received from a processor Unibus cycle. No memory parity error indicated if a DATI or DATIP operation.
K 607 CACHE*DATI H K 604 HIT H K 605 I/O PAGE 1	Clear Pulse Suppress if an NPR made cache unavailable when a CACHE*DATI was attempted.
K 606 INVAL*BBSY (PROC) L	Used during cache invalidate in initialization (power up) routine. Response to a dummy DATO.
K 709 START CLK (PWR) (0) H	Power up routine completed.
K 705 START CLK (JAM) (1) H	Jam routine started.
K 704 VECTOR LOAD (1) H	Processor has responded to (viz., loaded vector) an interrupt from some Unibus device or passive release of bus, or BG SACK timeout.

1.3.2 Stop Clocks

In addition to suppressing clocks to certain parts of the PDP-11/60, clocks to the entire machine are stopped at the time the processor asserts MSYN on the Unibus or in cases of a Unibus INIT. Normal clock generation is also halted when in the single stop mode; only a single set of the three pulses is produced each time the single step switch is pressed.

If a miss occurs when a DATI is sent to cache, the processor initiated a bus cycle to obtain the data from the backing store. The SET MSYN flip-flop is set just prior to the time MSYN is placed on the Unibus. K605 SET MSYN (1) H causes the assertion of K604 STOP CLOCK L, the D input to the STOP flip-flop, which is set on the trailing edge of K603 CLOCK P2 L (an unsuppressed P2 pulse).

K604 STOP (1) H, going high, clocks the CLOCK (RUN) flip-flop. With its D input grounded, CLOCK (RUN) is cleared and K604 CLOCK (RUN) (0) H, now high, inhibits K604 ENB CLOCK H. Connected to a voltage divider that provides the input voltage for a type 10216 differential amplifier (K603), K604 ENB CLOCK H causes the amplifier output voltage to go low. This output is applied to a (reconfigured) type 10109 OR/NOR gate whose other inputs are from the clock shift register (Figure 1-4).

The 10109 gate provides a stop clock window (Figure 1-3) that occurs after the clock shift register outputs (QA through QF) are negated and K603 CLK L goes low. Then, if K604 ENB CLOCK is low, K603 ENB OSCILLATOR L goes high and the oscillator outputs, K603 CLK L and K603 CLK H, are halted. Consequently, even though CLOCK (RUN) is negated at the trailing edge of P2, the clock continues to run until after the completion of P3 and the next microword is read out by UP3.

The clocks remain halted until K604 START CLOCK H sets the CLOCK (RUN) flip-flop and terminates the window. This signal is asserted when SSYN is returned in response to MSYN. (If SSYN is not received K705 SSYN TIMEOUT (0) H will start the clocks.) On power up, K709 START CLK (PWR) (0) H initiates the clock signals.

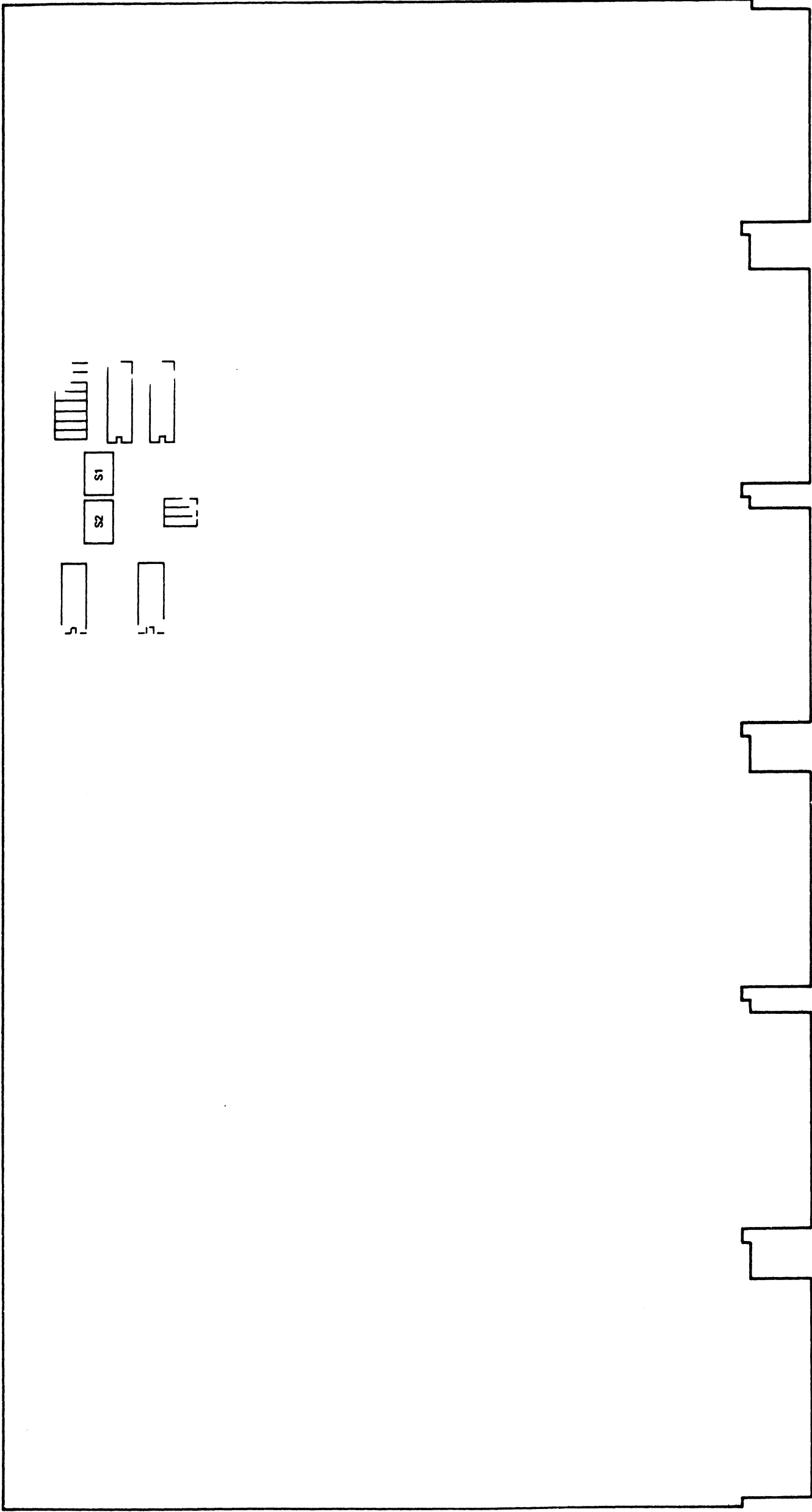
In DATO/DATOB and DATIP operations, the clocks are stopped when K606 BBSY (PROC) (1) H is asserted; a cache miss will also stop the clock.

1.3.3 Single Clock

The single step switches allow the operator to single step one microword at a time. Two switches, S1 and S2 on the M7876 module, provide the means for this (Figure 1-6 and drawing K604). Pushing the single step enable switch (S2) down stops normal clock generation (after the second P3 pulse). The clock is now in the single step mode. A single set of pulses (P1, P2, and P3) is produced each time the single step switch (S1) is flipped up and down. Figure 1-7 is a flow diagram of this operation. Return to normal clock generation can be effected by placing S2 in the up position and then flipping S1 up and down twice.

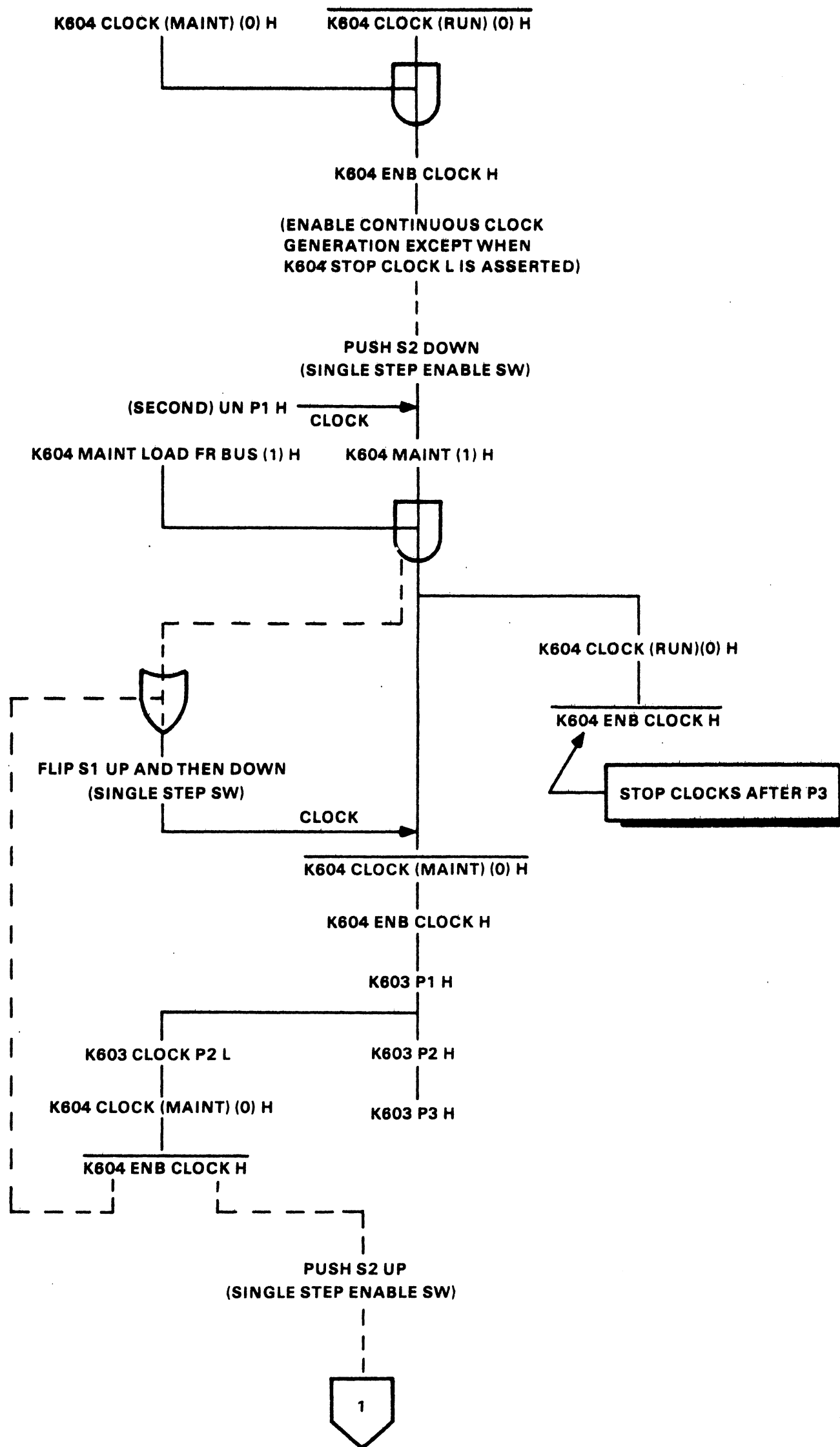
The jam routine can use the single clock if the console HALT/SI switch is pressed after S2 is pushed down (Figure 1-8). Pressing HALT/SI asserts K709 INIT A H and the Stop flip-flop is set at P2 time. (However, the clocks were previously stopped as the result of pushing S2 down.)

The initial address of the jam routine is established in the BUT MUX and then the clock is restarted at the moment HALT/SI is released. Because of the two flip-flops in the single clock circuit (K604) – they act as a synchronizer – two sets of clocks are produced before the clocks are stopped. Consequently, the address of the third microword in the jam routine is displayed at the next microbranch address (NUA) indicators when the clocks stop. (This is a special routine when powering up in the single step routine and means that you do not see the jam address, 777, in NUA because two cycles were executed).



11-5530

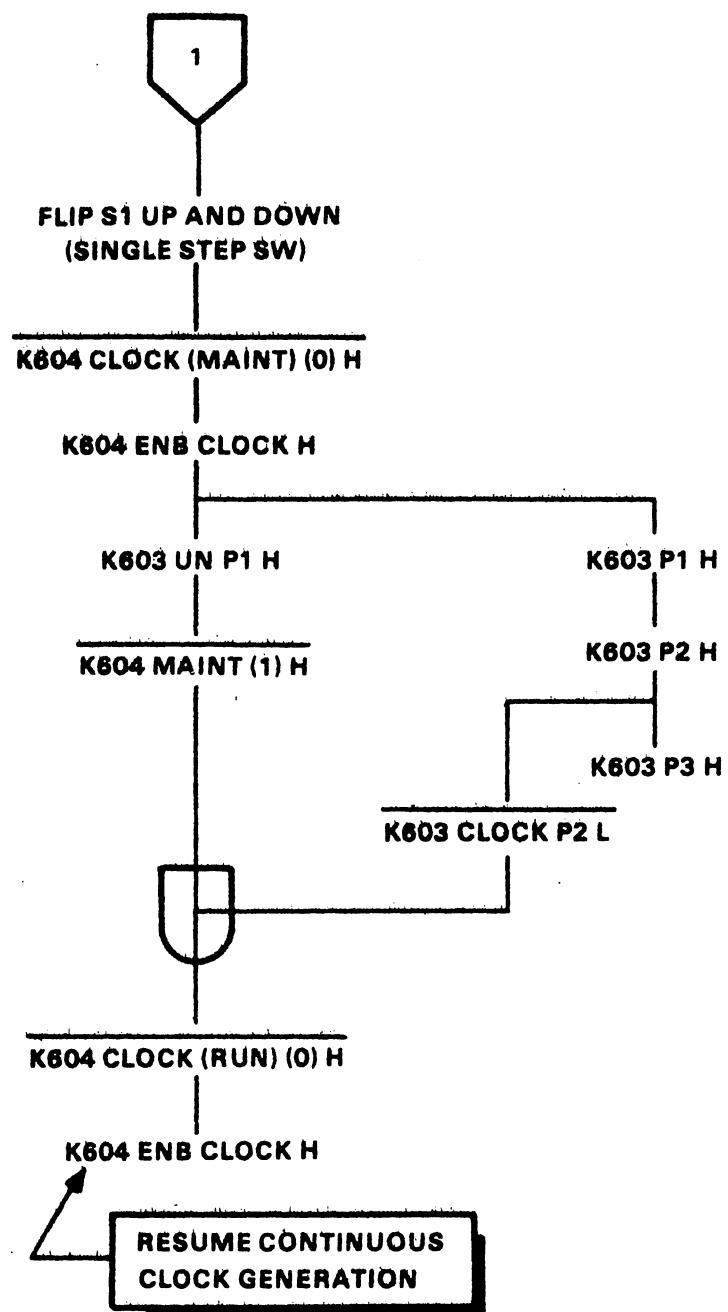
Figure 1-6 M7876 Board Showing S1 and S2 Placement



MA-0463

Figure 1-7 Single Step Clock Operation (Sheet 1 of 2)

19



MA-0464

Figure 1-7 Single Step Clock Operation (Sheet 2 of 2)

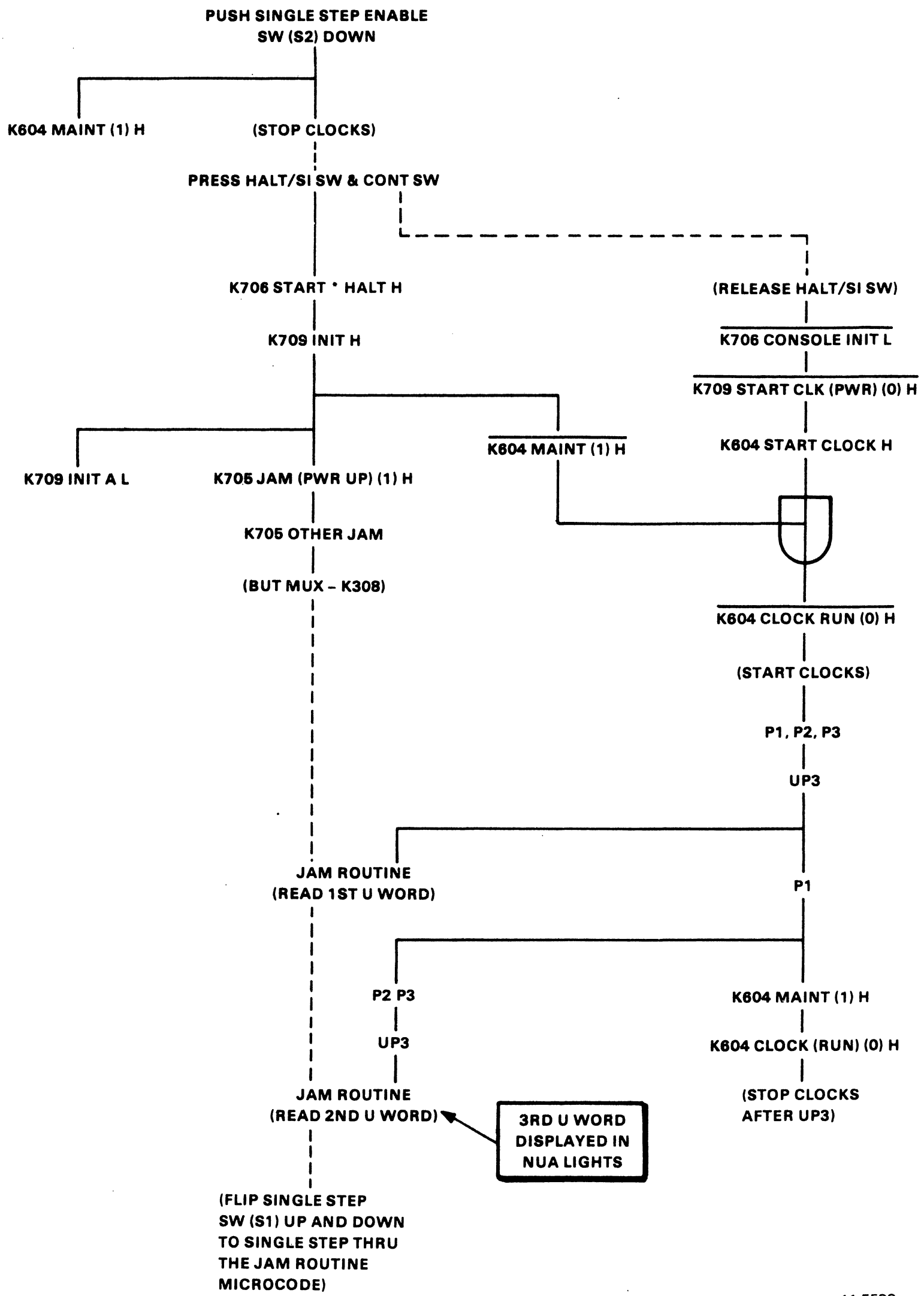


Figure 1-8 HALT/SI Switch (Single Step Jam Routine)

1.4 INTERNAL ADDRESS REGISTERS

Registers that are physically located in the base machine, or WCS option, which have Unibus addresses even though they are not connected to the Unibus, are called internal address registers. The addresses are called internal addresses and are located in the I/O page (Paragraph 1.5). Table 1-3 lists these registers and their addresses. (As noted, two of the registers do not actually exist; the information at the Unibus address is obtained from various logs in the A scratchpad.)

The circuit, shown on drawing K605, consists of two type 74187 ROM chips and several gates connected to the ROM outputs. The outputs, M0 (1) and M1 (1), are asserted (low) when an internal address configuration is present at the inputs to the two ROMs. [Both ROMs assert a low at M0 (1) or M1 (1) at the same time if an internal address is detected.]

20A) K605 INTERNAL ADRS L, asserted if either ROM output M0 (1) or M1 (1) go low, inhibits K606 BUS RQST (PROC) (1) H and K606 BBSY (PROC) (1) H because a Unibus cycle is not required for a read or write to an internal address. A parallel signal, K605 INTERNAL ADRS H, is clocked into a register on the status module at UN P3 time if K606 DATA XFER (1) H is asserted to assert K705 INTERNAL ADRS (1) H. This latter signal will cause a jam (K705 JAM L) and the jam routine is entered. Concurrent with this, a microcode entry point address is developed from the internal Unibus address and the type of bus operation (Table 1-4). Three ROM chips, shown on drawing K507, are used in the process. One of several entry addresses is asserted on the DIN lines when an internal address is detected; the bus operation determines the specific address. A DATI (absence of a DATO or DATOB) causes a base microcode entry point to be placed on BUS DIN; a DATO or DATOB to the same internal register will assert a modification of this address (one, two, or three locations higher). From BUS DIN the address is routed through the DMUX, CSP, and D register before it is loaded into the return register for transfer to the basic microaddress (BUA) and NUA registers (Figure 4-2).

The internal register is now accessed and either read or written to. If read, the data at the internal address is routed to the C scratchpad via the DMUX; if written to, the data is delivered to the internal address over one of several paths. For example, the page address and page description (PAR and PDR) registers receive data from the data path D register via the DOUT lines to the memory management logic on the cache module. On the other hand, console key data travels the route from the console interface on the status module via the BUS DIN lines, DMUX, and C scratchpad, to the A scratchpad.

The internal address ROMs are enabled by UCON bit 12 (U35) ANDed with K209 UCON KT (1) H and K607 INH UCON (BUS XFER) L (Figure 2-9).

1.5 I/O PAGE EXPANSION

The top 4K word locations (124-128K) in Unibus address space are called the I/O page. These addresses are reserved for internal registers (Paragraph 1.4), peripheral device registers, and for customer designated purposes. The I/O page can be expanded, in 4K increments, to a 32K block of word locations (96-128K) through the selective removal of jumpers on the timing module.

Six jumpers, W1 through W6 (shown on drawings K605 and K607), control the size of the I/O page (Figure 1-9). The module is supplied with the jumpers installed to provide a 4K I/O page (124-128K or 760000-777777₈). The jumpers can be cut as indicated to move the lower boundary of the I/O page downward to 96K (600000₈).

If the processor is not in the console mode or address relocation is not specified by memory management, the signal K605 I/O PAGE (and inverted variations) is asserted when an address in the I/O page is indicated. Note that this is a physical address which, when not relocating, is synonymous with the bus address (BA). The functions of K605 I/O PAGE H are listed in Figure 1-9. Under the same conditions, BA bits 16 and 17 are forced high when K606 BBSY H is asserted; a lower address is changed to the highest possible Unibus address.

Table 1-3 Internal Address Selection (58 and E49)

Register	Address (Note 2)	Physical Location (Module and Drawing No.)	Input															Output	
			E58								E49							(Asserted Low)	
			K503 PBA XXH								K503 PBA XXH							E58	E49
15 (A0)	14 (A1)	13 (A2)	12 (A3)	11 (A4)	10 (A5)	9 (A6)	8 (A7)	08 (A0)	07 (A1)	06 (A2)	05 (A3)	04 (A4)	03 (A5)	02 (A6)	01 (A7)	E58	E49		
PSW	777776	{ U WORD, K208 DECODE, K306	1	1	1	1	1	1	1	1	1	1	1	1	1	1	M1(1)	M1(1)	
STACK LIMIT	777774		CACHE, K506	1	1	1	1	1	1	1	1	1	1	1	1	1	0	M1(1)	M1(1)
U BREAK	777770	DECODE, K311	1	1	1	1	1	1	1	1	1	1	1	1	0	0	M1(1)	M1(1)	
CPU ERROR	777766	(NOTE 1)	1	1	1	1	1	1	1	1	1	1	1	0	1	1	M1(1)	M1(1)	
HIT/TAG	777752	CACHE, K514	1	1	1	1	1	1	1	1	1	1	0	1	0	1	M1(1)	M1(1)	
CACHE CONTROL	777746	CACHE, K514	1	1	1	1	1	1	1	1	1	1	0	0	1	1	M1(1)	M1(1)	
MEMORY ERROR	777744	(NOTE 2)	1	1	1	1	1	1	1	1	1	1	0	0	1	0	M1(1)	M1(1)	
{ USER PAR	777656	CACHE, K503	1	1	1	1	1	1	1	1	1	0	1	0	1	1	M1(1)	M1(1)	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	0	777640	1	1	1	1	1	1	1	1	1	0	1	0	0	0	M1(1)	M1(1)	
{ USER PDR	777616	CACHE, K504	1	1	1	1	1	1	1	1	1	0	0	0	1	1	M1(1)	M1(1)	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	0	777600	1	1	1	1	1	1	1	1	1	0	0	0	0	0	M1(1)	M1(1)	
MMR2	777576	CACHE, K505	1	1	1	1	1	1	1	1	0	1	1	1	1	1	M1(1)	M1(1)	
MMR1	777574	(NOTE 3)	1	1	1	1	1	1	1	1	0	1	1	1	1	0	M1(1)	M1(1)	
MMR0	777572	CACHE, K505	1	1	1	1	1	1	1	1	0	1	1	1	1	0	M1(1)	M1(1)	
SW/DISPLAY	777570	DATA PATH ASPHI LOC 6, K406	1	1	1	1	1	1	1	1	0	1	1	1	1	0	M1(1)	M1(1)	
WCS DATA	777544	PSEUDO REGISTER - ACTUALLY THE WCS DATA ARRAY	1	1	1	1	1	1	1	0	1	1	0	0	1	0	M1(1)	M1(1)	
WCS ADDRESS	777542	WCS, K103, ALSO COPY IN ASPHI LOC 21, K406	1	1	1	1	1	1	1	0	1	1	0	0	0	1	M1(1)	M1(1)	
WCS STATUS	777540	WCS, K103, K109	1	1	1	1	1	1	1	1	0	1	1	0	0	0	M1(1)	M1(1)	
{ KERNEL PAR	772356	CACHE, K503	1	1	1	1	0	1	0	0	1	1	1	0	1	1	M0(1)	M0(1)	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	0	772340	1	1	1	1	0	1	0	0	1	1	1	0	0	0	M0(1)	M0(1)	
{ KERNEL PDR	772316	CACHE, K504	1	1	1	1	0	1	0	0	1	1	0	0	1	1	M0(1)	M0(1)	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	0	772300	1	1	1	1	0	1	0	0	1	1	0	0	0	0	M0(1)	M0(1)	

Notes: 1. Does not physically exist; address retained for system software compatibility. Data at the address obtained from log information.

2. 15-bit address except in console mode or when relocating which require an 18-bit address.

3. Not implemented, but will always read 0 when accessed.

Table 1-4 Internal Address (E59, E70, E77)

Register	Address (ROM Input)	Convert DATOB to DATO	Microcode Entry Point Address [ROM Output – BUS DIN (14:00) H]			
			DATI	DATO	DATOB (Even Bytes)	DATOB (Odd Bytes)
PSW	777776	NO	3030	3031	3032	3033
STACK LIMIT	777774	YES	3040	3041	3041	3041
U BREAK	777770	NO	3060	3061	3062	3063
CPU ERROR	777766	YES	3070	3071	3071	3071
HIT/TAG	777752	YES	3050	3051	3051	3051
CACHE CONTROL	777746	YES	3134	3135	3135	3135
MEMORY ERROR	777744	YES	3104	3105	3105	3105
USER PAR {	777656	NO	3120	3121	3122	3123
	.	.				
	.	.				
USER PDR {	777640	NO	3130	3131	3132	3133
	777616	NO				
	.	.				
MMR2 {	777600	NO	3014	3015	3015	3015
	777576	YES				
	777574	YES				
MMR1	777572	NO	3000	3001	3002	3003
SW/DISPLAY	777570	YES	3100	3101	3101	3101
WCS DATA	777544	YES	3124	3125	3125	3125
WCS ADDRESS	777542	YES	3110	3111	3111	3111
WCS STATUS	777540	YES	3114	3115	3115	3115
KERNEL PAR {	772356	NO	3120	3121	3122	3123
	.	.				
	.	.				
KERNEL PDR {	772340	NO	3130	3131	3132	3133
	772316	NO				
	.	.				
	772300	NO				

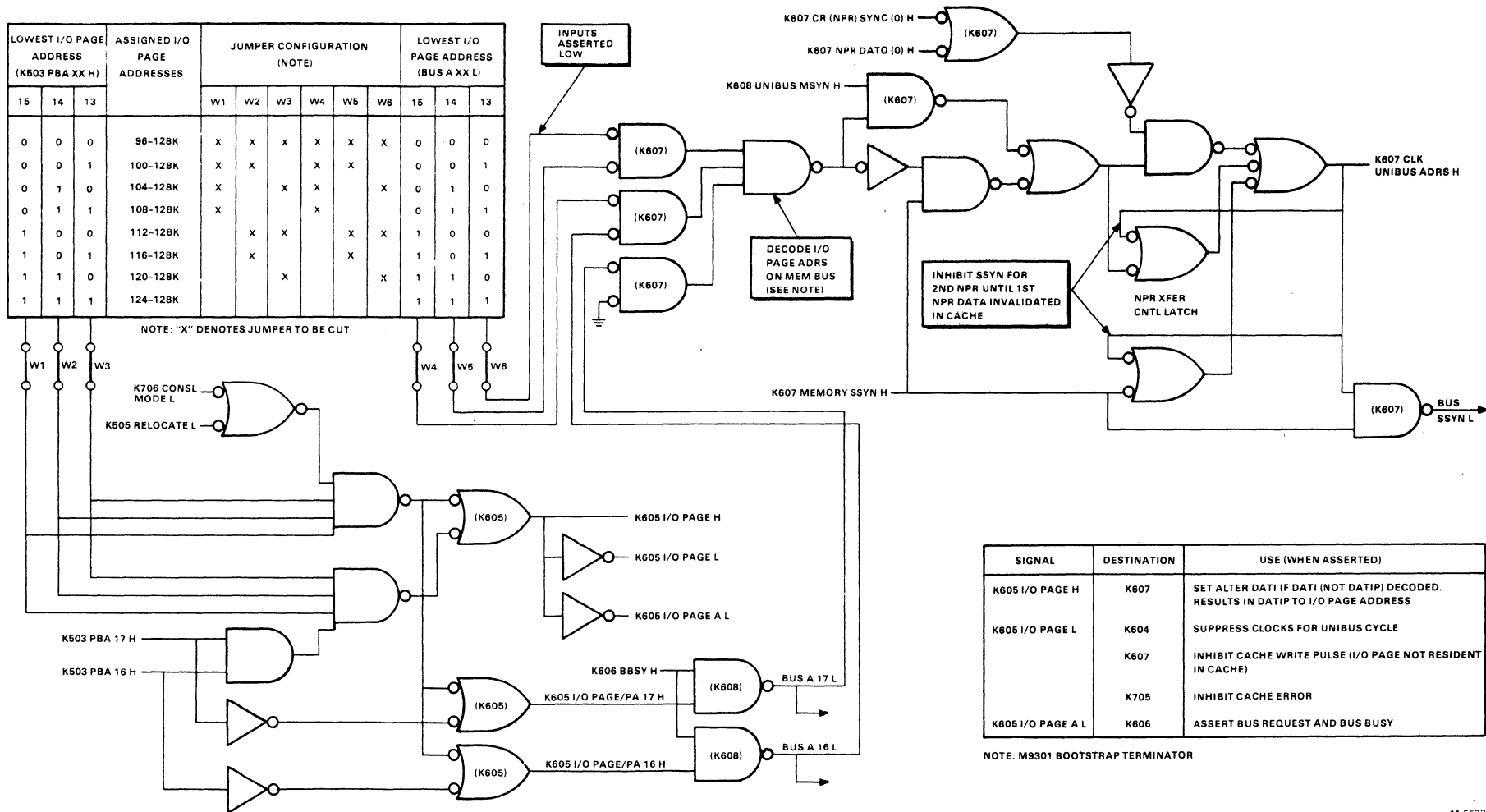


Figure 1-9 I/O Page Expansion,
NPR Transfer Control Latch

However, if either K706 CONSL MODE L or K505 RELOCATE L is asserted, physical address bits 16 and 17 must both equal 1 in order to assert K605 I/O PAGE H. (During relocation, the condition of bits 16 and 17 is determined by the states of PAR bits 10 and 11 in the cache module.)

1.6 DATA TRANSFERS

The execution of macroinstructions requires the exchange of information between the processor, cache memory, and memory devices on the Unibus. The processor controls the procedure, arbitrating requests to use the Unibus and, through the use of bus control signals, monitors the purpose of bus operations.

Two general types of bus operations can take place: read and write. In a read, or data in (DATI), information is brought in to the processor control and/or data path from the cache (if it is resident there), or from memory, or some other device, on the Unibus. In a write, or data out (DATO), data is transferred from the data paths to a Unibus memory device and to the cache. (Exceptions occur when the I/O page, which occupies the upper 4K of available memory space, is addressed.)

The different types of read and write operations that can be performed in the PDP-11/60 are listed in Table 1-5. These operations, initiated by bus commands (described in the following paragraph), require a number of microcycles for completion. The bus command, defined in the first microword, initiates the bus operation; the data transfer takes place during several subsequent microcycles. For example, DATI [K605 DATI (1) H] does not actually cause the MD register to be loaded; this is done explicitly by microcode in the microcycle following the DATI. DATI only invokes the bus control and causes the transfer of data into the DMUX, either from the cache or the Unibus data buffer.

1.6.1 Bus Command Generation

Read and write data transfers are initiated by bus commands that originate in the timing module. The commands are generated by a type 6331 ROM (E44), shown on drawing K605. The ROM inputs and outputs are listed in Table 1-6.

Along with the ROM, the bus command circuit contains a flip-flop register (one 74H74 and one 74S174) that is clocked at P3 time. The commands, which correspond to the types of data transfers listed in Table 1-5, pass through the register to the bus control logic. One exception is a ROM output to the D input of the JAM INT ADRS flip-flop. A low output from the ROM allows the assertion of K705 ILLEGAL ADRS (1) H if an internal address is detected. This causes the assertion of K705 JAM L and entry of the jam routine (Paragraph 1.7.3). Note that K605 INTERNAL ADRS H causes JAM L for internal address manipulation, but K705 ILLEGAL ADRS (1) H is a jam error condition.

K605 DATI (ENB DATIP) H, in addition to being clocked into the 74H74, is ANDed with K606 BEGIN DATA XFER H to generate a cache request. K605 DATI (ENB DATIP) H, as indicated in Table 1-6, is produced each time a DATI, DATI (CLKIR), or DATI (ODD ADRS) is decoded. Consequently, a cache request is generated for all DATI, DATI (CLK IR), and DATI (ODD ADRS) operations.

The bus command circuit, as it applies to DATI (CLK IR) and DATI (ENB DATIP), is shown in Figure 4-9. Inputs to the ROM are three of the bus control field bits, (22:20), and K303 BYTE (1) H, which are asserted when a byte instruction is decoded. Microword bit 20 is gated with K306 PREFETCH H. The latter signal, high when a macrofetch is performed and the next instruction is one of the types listed in Table 4-5, converts a DATI to a DATI (CLK IR). This allows an instruction prefetch to be carried out (Paragraph 4.7).

Table 1-5 Read and Write Data Transfers

Type	Operation
DATI	Transfer one 16-bit instruction/data word from the cache, if resident, or from the backing store on the Unibus if a cache miss, to the MD register in the data path C scratchpad (CSP). A cache miss causes the Unibus data to be written into the appropriate cache location.
DATI (CLK IR)	Transfer one 16-bit instruction word from the cache, if resident, or from the backing store if a cache miss, to the IR in processor control. Unibus data is written to the cache if a miss occurs. DATI (CLK IR) is performed in conjunction with DATI or DATI (ENB DATIP). An instruction is loaded into both the IR and the MD register. If DATI (ENB DATIP) is also asserted, the contents of an I/O page address can be clocked into the IR by a DATIP operation [see DATI (ENB DATIP) below].
DATI (ODD ADRS)	Same as a DATI except that an odd cache or backing store address can be accessed. An I/O page address can also be read.
DATI (ENB DATIP)	Generates a cache request. Asserted concurrently with DATI, DATI (CLK IR), or DATI (ODD ADRS). Sets the Alter DATI flip-flop, which causes a DATIP operation, if an I/O page address is specified. Transfers one 16-bit word from the I/O page address to the MD register and to the IR if DATI (CLK IR) is asserted. Transfers the high byte at the I/O page address to the MD register if DATI (ODD ADRS) is asserted. [In all cases, K311 ALLOW DATIP (1) H, a microcontrol bit, must also be asserted to generate K607 ALTER DATI (1) H to force a DATIP.] After the DATIP data transfer, BUS BUSY remains asserted, which prevents other Unibus devices from transferring data, until a DATO or DATOB to the same I/O page address is completed. (BUS BUSY is cleared automatically as a function of the DATO/B sequence. BUS BUSY may also be cleared by BUT (LAST).
DATIP (LOCK) (called DATIBP in the microcode)	Asserted only in the microcode flow for an ASRB instruction. Transfers one 16-bit instruction/data word from backing store, or an I/O page address, to the MD register (odd or even addresses are allowed); a cache cycle is not performed. BUS BUSY remains asserted after the data transfer, preventing data transfers by other devices on the Unibus until a DATO or DATOB to the same address is completed. (BUS BUSY is cleared automatically as a function of the DATO/B sequence. BUS BUSY may also be cleared by BUT (LAST). DATIP (LOCK) inhibits the assertion of K607 ODD ADRS ERR L.
DATO	Transfer one 16-bit word from the data (D) register in the data path to both backing store and cache. Data can be stored only at even addresses.
DATOB	Transfer one 8-bit byte from the D register to backing store and, if a hit, to cache. Odd or even addresses are allowed.

Table 1-6 Bus Command ROM (E44)

BUS CONTROL FIELD (BUS UXX H)															
24	23	22	21	20											
ROM INPUT (NOTES 1 AND 2)						ROM OUTPUT (NOTE 3)									
	(A4) K307 BYTE(1)H	(A3) K207 BC0/PREFETCH H	(A2) K409 BC1(1)H	(A1) K409 BC2(1)H	(A0) (Ground)	(M7) (JAM INT ADRS)	(M6) K605 DATI(ENB DATIP)H	(M5) K605 DATI(1)H	(M4) K605 DATIP(LOCK)(1)H	(M3) K605 DATO(1)H	(M2) K605 DATI(CLK IR)(1)H	(M1) K605 DATI(ODD ADRS)(1)H	(M0) K605 DATOB(1)H		
1 0	0	0	0	0	0	0	1	0	0	0	1	0	0	Produced by word instructions	
1 0	0	0	0	1	0	1	0	0	1	0	0	0	0		
1 0	0	0	1	0	0	1	0	0	0	1	0	0	0		
1 0	0	0	1	1	0	1	1	1	0	0	0	0	0		
1 0	0	1	0	0	0	0	1	1	0	0	0	0	0		
1 0	0	1	1	0	0	1	1	1	0	0	0	0	0		
1 0	0	1	1	1	0	1	0	0	0	1	0	0	0		
1 0	1	0	0	0	0	0	1	0	0	0	1	0	0	Produced by byte instructions	
1 0	1	0	0	1	0	1	0	0	1	0	0	0	0		
1 0	1	0	1	0	0	1	0	0	0	1	0	0	0		
1 0	1	0	1	1	0	1	1	1	0	0	0	0	0		
1 0	1	1	0	0	0	0	1	1	0	0	0	0	0		
1 0	1	1	0	1	0	1	0	0	0	0	0	0	1		
1 0	1	1	1	0	0	1	1	0	0	0	0	1	0		
1 0	1	1	1	1	0	1	0	0	0	1	0	0	0		

- NOTES:
1. All outputs are low (0) if the A0 input is not grounded.
 2. K306 PREFETCH H forces the A3 input low; changes DATI to DATI (CLK IR).
 3. All outputs are clocked by K605 BEGIN DATA XFER H (K306 INH SPLO L· K604 PULSE SUPPRESS (0) H. K405 begin (BUS/UCON)(1) H. The latter signal is asserted by BUS U24 = 1 and BUS U23 = 0.) K605 DATI (ENB DATIP H (M6) also has a nonclocked output.

The bus commands are clocked by K606 BEGIN DATA XFER H at P3 time. This signal is high if K306 INH SPLO L, K604 PULSE SUPPRESS (0) H, and K405 BEGIN (BUS/UCON) (1) H are all high (Figure 4-9). K306 INH SPLO L is high unless K306 OVERLAP L is negated during a macrofetch (Figure 4-21), i.e., a fetch overlap is not allowed. K604 PULSE SUPPRESS (0) H is high until (unless) the Pulse Suppress flip-flop is set by K604 DATI*MISS L due to a cache miss. This would occur after the command is clocked. K405 BEGIN (BUS/UCON) H is high when microword bits 23=0 and 24=1. These bits must be set to this configuration in the first microinstruction of every bus command microroutine.

1.6.2 Bus Command Flow Diagrams

The bus commands listed in Table 1-5 use a large portion of the logic contained on the timing and status modules. The logic events that occur during the execution of these commands are described in the following paragraphs; Figures 1-10 through 1-12 illustrate the sequential events that occur during the different command cycles. Individual steps itemize signals or conditions that are necessary for the completion of the entire operation. Solid lines connect signals with their direct logical consequences; dashed lines connect other events in their chronological order, e.g., MSYN is followed by SSYN on the bus.

Signals are shown in their asserted or negated states; either (or both) the 1 or 0 outputs of flip-flops are shown [e.g., K606 BBSY (PROC) (1) H and K606 BBSY (PROC) (0) H indicate the flip-flop in the set state]. AND and OR symbols are shown, where necessary, to emphasize the requisites for certain signals or events (inputs are to the top and sides of a symbol; the result of the function is from the bottom of a symbol). Off-page connectors provide extra-figure continuity. The time frame for each command is divided into microcycles, e.g., U CYCLE 1, U CYCLE 2, etc., that are necessary for the completion of the operation. (A microcycle is that period between the trailing edges of two successive UP3 pulses.) Other periods, such as time waiting for SSYN to be returned, are also shown.

1.6.3 DATI Flow

The DATI flow (Figure 1-10) depicts the sequence of events in the bus control logic for a DATI, DATI (CLK IR), or DATI (ODD ADRS) operation. The differences between the three are described in Table 1-5.

Early in U CYCLE 1 (e.g., FET01, P1 time) the macroaddress in the PC is clocked into the BA register if the microword has bit 26=1. Part of the address is transferred to the cache ALU (where it may be relocated) to generate the physical address. The physical address bits, together with the five low-order BA register bits, produce K509 PA MUX (17:01) L. The seven high-order bits, K509 PA MUX (17:11) L, are then compared with the cache tag, K511 CACHE ADRS (17:11) L (Paragraph 2.2.3.1). K514 HIT L is asserted if they compare, i.e., cache contains the desired data. If there is no comparison, or if there is bad (odd) parity detected in the tag or data stores, K514 HIT L is inhibited, i.e., a cache miss. (If the valid bit=0, K511 TAG PAR OK H is negated to indicate a cache miss.) Concurrent with this, the selected cache data store location [K509 PA MUX (10:01) L] is read and the data [K512/K513 CACHE DATA (15:00) H] is present at one of the DMUX input ports.

Either the DATI, DATI (CLK IR), or DATI (ODD ADRS) command is latched by the UN P3 pulse (at the end of the microcycle) providing the same microword has asserted K405 BEGIN (BUS/UCON) H (BUS U23=0, BUS U24=1). This signal, together with K409 ENB CLK BA (1) H, is defined at UP3 time, i.e., at the very beginning of the microcycle.

If OK DATI (CLK IR) is latched, the MD receives the data because it is specified in the microcode (MD-DATA in FET02, for example; refer to Figure 1-10). In addition, the IR also receives the data if DATI (CLK IR) is decoded. When DATI (ODD ADRS) is decoded, K607 ODD ADRS ERR L is inhibited to allow an odd address to be accessed. A 16-bit word is loaded into the MD register.

K605 DATI (ENB DATIP) H, asserted concurrent with one of the three DATI commands, generates a cache request [K607 CACHE RQST (DATI) (1) H]. Consequently, regardless of the type of DATI performed, a read operation to cache is initiated; the bus command simply defines the nature of the DATI.

K607 CACHE*DATI L, asserted when the cache request is made, generates K607 ENB DMUX FR CACHE L to open the cache port to the DMUX; the DMUX receives the data from the selected location. In U CYCLE 2, if there is no cache miss (K604 DATI*MISS L), K603 P3H is not inhibited and the data is clocked into the MD register from the DMUX. Macroinstructions, in addition to going to the MD register, are also clocked into the IR; this occurs first, at P2 time.

A cache miss, K514 HIT L, has several consequences: K604 DATI*MISS L inhibits K603 P3 H and the DMUX data is not clocked into the MD register, the Pulse Suppress and Cache Miss flip-flops are set, and a processor bus request [K606 BUS RQST (PROC) (1) H] is generated. All these events occur in U CYCLE 2. Also at this time, K607 ENB DMUX FR CACHE L is negated to open the DMUX Unibus port in anticipation of the data from the backing store.

K606 BBSY is asserted in U CYCLE 3 and BUS BUSY, the control bits, and the physical address are placed on the Unibus lines. K606 BBSY (PROC) (1) H, asserted at P3 time, and K607 CACHE MISS (0) H (low) set the Set MSYN flip-flop at P2 time in U CYCLE 4. Then, at P3 time, the Start MSYN flip-flop is set and all clock pulses are stopped for the duration of the forthcoming bus cycle. (U CYCLE 4 provides the time required to deskew the address and control lines after they are asserted on the Unibus in U CYCLE 3.)

After a short additional delay, MSYN is asserted on the Unibus to request the DATI operation in the backing store. There is a wait at this point until the DATI is performed and the data is placed on the Unibus D lines. The backing store signifies when this is completed by asserting BUS SSYN L.

Processor operations are resumed on receipt of SSYN: K605 START CLOCK (SSYN) H restarts the production of internal clock pulses (except for suppressed clocks) beginning with UN P1. This marks the start of U CYCLE 5. K704 DATA LOAD (1) H, also asserted when SSYN is received, enables the data (Unibus) register output to BUS DIN (K704 ENB DIN FR DU L). Then, at P3 time, the SSYN Delayed flip-flop is set and K704 CLOCK DU H clocks the 16-bit read data from the Unibus into the data (Unibus) register. From here the data is placed on the DIN lines and routed to the DMUX where it is called K515 DMUX (15:00) H.

K605 SSYN DELAYED (1) H also causes the assertion of another signal in U CYCLE 5: K605 ENB CLK IR L. This signal was previously asserted in U CYCLE 1 to clock the cache data into the IR. It is asserted again in U CYCLE 5 to clock Unibus data into the IR (U CYCLE 6). (Since there was a cache miss, the data clocked in from the cache is irrelevant; this is overwritten with the correct information when the Unibus data is loaded.) Figure 4-11 shows the pertinent timing for clocking the IR.

If a parity error is detected when the Unibus DATI is performed, the backing store asserts BUS PA L and BUS PB L at the same time the data is placed on the Unibus D lines. This configuration of PA and PB is clocked by K605 SSYN DELAYED (1) H to set the MEM PAR ERR flip-flop shown on drawing K705. K705 MEM PAR ERR (0) H, now low, asserts K705 OTHER JAM H and K705 JAM L. The latter signal causes entry into the microcode jam routine (Paragraph 1.7.3).

In U CYCLE 6 the IR is loaded at P2 time and the MD register at P3 time. Also at P2 time, the Pulse Suppress flip-flop is cleared and the Cache R/W ENB flip-flop is set; K607 CACHE R/W ENB (1) H causes the generation of the cache write pulse (K607 CACHE WR PULSE H) and the data from the backing store is written in the cache. At P3 time, the cache write pulse is terminated and BUS BUSY is reset.

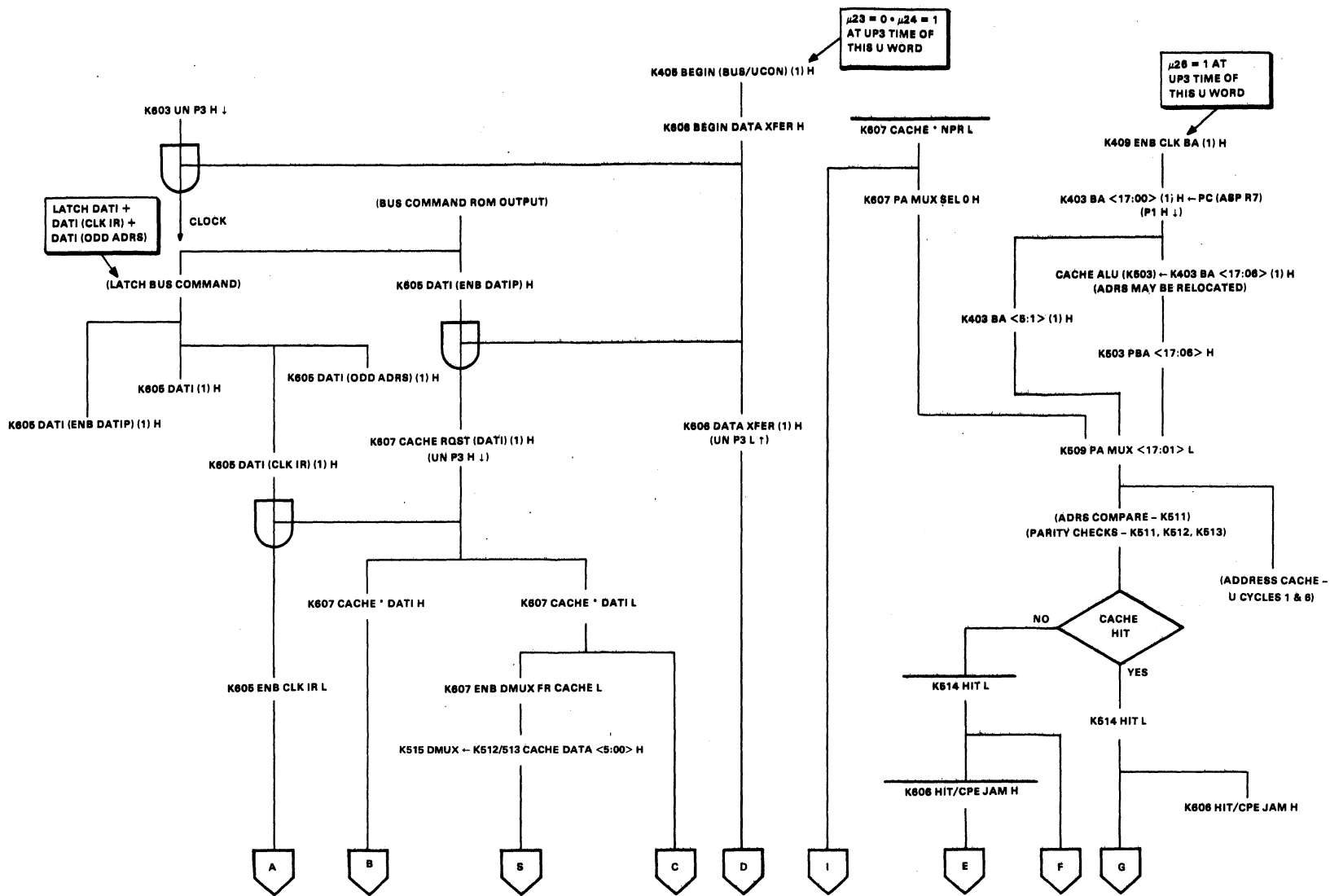
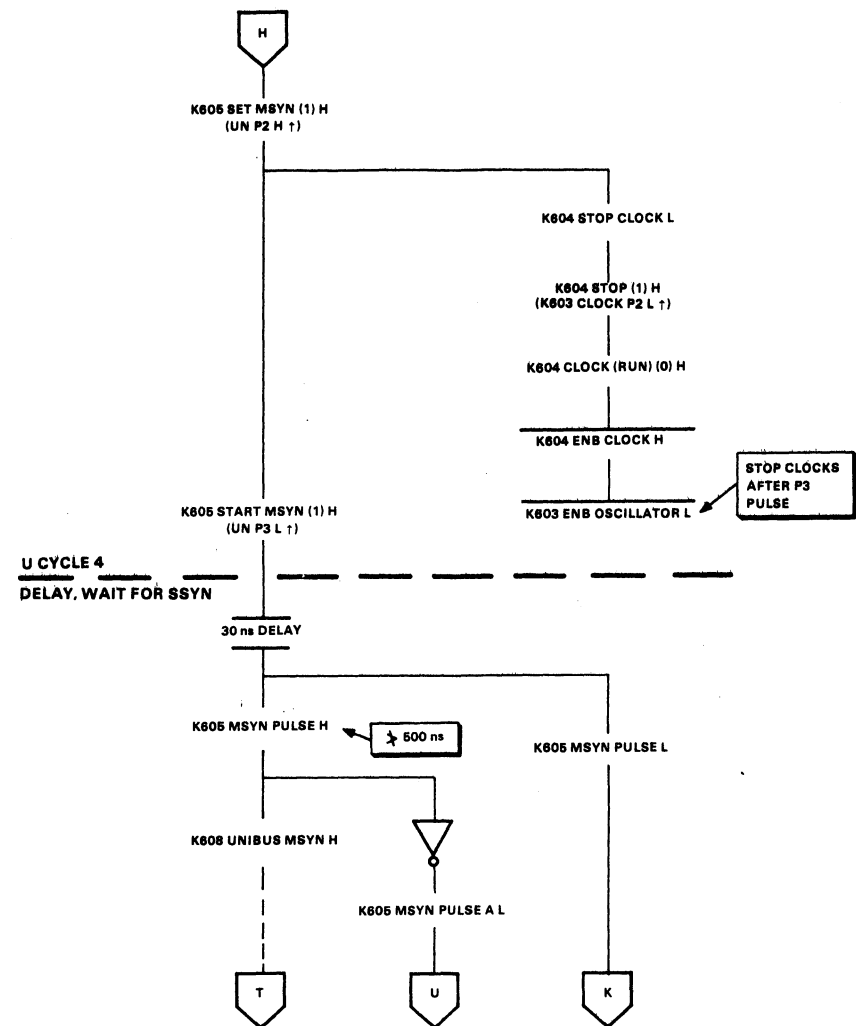
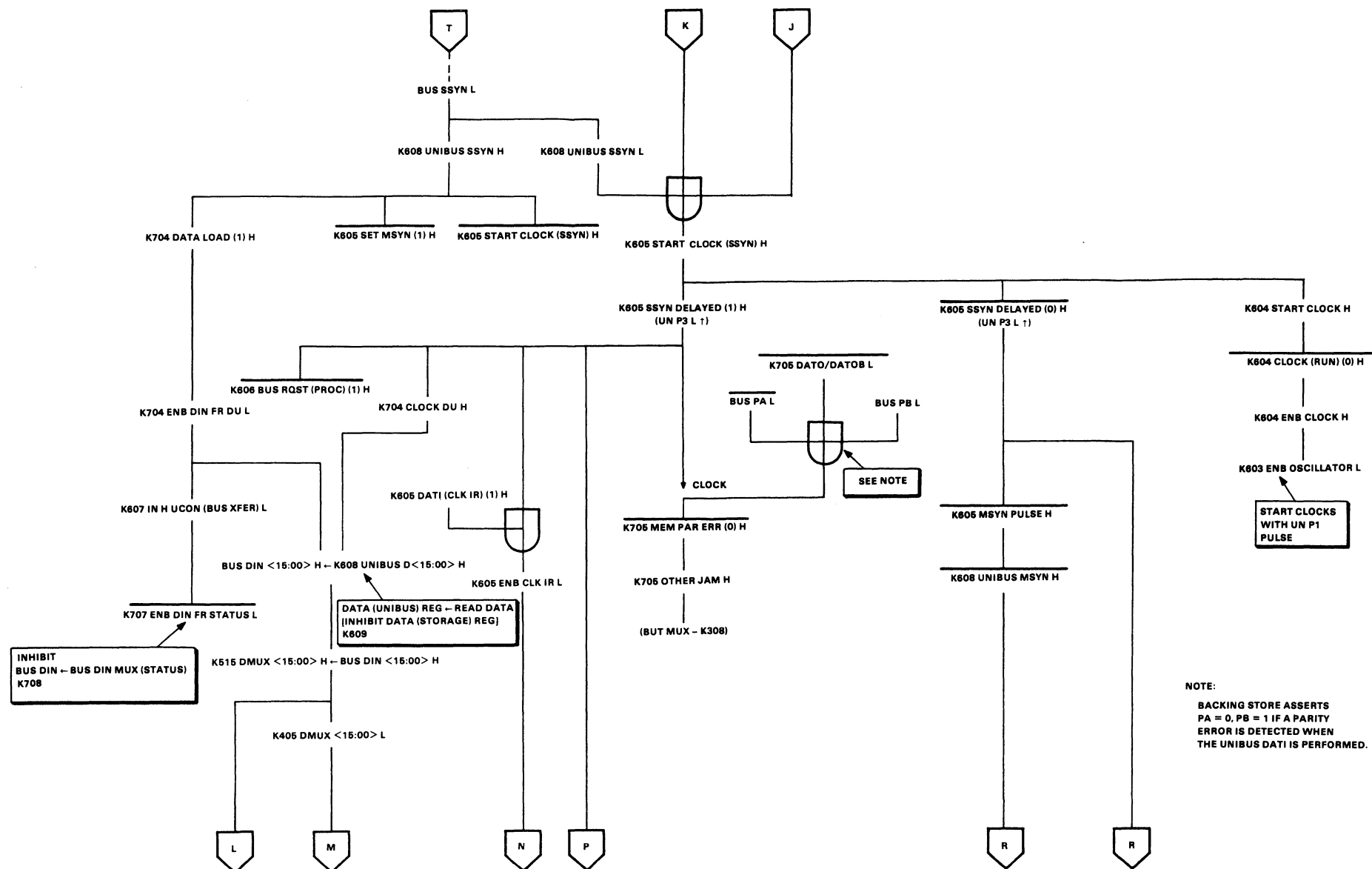


Figure 1-10 DATI Flow (U CYCLE 1)
(Sheet 1 of 6)



MA-0412

Figure 1-10 DATI Flow (U CYCLE 4 and
WAIT FOR SSYN) (Sheet 3 of 6)



MA-0411

Figure 1-10 DATI Flow (U CYCLE 5) (Sheet 4 of 6)

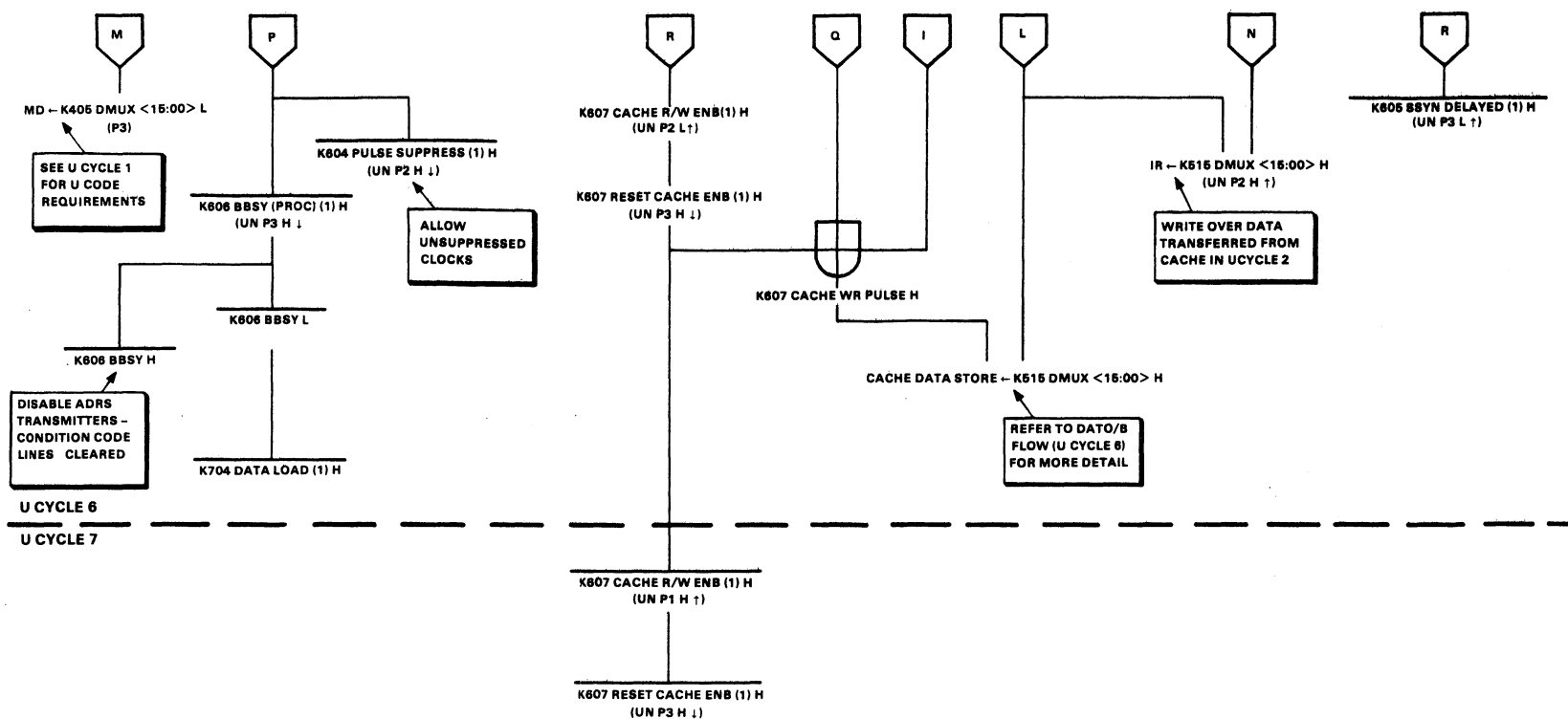
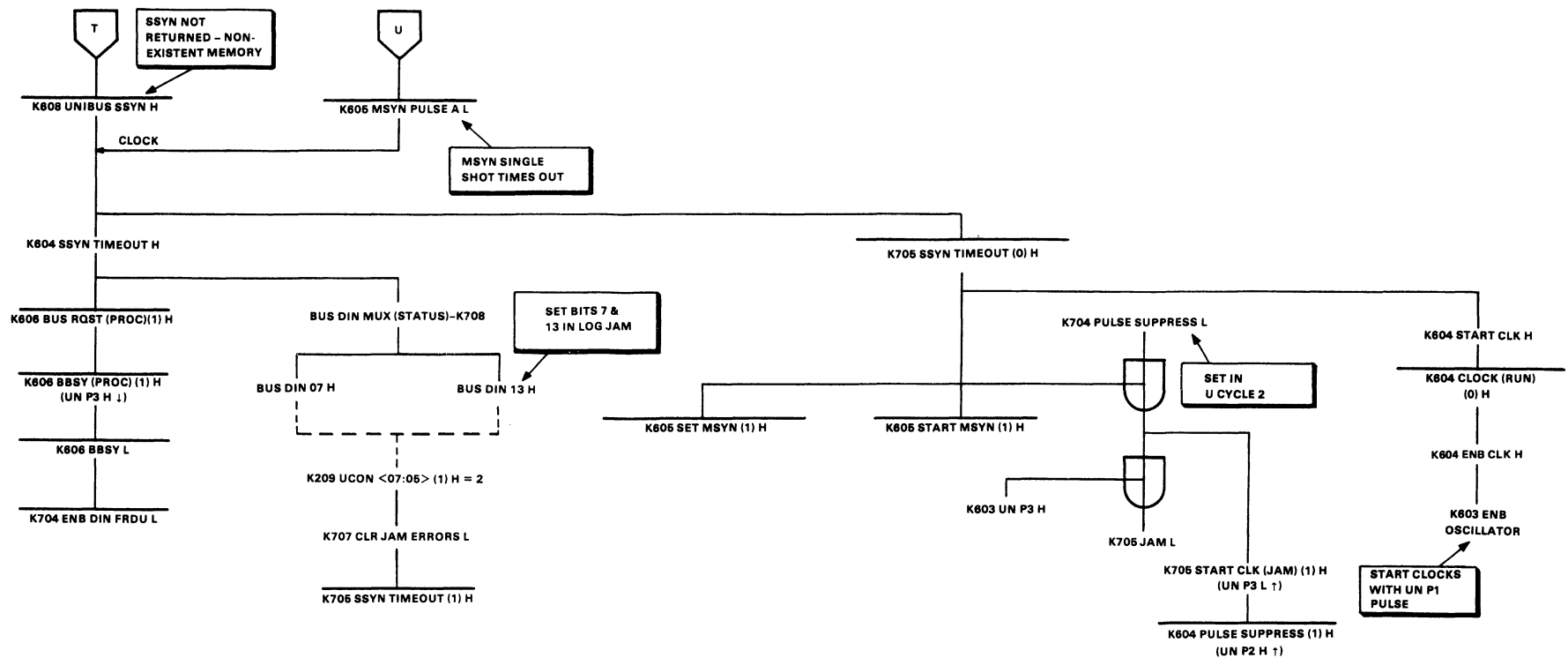


Figure 1-10 DATI Flow (U CYCLES 6 and 7) (Sheet 5 of 6)



MA-0410

Figure 1-10 DATI Flow (SSYN TIMEOUT) (Sheet 6 of 6)

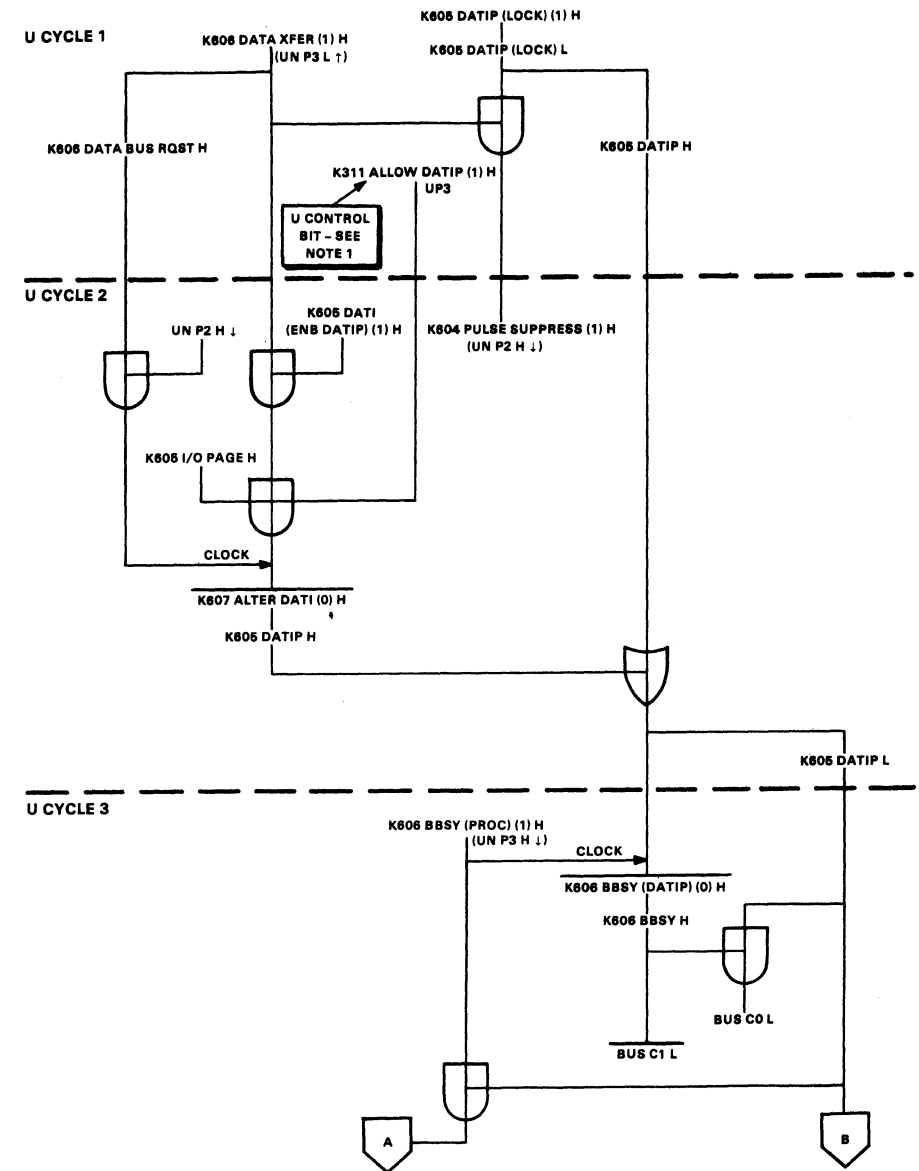
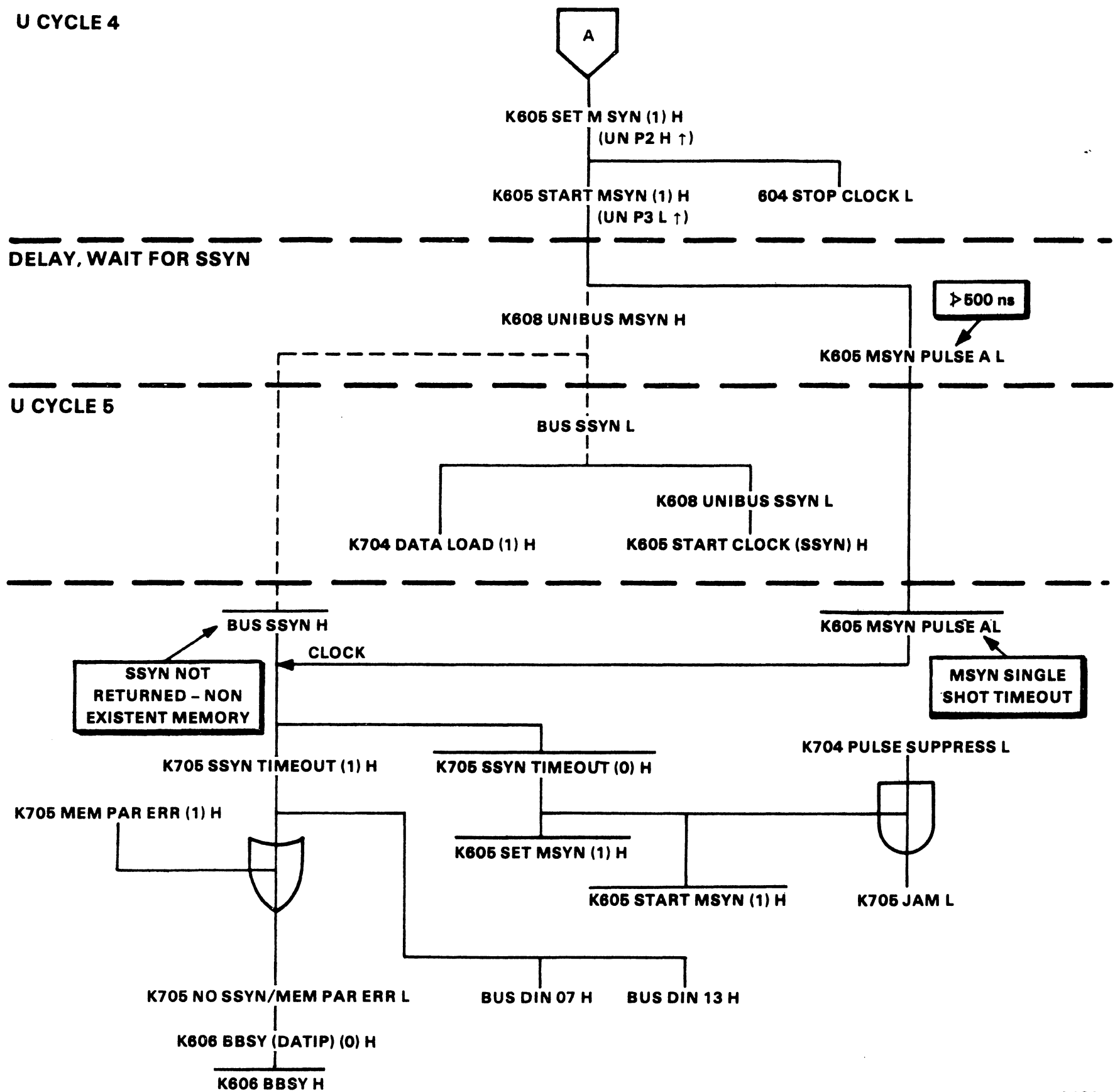


Figure 1-11 DATIP Flow (U CYCLES 1, 2, and 3) (Sheet 1 of 3)

U CYCLE 4



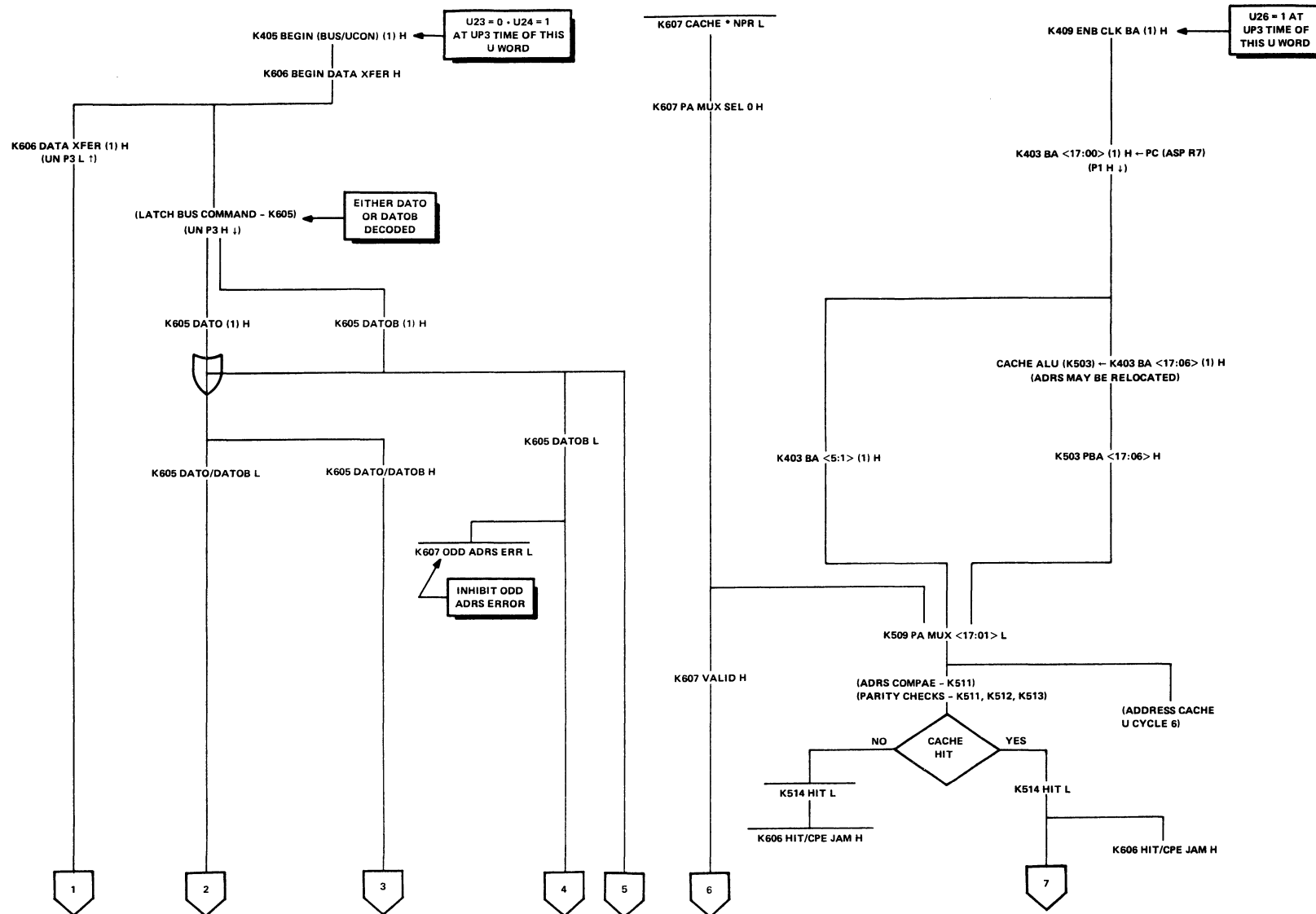
MA-0408

Figure 1-11 DATIP Flow (U CYCLES 4 and 5) (Sheet 2 of 3)

24

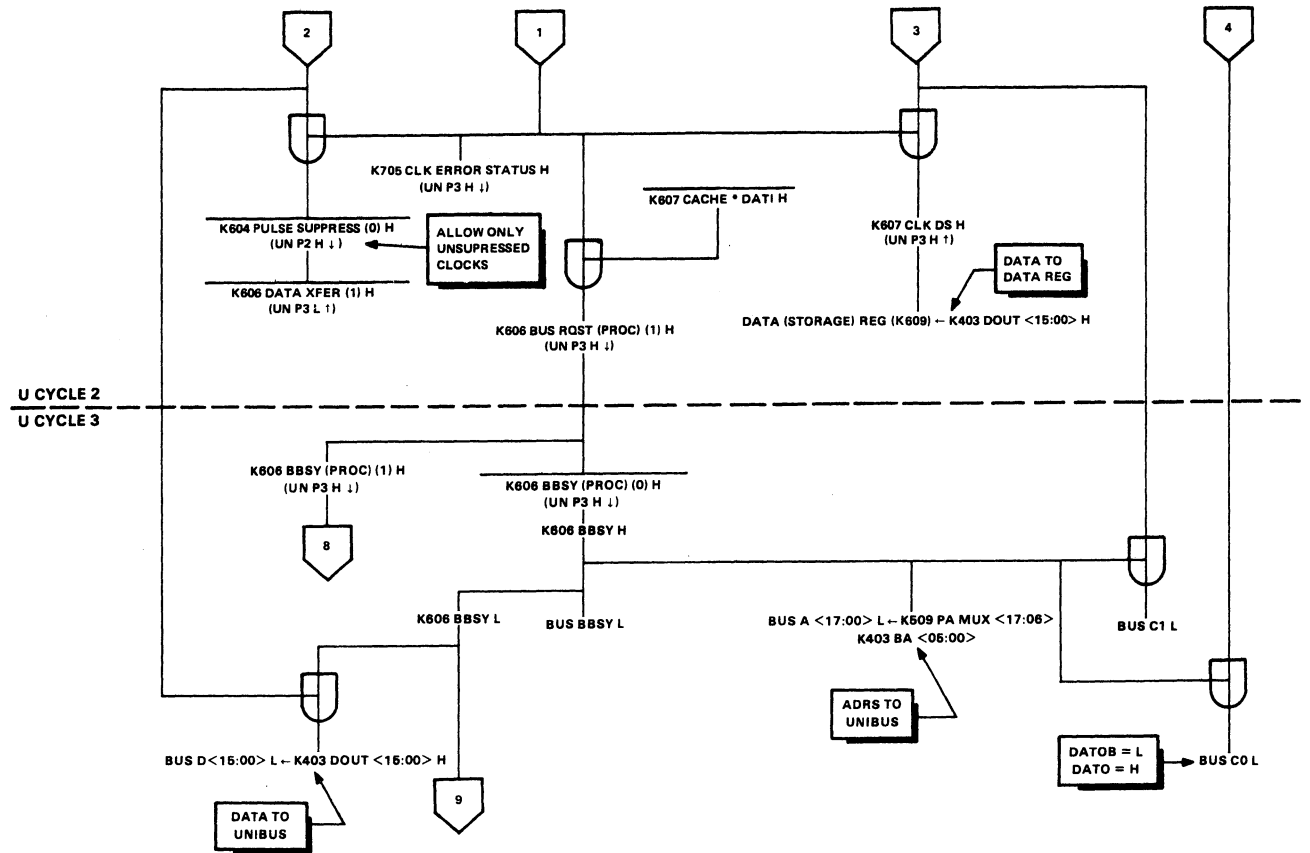


•



MA-0480

Figure 1-12 DATO/DATOB Flow
(U CYCLE 1) (Sheet 1 of 5)

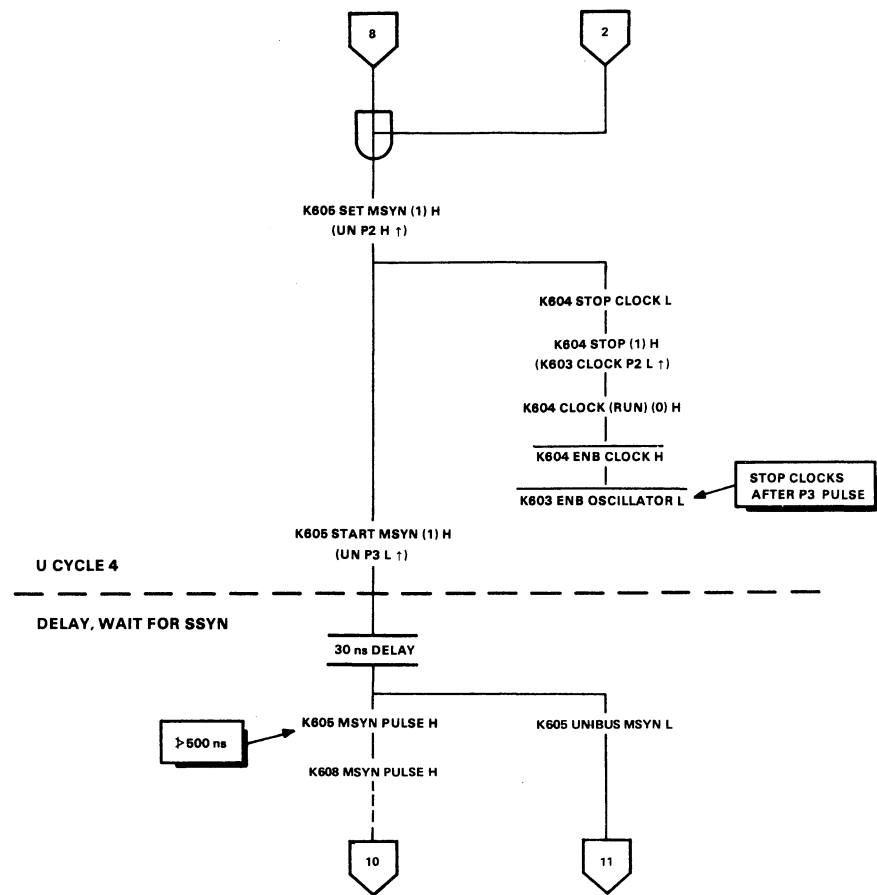


248

MA-0481

Figure 1-12 DATO/DATOB Flow
(U CYCLES 2 and 3) (Sheet 2 of 5)

24C



MA-0482

Figure 1-12 DATO/DATOB Flow (U CYCLE 4 and WAIT FOR SSYN) (Sheet 3 of 5)

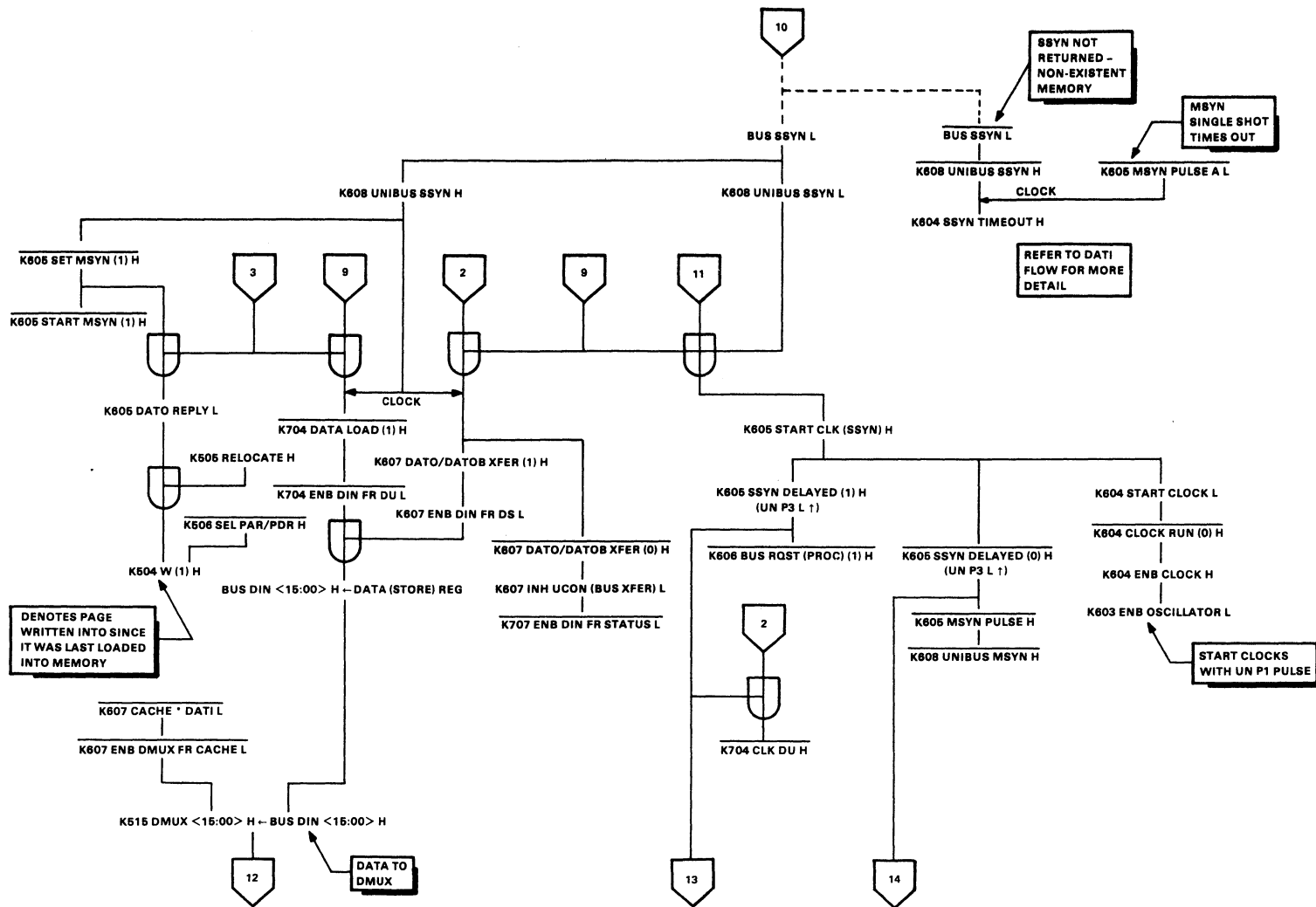
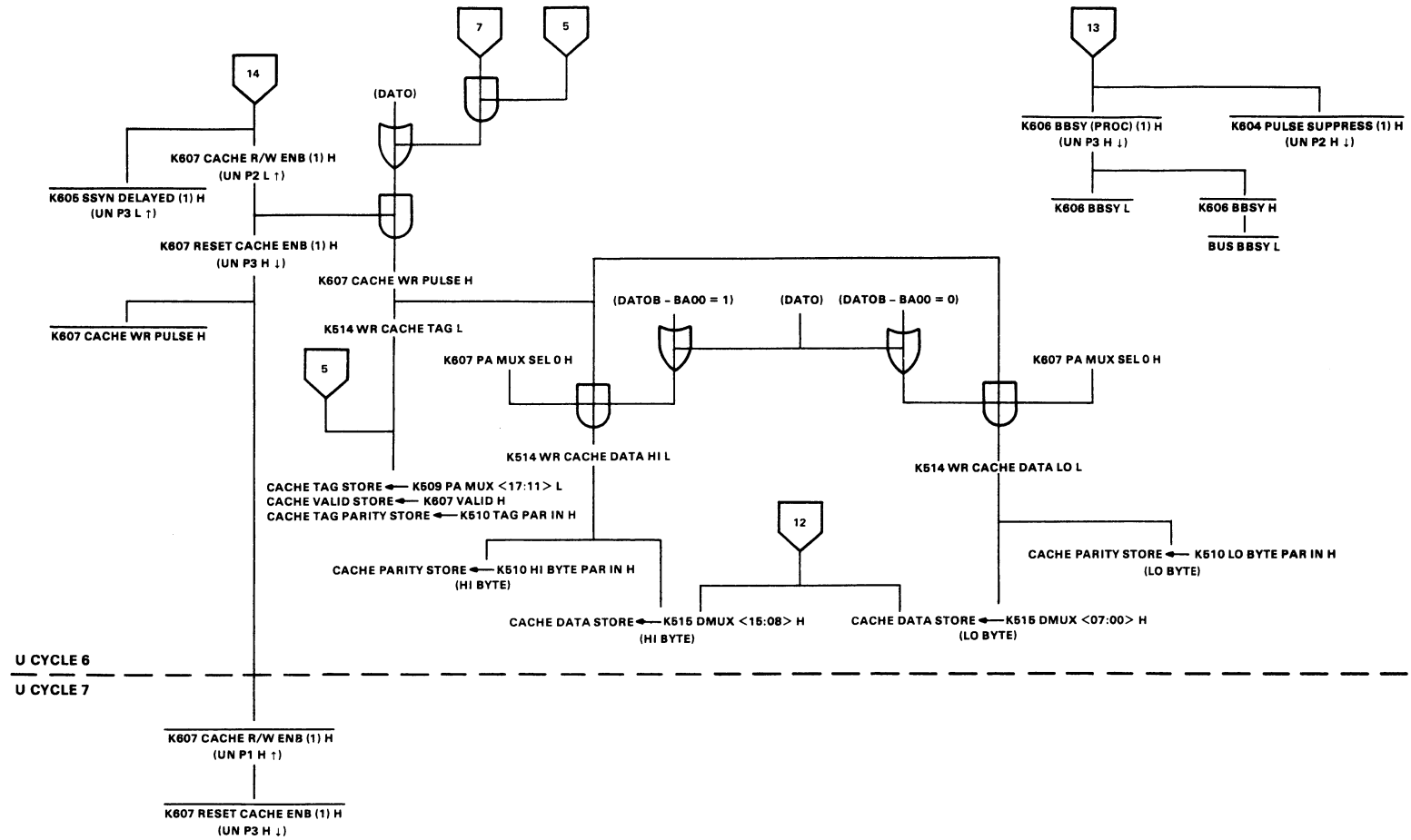


Figure 1-12 DATO/DATOB Flow
(U CYCLE 5) (Sheet 4 of 5)

24E



MA-0484

Figure 1-12 DATO/DATOB FLOW
(U CYCLES 6 and 7) (Sheet 5 of 5)

Several flip-flops are cleared in U CYCLE 7 to conclude the DATI operation.

SSYN is not returned by the memory device if the address placed on the Unibus in U CYCLE 3 is nonexistent. The single shot that produces K605 MSYN PULSE H times out and one of the associated signals, K605 MSYN PULSE A L (going high), clocks the SSYN Timeout flip-flop. This causes a series of events the most notable of which is a jam. In addition, the BUS RQST (PROC) flip-flop is reset (and consequently BUS BUSY is negated) and K605 SSYN TIMEOUT (1) H sets bits 7 and 13 in the C scratchpad jam register via the BUS DIN status multiplexer.

The jam routine (JAM 01), entered when the jam occurs, reads the jam register and branches on bit 7 to the SSYN timeout routine (SST 01) where a trap to location 4 occurs. (Jam register bit 13 is transferred to the CPU error register, bit 4.) UCON bits (7:5), set to a configuration of 2 by the microcode, assert K707 CLR JAM ERRORS L, which clears the SSYN Timeout flip-flop.

1.6.4 DATIP Flow

The DATIP sequence is quite similar to the DATI procedure (Paragraph 1.6.3). The primary difference occurs after the data has been read; BUS BUSY remains asserted at this time until a DATO/B has been completed to the same address. The bus cannot be used by the processor or other devices during this period.

The DATIP flow (Figure 1-11) shows the events in the order in which they occur. With the exception of several signals included for clarity, the signals shown are unique to DATIP.

A DATIP can be invoked in two ways: when DATIP (LOCK) is decoded in the bus command ROM (Table 1-6), or when DATI (ENB DATIP) is decoded. In the latter case, the address must be in the I/O page and K311 ALLOW DATIP (1) H must be asserted. This signal is derived from a microcontrol bit, K310 U (ALLOW DATIP) H.

In U CYCLE 1, K605 DATIP (LOCK) (1) H is latched at P3 time (Figure 1-11), this is one way to assert K605 DATIP H. The other way, when the Alter DATI flip-flop is set, is used if an I/O page address is specified. K606 DATA XFER (1) H, derived from K405 BEGIN (BUS/UCON) (1) H, is asserted in U CYCLE 1, the same as it is in the DATI sequence. This signal is ANDed with several others: I/O PAGE H, K605 DATI (ENB DATIP) (1) H (a Bus Command ROM output), and K311 ALLOW DATIP (1) H (which must be specified for the second microword, i.e., prior to P2 of U CYCLE 2). The output from the AND function constitutes the D input to the Alter DATI flip-flop, which is clocked at P2 time of U CYCLE 2.

The BBSY (PROC) flip-flop is set at P3 time in U CYCLE 3; K606 BBSY (PROC) (1) H, going high, clocks the BBSY (DATIP) flip-flop. BUS BUSY is now asserted and the control lines have the value for DATIP (C1=0, C0=1). In U CYCLE 4, the Set MSYN and Start MSYN flip-flops are set. MSYN is asserted on the Unibus and the processor waits for SSYN. The assertion of SSYN indicates that the requested data is on the Unibus D lines. If SSYN is not returned by the time the MSYN single shot times out, the SSYN Timeout flip-flop is set and K705 NO SSYN/NEM PAR ERR L negates the BUS BUSY signals. SSYN Timeout also causes a jam.

The cache write pulse is produced in U CYCLE 6 of DATI operations. However, K605 DATIP L inhibits the write pulse during a DATIP; the cache location will be written to during the subsequent DATO/B when the bus control logic transmits the new data to this address.

BUS BUSY is not negated until the completion of the DATO/B. During U CYCLE 3 of the DATO/B flow, the BBSY (DATIP) flip-flop is reset when K606 BBSY (PROC) (1) H goes high (K605 DATIP H is not high during the DATO/B). K606 BBSY H is then negated in U CYCLE 6 of the DATO/B when the BBSY (PROC) flip-flop is reset.

1.6.5 DATO/DATOB Flow

DATO and DATOB operations involve writing new data in backing store locations (Table 1-5). The sequence of events for the two commands (Figure 1-12) are quite similar: the address is placed on the Unibus, a bus cycle is initiated, the data is written into the backing store location, and, finally, the data is written into the appropriate cache location during a cache cycle.

However, there are several differences between a DATO and a DATOB: an odd backing store location can be accessed only by a DATOB; the C1, C0 control bit configuration is 10 for a DATO and 11 for a DATOB; and a cache write is inhibited for a DATOB to an odd address if there is a cache miss. (Since a cache location is used to retain a copy of the data held at any one of 128 backing store addresses, the latter precludes writing a byte over a word, and thus generating incorrect word data, at any of the 127 "miss" addresses.)

The first microword (U CYCLE 1) of a DATO/B operation defines the command and causes the assertion of two signals that start the sequence, K409 ENB CLK BA (1) H and K405 BEGIN (BUS/UCON) H. At P1 time, the address is clocked into the BA register where it is transferred to cache for address comparison and parity checks. K606 BEGIN DATA XFER H latches either DATO or DATOB at P3 time; several signals are developed from these commands.

In U CYCLE 2 the clocks are suppressed, a bus request is generated, and data is clocked into the data storage register at P3 time. This is the data destined for cache via BUS DIN and the data multiplexer. (However, it is not placed on the DIN lines until after SSYN signifies that the data has been written into the backing store. Figure 2-10 shows the path followed by write data to both storage devices.

25 The processor asserts BUS BUSY in U CYCLE 3. Then, with BUS BBSY L asserted, the data, address, and control bits are placed on their respective Unibus lines. With K606 BBSY (PROC) (1) H high, the Set MSYN flip-flop is set in U CYCLE 4; the clocks are stopped (they will remain stopped until SSYN is returned), and the Start MSYN flip-flop is set. U CYCLE 4, and a 30 ns delay after Start MSYN is set, provides the time necessary to deskew the address, data, and control bits after they are placed on the Unibus in U CYCLE 3.

After the 30 ns delay, the processor asserts K608 UNIBUS MSYN H to request the backing store to perform the DATO/B; the backing store asserts BUS SSYN L to indicate completion of the operation.

The same data sent to the backing store will now be written into the appropriate cache location. U CYCLE 5 begins when K608 UNIBUS SSYN L asserts K605 START CLK (SSYN) H; clock pulse generation resumes with K603 UN P1 H. (The Pulse Suppress flip-flop remains set from U CYCLE 2.) K605 START CLK (SSYN) H also sets the SSYN Delayed flip-flop, which clears the processor bus request and MSYN in this microcycle and asserts K607 CACHE R/W ENB (1) H in U CYCLE 6.

K608 UNIBUS SSYN H, which is asserted simultaneously with K608 UNIBUS SSYN L, sets the DATO/DATOB XFER flip-flop and clears the Data Load flip-flop (if not already cleared). The consequence of these two actions is the assertion of K607 ENB DIN FR DS L and the negation of K704 ENB DIN FR DU L which, together, cause the transfer of the data in the data storage register to BUS DIN. K607 ENB DMUX FR CACHE L, high (negated) at this time because a cache DATI is not being performed, selects the data on BUS DIN as the input to the data multiplexer on the cache module.

In U CYCLE 6, K607 CACHE R/W ENB (1) H, asserted at P2 time, initiates the cache write pulse. Either one of two conditions must also exist: a DATO bus command, or a DATOB and a hit (K514 HIT L). The write pulse, K607 CACHE WR PULSE H, results in the assertion of K514 WR CACHE TAG L, K514 WR CACHE DATA HI L, and K514 WR CACHE DATA LO L that function as write enable signals to the type 93415A cache store chips. The assertion of the latter two signals is determined by the command type (DATO or DATOB) and the state of the address LSB, K403 BA00 (1) H.

Refer to Figure 1-12, U CYCLE 6, which shows how these write pulse derivatives control the storage of the different parts of the tag and data word. The cache write pulse, with a duration of approximately 55 ns, is negated at P3 time.

Several other signals are negated in U CYCLE 6: K604 PULSE SUPPRESS (1) H, which allows production of suppressed clock pulses; K606 BBSY (PROC) (1) H, which releases the bus; and SSYN delayed. In U CYCLE 7, two flip-flops are cleared to conclude the DATO/B flow: Cache R/W ENB and Reset Cache ENB.

If SSYN is not returned (the result of an unsuccessful attempt to access a nonexistent memory location) the MSYN pulse, going low, clocks the SSYN Timeout flip-flop to initiate a jam. This is shown in detail in the DATI flow, Figure 1-10.

1.7 SERVICE

During the execution of the microprogram there are several devices and conditions that can cause an interruption of the program so that the device or condition can be attended to or serviced. For example, if a console key is pressed, the action is recognized after the execution of the current macroinstruction. A service routine in the microprogram is then entered during which the key is identified and the particular key function is serviced.

The logic that pertains to service operations is shown in Figure 1-13. The six devices and conditions (events) that can invoke a service request are listed, along with their priority, in this figure. The service routine identifies the highest priority event that caused the request for service and then passes control to a subroutine that handles that event. Each event that initiates a service request sets a flag that is sent to the BUS DIN MUX (STATUS), shown on drawing K708. From there it is later read into the log service register in the C scratchpad (CSP) to identify which event requested that service. If any of the flags are set, one or more of the BUTs which test for service will be true.

A BUT on service, BUT (20), is done at the end of every macroinstruction and at the end of every shared trap flow. At this time any one of the six events will cause entry into the service flow. Then, early in the service flow, BUT (26) tests the state of K210 INTR HIGH H. If none of the top five priority events are pending, the service flip-flop remains reset and K210 INTR HIGH H is high. From this it can be assumed that an interrupt is the highest pending service condition and that part of the flow designed to deal with the five top priority events is bypassed. (Note that even though an interrupt is the lowest priority service condition, it is the first event tested for. This is done to decrease the latency of interrupts. Note also that the service routine does not test if an interrupt is pending but that the interrupt is the highest service condition pending.)

If an interrupt is the highest pending service request, a bus grant is allowed and the interrupt vector is read from the Unibus. (Refer to Paragraph 1.6.1 for a detailed description of the interrupt sequence.) Then, BUT (21) tests if the interrupt actually occurred or if there was a passive release of the Unibus.

The low five-order bits of the log service register are now read in preparation for branching of the top five priority service events to the appropriate service subroutine. Next, the T-bit is tested with BUT (14). If it is true, a T-bit trap is performed by passing a constant through the MD to the shared trap flow. If it is not true, the flow branches on the low five-order bits of the BUS DIN MUX (STATUS) – from low order (highest priority event) to higher order – and the pending request is serviced.

BUT (07) is used to test for an interrupt during a floating-point operation.

1.7.1 Unibus Interrupt Flow

Peripheral services can interrupt processor operation in order to service an immediate condition. For example, a tape reader interrupts the processor when information is ready to be transmitted over the data lines. A device can interrupt the processor only after it has gained control of the bus as the result of a bus request (BR); BRs are honored on a priority basis.

Along with the interrupt signals, the device sends an interrupt vector address. This data at the address is the starting address of the interrupt service routine. Loaded into the program counter (PC), the address directs the processor to the beginning of the macroroutine that is used to service the interrupting peripheral. The next word (at vector address +2) is a new processor status word (PSW), which establishes the initial conditions required for the service routine. The vector address and the location of the new PSW use memory locations 040 through 776.

Table 1-7 shows the handshaking that takes place between the processor bus control logic and the peripheral device that sends the interrupt.

Figure 1-14 expands on this to show the activity, internal to the processor, that occurs as the result of an interrupt. Only those microcycles, e.g., U CYCLE A, U CYCLE B, etc., where a clocked event in the processor logic is directly involved with the interrupt sequence are shown. For example, between U CYCLES A and B there is microcode that executes a BUT that results when BG SERVICE is set in U CYCLE A. This microcode is not detailed. However, as a result K209 UCON (7:5) (1) H are set to a configuration of 6 and the sequence continues in the processor. A dashed line is used to connect events that occur in sequential order and not as a direct result of signal assertions/negations internal to the bus control logic.

26A
In the first microcycle (U CYCLE A) a BR is received from a device that wants to interrupt the processor. The BR is arbitrated (its priority compared with the processor priority) and, if accepted, the BG SERVICE flip-flop is set. The branch microcode (BUT 26) is executed, K209 UCON (7:5) (1) H is set to 6 and, in U CYCLE B, SERVICE BG is set.

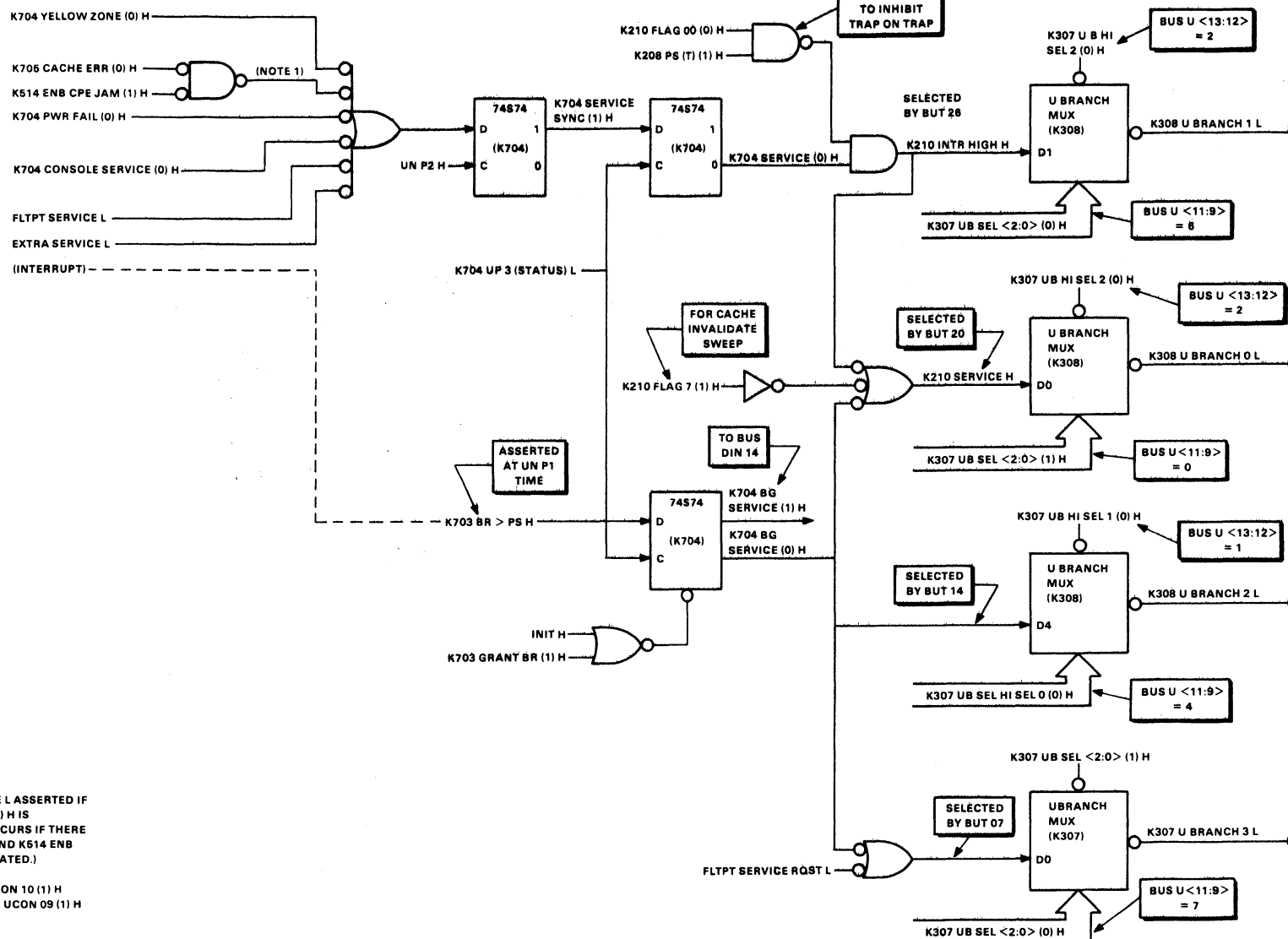
In the next microcycle (U CYCLE C), GRANT BR is set and a BUS GRANT (BG) is sent to the requesting device. Also, at this time, a 10 μ s single shot is started. (The output is used if SACK is not received before the single shot times out. This is described later.) Pulses are suppressed (only unsuppressed pulses are allowed) and ALLOW BG is cleared in this microcycle (unless there is some NPR traffic).

The processor now waits for SACK and, when this is received, GRANT BR is cleared. The BG is negated, the BR register is cleared, and BR arbitration is again allowed. A flip-flop is set (K704) that is used if there is a passive release of the Unibus by the requesting device (off-page connector G). Bus busy (BUS BBSY L), after being negated by the previous bus master, is asserted by the requesting device that now becomes the new bus master.

The interrupt vector address is now placed on the Unibus data lines and BUS INTR L is asserted. (Note that Unibus and processor activity from the time BG was asserted until after BUS INTR L was received is asynchronous; events are not dependent on internal clock pulses.)

In U CYCLE D, a flip-flop (K704) is set. This flip-flop is situated between the BUS INTR L input and the Vector Load flip-flop. It provides vector deskew time, i.e., compensates for the skew between the assertion of INTR and the data (vector) on the D lines.

PRIORITY	BUS DIN (NOTE 2)
1	00
2	01
3	02
4	03
5	04
6	14



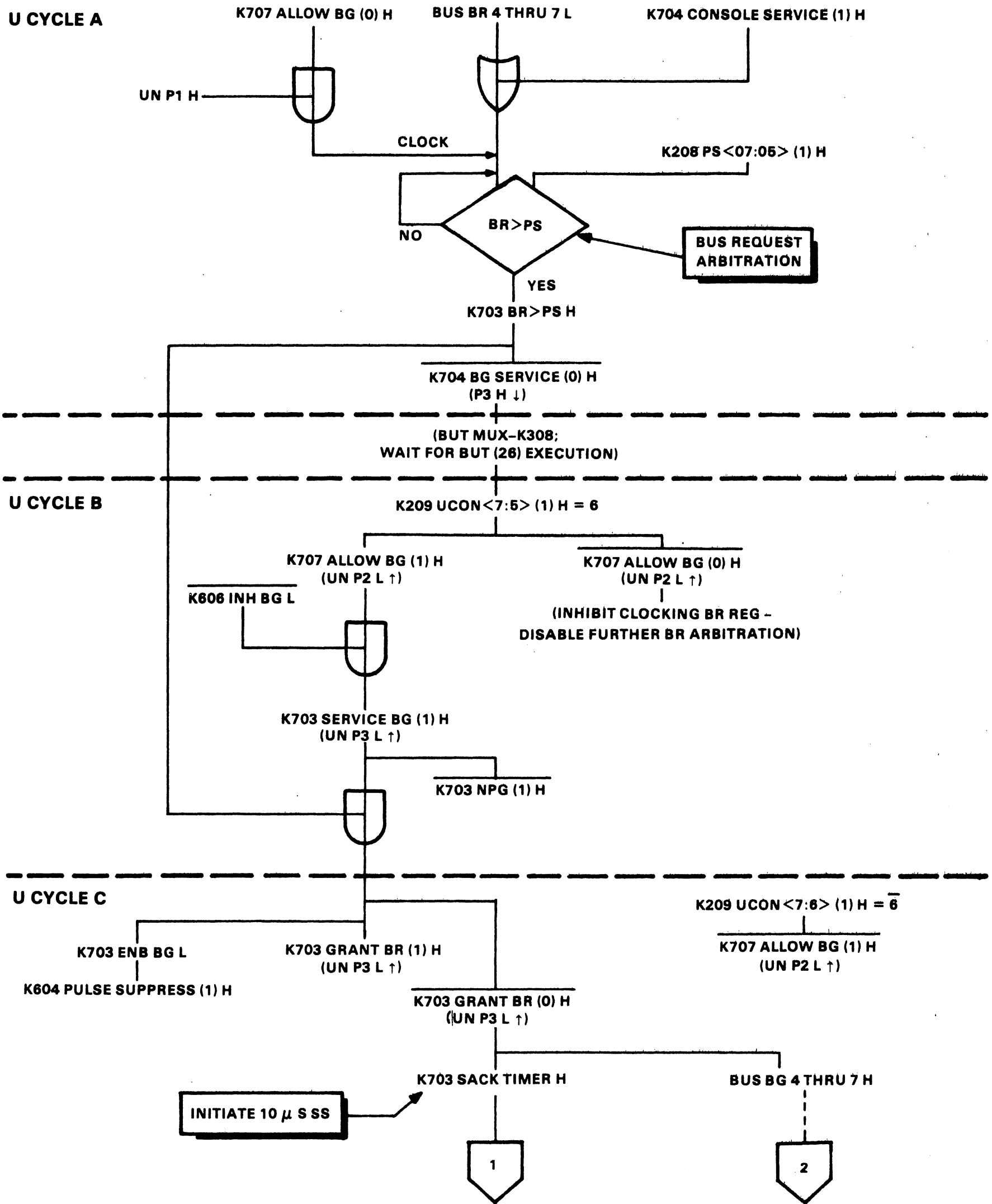
NOTES

1. K705 CACHE SERVICE L ASSERTED IF K514 ENB CPE JAM (1) H IS NEGATED. (A JAM OCCURS IF THERE IS A CACHE ERROR AND K514 ENB CPE JAM (1) H IS NEGATED.)
2. SELECTED IF K209 UCON 10 (1) H (BUS μ 47) = 0 & K209 UCON 09 (1) H (BUS μ 42) = 1

Figure 1-13 Service Requests

Table 1-7 Interrupt Sequence

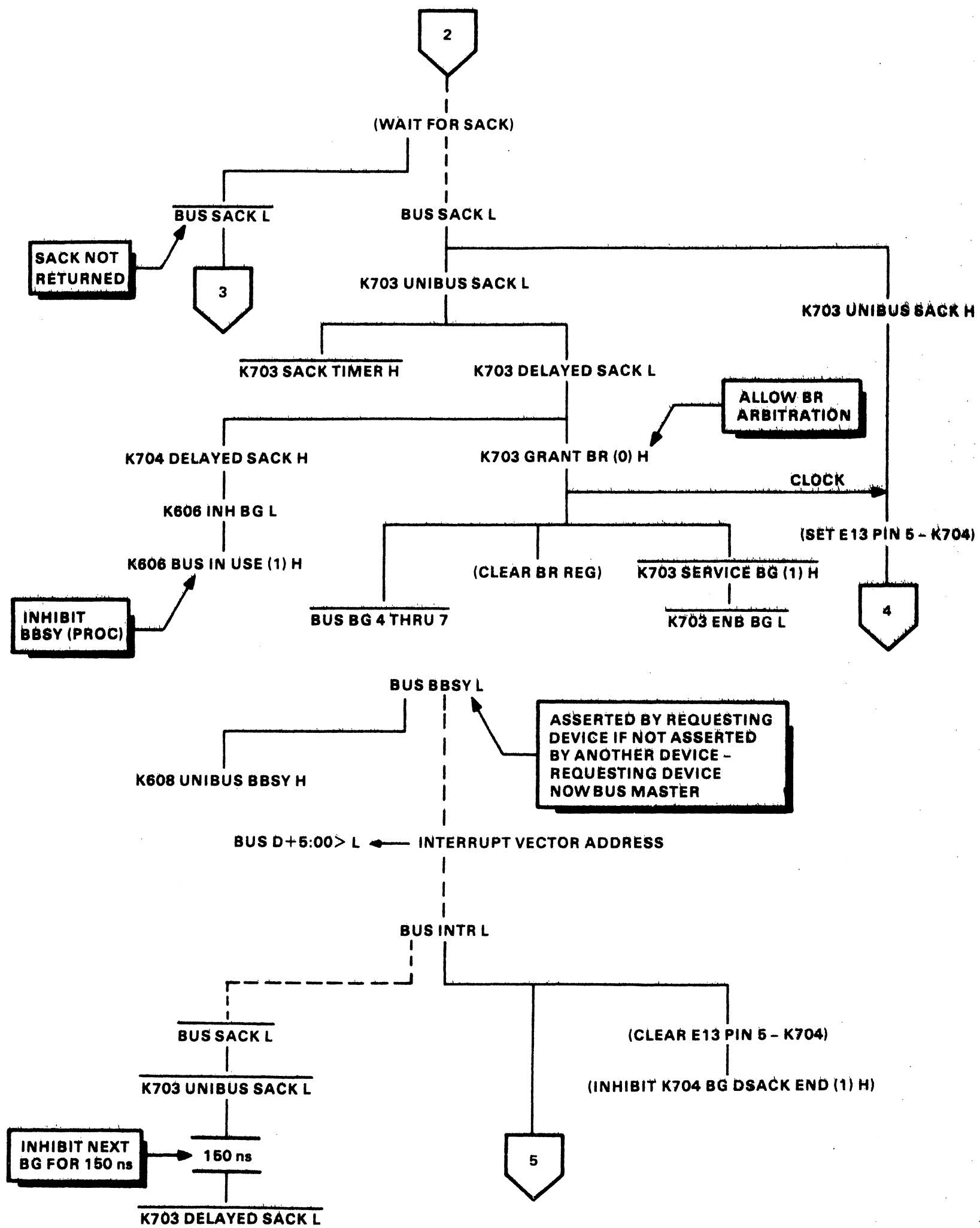
Order	Processor	Device	Meaning
1		BR (asserted)	Requests control of the bus
2	BG (asserted)		Request granted, slave can take over when the Unibus is available (BUS BBSY L)
3		SACK (asserted)	Acknowledges receipt of the bus grant (BG)
4	BG (negated)		Response to SACK by processor
5		BBSY (negated) (from previous bus master)	Address and control lines are clear
6		SSYN (negated) (from previous bus master)	Bus is now free
7		BBSY (asserted)	The requesting device takes over control as the new bus master
8		DATA (vector ADRS asserted) INTR (asserted)	Tells processor to go to interrupt service routine by reading interrupt vector address on the bus data (D) lines
		SACK (negated)	Cleared after INTR, prior to start of last bus cycle device uses
9	SSYN (asserted)		Starting address has been read in by the processor
10		DATA (negated)	Interrupt vector address is cleared
		INTR (negated)	Active release of the bus by the device
	SSYN (negated)		
		INTR (negated)	
		BBSY (negated)	Permits another Unibus device to become bus master
11	SSYN (negated)		Response to INTR clear
12	BBSY (asserted) (if processor wants the bus)		Processor is now the bus master



28

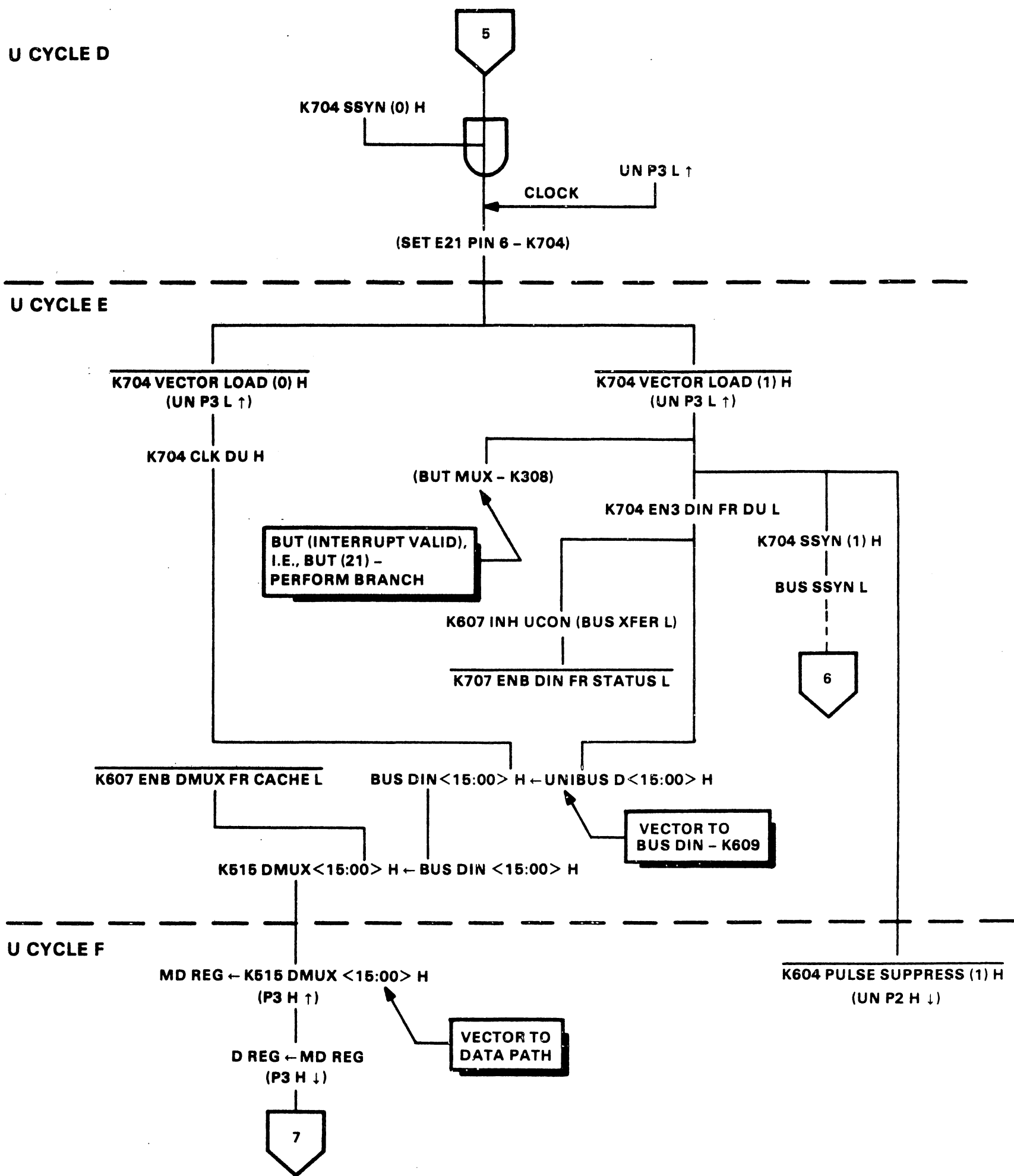
MA-0465

Figure 1-14 Bus Interrupt Flow (Sheet 1 of 6)



MA-0466

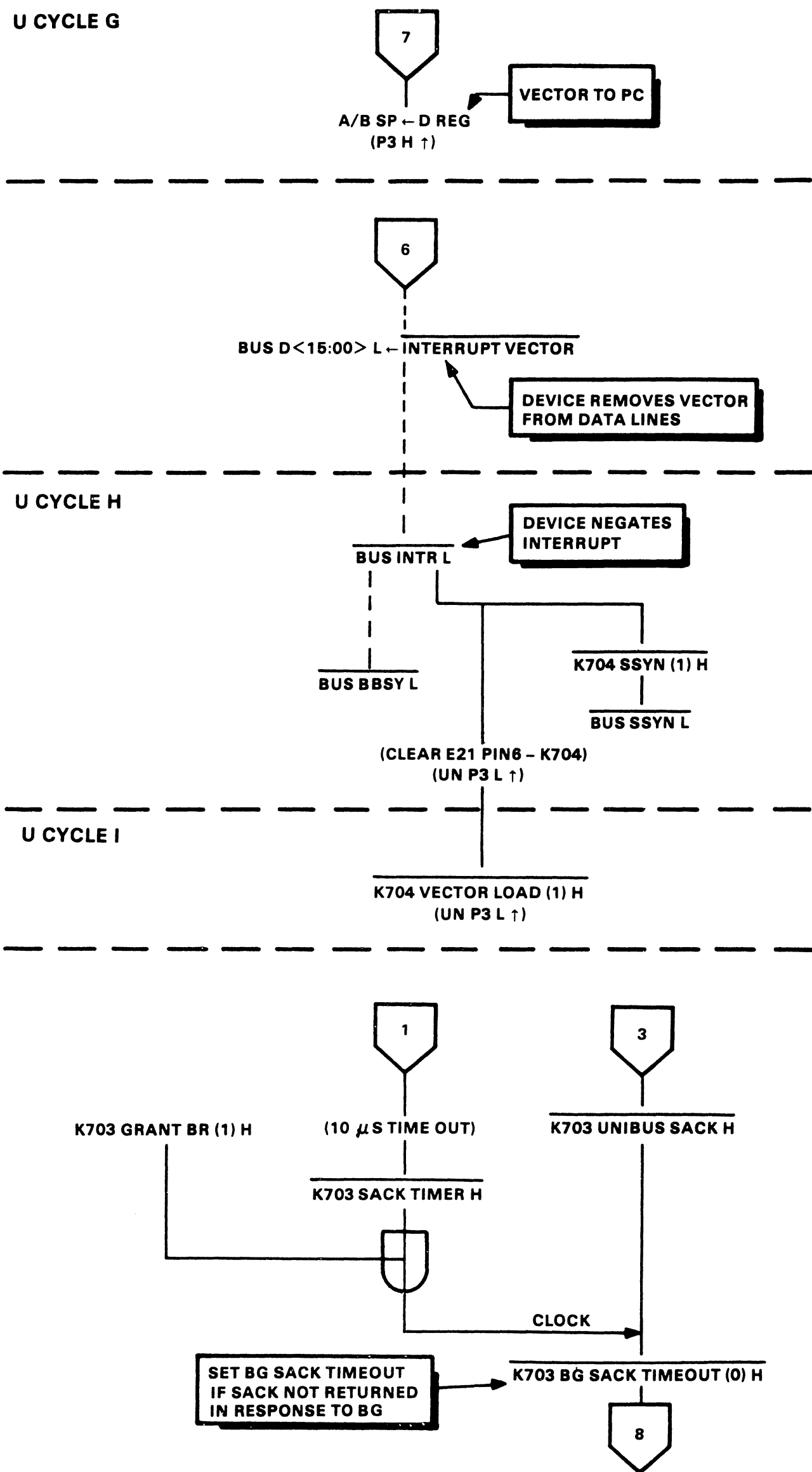
Figure 1-14 Bus Interrupt Flow (Sheet 2 of 6)



MA-0467

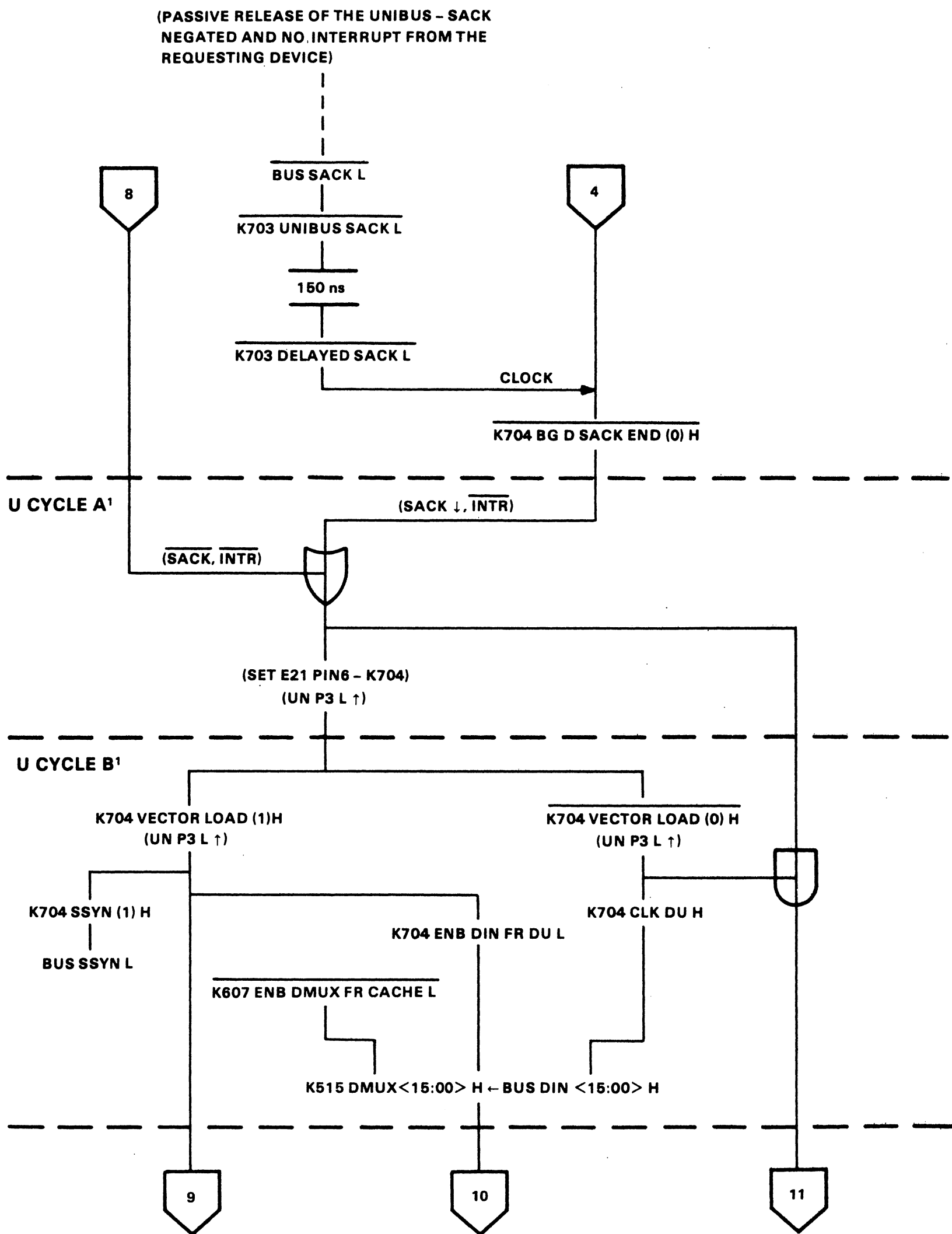
Figure 1-14 Bus Interrupt Flow (Sheet 3 of 6)

31



MA-0468

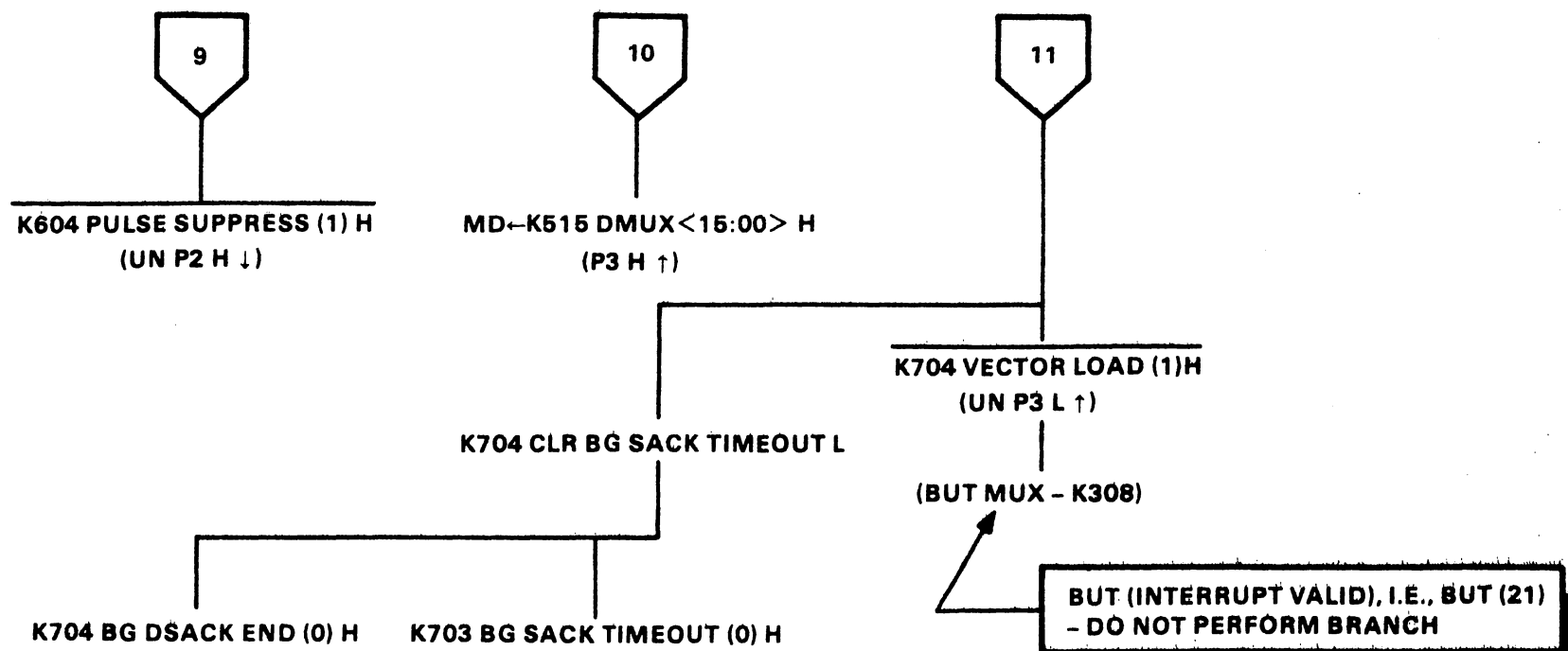
Figure 1-14 Bus Interrupt Flow (Sheet 4 of 6)



MA-0469

Figure 1-14 Bus Interrupt Flow (Sheet 5 of 6)

U CYCLE C'



MA-0470

Figure 1-14 Bus Interrupt Flow (Sheet 6 of 6)

VECTOR LOAD is set in U CYCLE E. This is followed by the execution of the BUT (INTERRUPT VALID) microcode (BUT 21), the transfer of the vector address from the Unibus data lines to the DIN lines, and the assertion of BUS SSYN L. The vector is then routed through the DMUX, and in U CYCLE F, clocked into the MD register in the CSP. From the MD register, the vector is circulated (U CYCLE G) through the data path to the PCs (R7) in the A and B scratchpads. The device drops the vector from the Unibus D lines, and then negates BUS INTR L and BUS BBSY L; BUS SSYN L and VECTOR LOAD are negated by the processor during U CYCLE H and U CYCLE I.

If BUS SACK L is not returned by the requesting device in response to the BG within 10 μ s, the single shot times out. This negates K703 SACK TIMER H and the BG SACK TIMEOUT flip-flop is clocked (set). This configuration (SACK) sets a flip-flop on K704 in U CYCLE A' which sets VECTOR LOAD in U CYCLE B'. VECTOR LOAD is also set if there is a passive release of the Unibus by the requesting device. Although no vector is loaded into the processor (DMUX) under these conditions, VECTOR LOAD is set in order to "unsuppress" the clocks in U CYCLE C'. VECTOR LOAD is cleared during this cycle; the BUT (INTERRUPT VALID) branch is not made.

1.7.2 NPR Flow

Direct memory or device access data transfers can be performed between two Unibus devices without processor supervision. Termed NPR level data transfers, these transfers are made between the memory and a mass storage device (e.g., a disk) or some other peripheral device.

During NPR operations it is not necessary for the processor to be in the information transfer path between the memory (backing store) and a Unibus device. The Unibus structure allows device-to-device transfers; customer designed peripheral controllers are permitted to directly access other devices on the Unibus. This direct access capability, for example, permits operations such as a disk directly refreshing a CRT display.

An NPR device provides extremely fast access to the Unibus and can transfer data at high rates after it gains control. The processor is not affected by this type of transfer. Therefore, the processor can relinquish control of the bus while an instruction is in progress. Normally, the processor releases the bus when it is not using it. However, the bus is never released between the read and write cycles of a DATIP operation. An NPR device in control of the bus transfers 16-bit words, or 8-bit bytes, to memory at the same speed as a processor write (DATO/B) to memory.

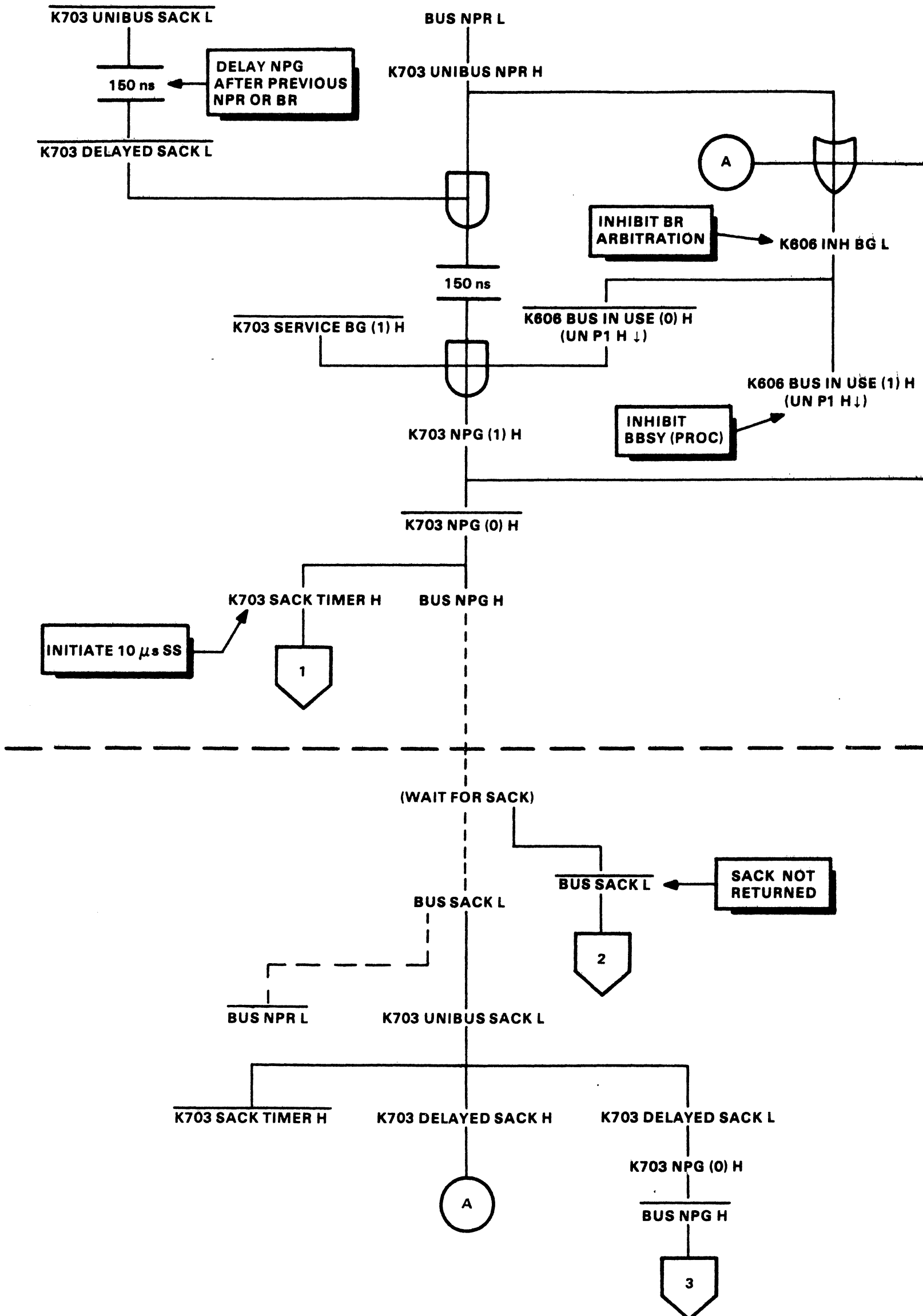
Table 1-8 lists the signals exchanged between the NPR device and the processor when an NPR is issued. Figure 1-15 shows this in more detail. In this figure only those microcycles, e.g., U CYCLE A, U CYCLE B, etc., where one or more clocked events occur in the processor NPR logic are shown. The processor allows NPR operations when not using the bus. At this time the processor is occupied in other activities, e.g., reading cache, executing microcode, etc., which are not shown in Figure 1-15.

NPRs have a higher bus priority than BRs. Consequently, there is no request arbitration when an NPR is received at the processor. However, BRs in progress are completed before an NPR can be honored. On receipt of the NPR there is a 150 ns delay (U CYCLE A) before a non-processor grant (NPG) (BUS NPG H) is sent to the requesting device. If an NPR or BR is in progress at the time the NPR is received, the signal K606 BUS IN USE (0) H will inhibit the NPG until the present NPR or BR is concluded. A 10 μ s single shot signal, K703 SACK TIMER H, is initiated at the time the grant is asserted. This circuit is described below.

Table 1-8 NPR Sequence

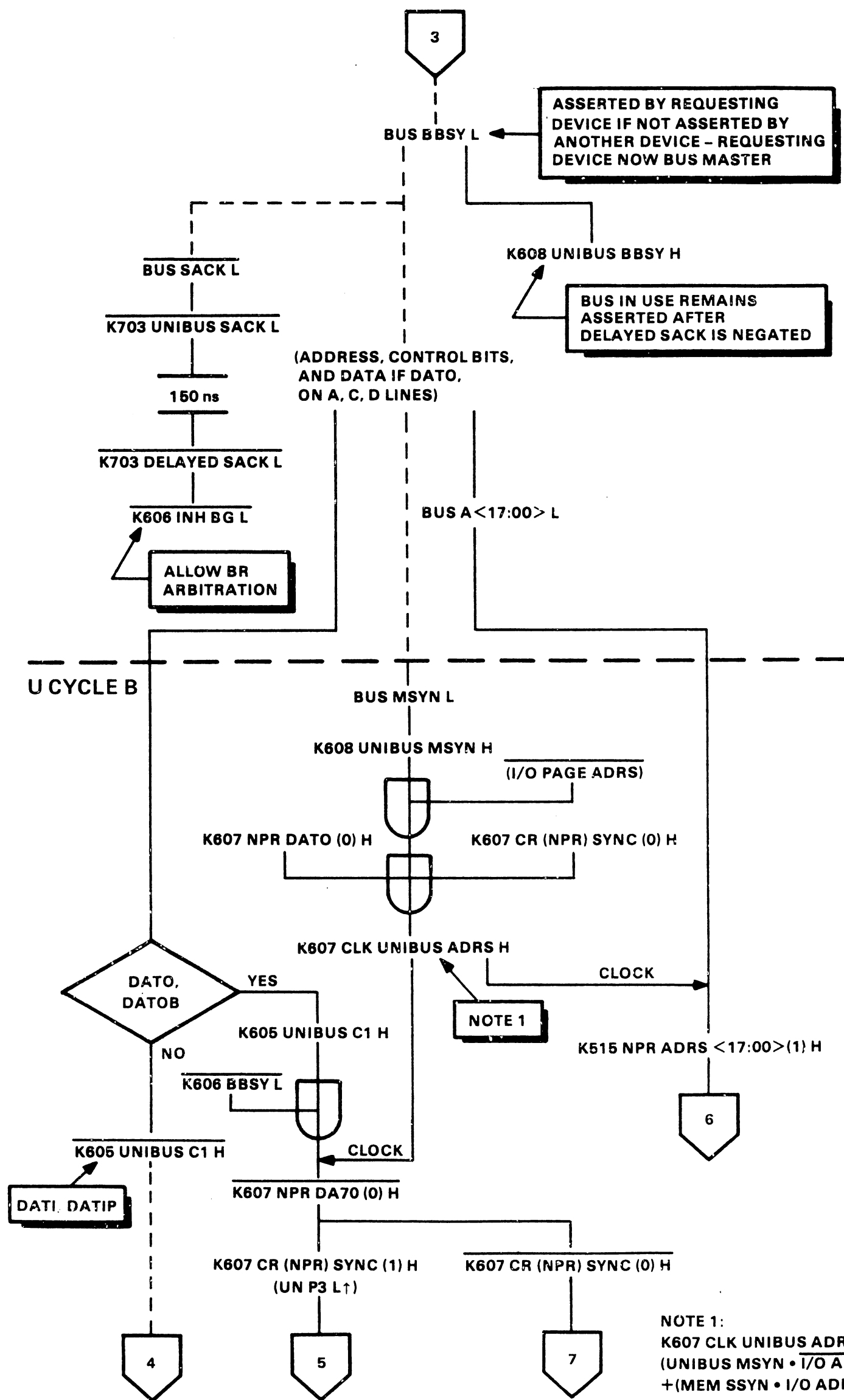
Order	Processor	Device	Meaning
1		NPR (asserted)	Request control of the bus
2	SACK (negated)		From previous priority arbitration
3	NPG (asserted)		Response to NPR
4		SACK (asserted)	Acknowledges receipt of NPG
5	NPG (negated)		Processor's response to SACK
6	BBSY (negated)		From previous bus master
7		BBSY (asserted)	Requesting device now bus master. Data transfer cycle(s) start.
8		MSYN (asserted)	Asserted by bus master after ADRS and control lines asserted
9		SACK (negated)	After BBSY asserted and before last data transfer cycle. (Must occur after NPG negated)
10		MSYN (negated)	Last data transfer completed
11		BBSY (negated)	Master releases bus

U CYCLE A



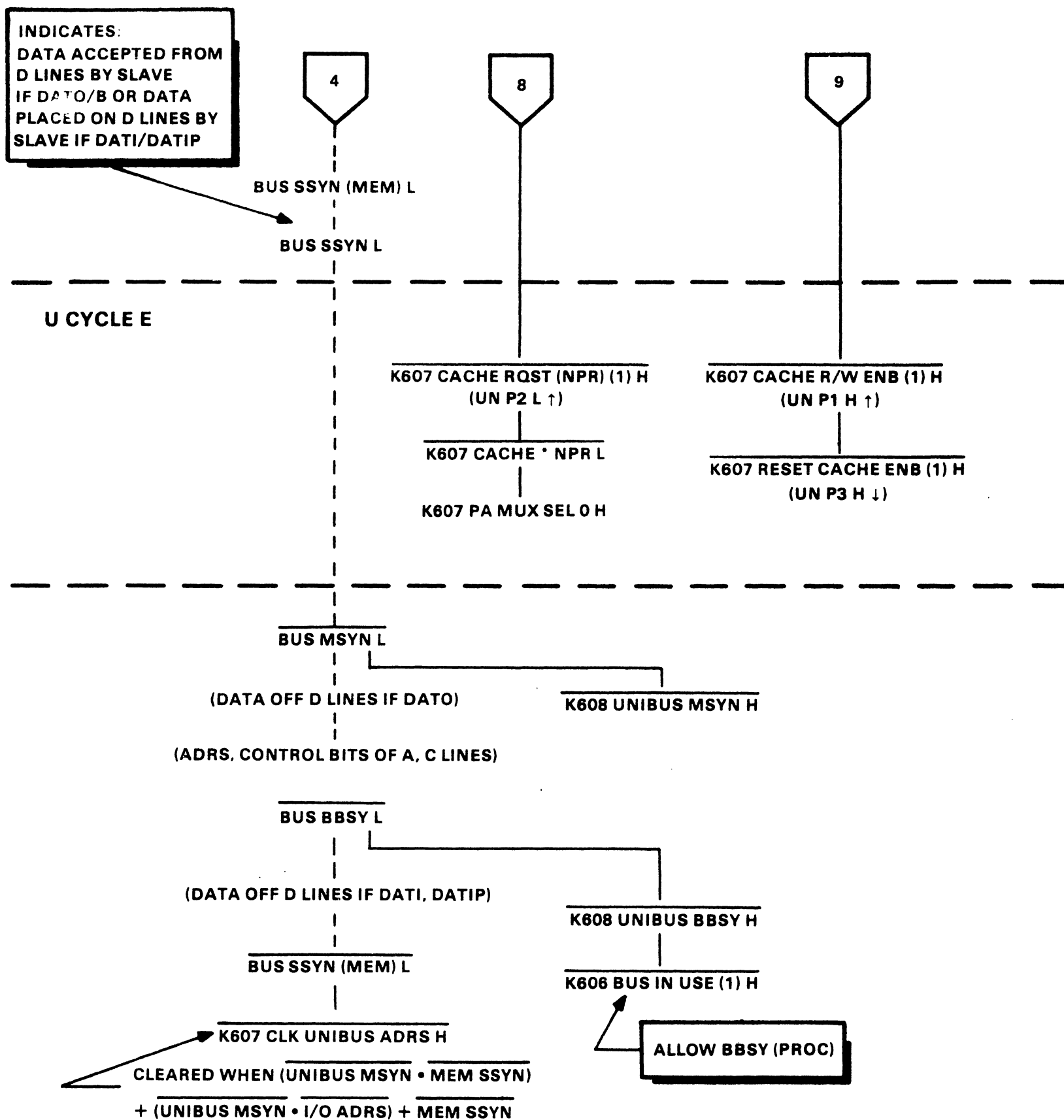
MA-0471

Figure 1-15 NPR Flow (Sheet 1 of 5)



MA-0472

Figure 1-15 NPR Flow (Sheet 2 of 5)



MA-0474

Figure 1-15 NPR Flow (Sheet 4 of 5)

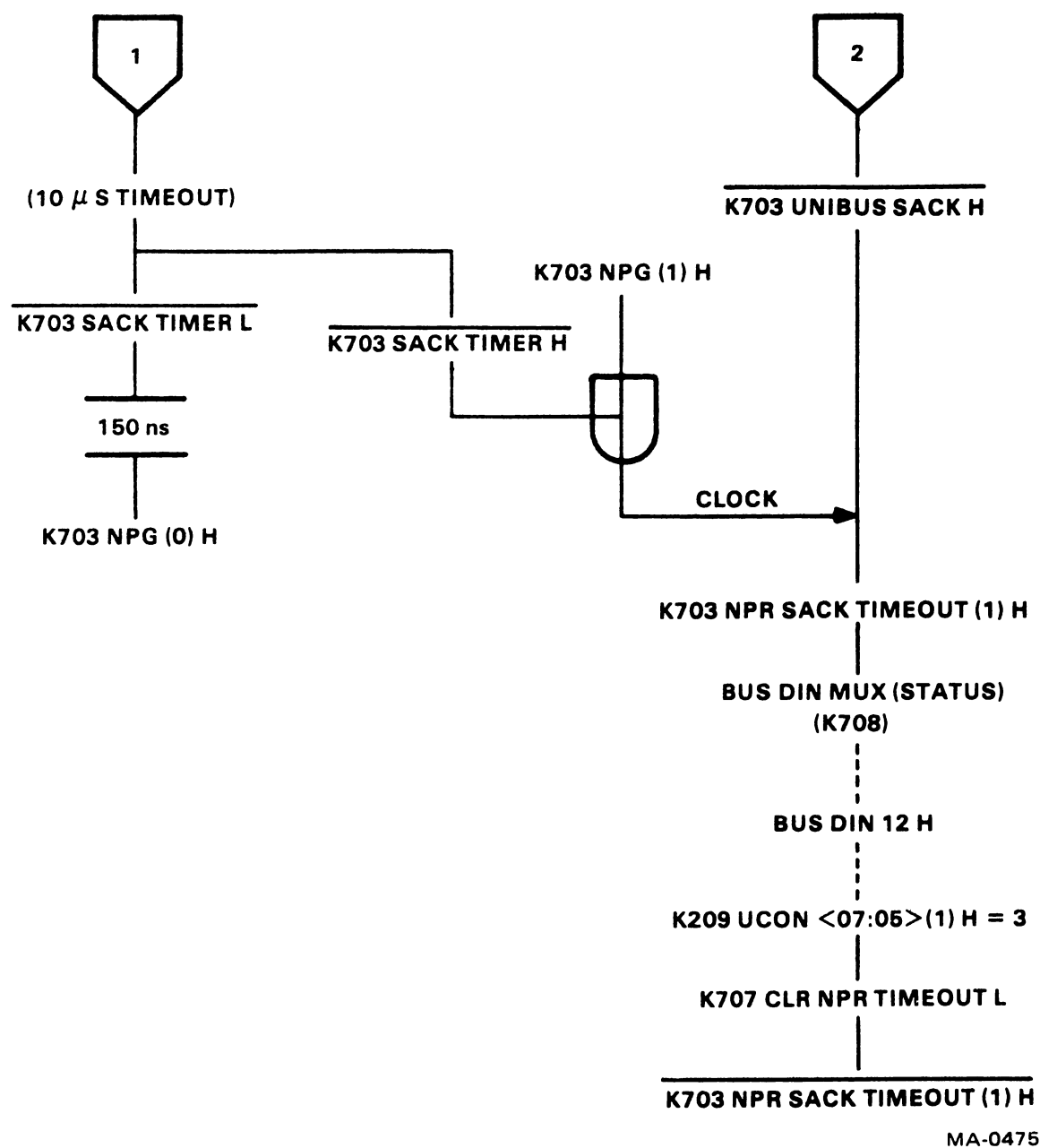


Figure 1-15 NPR Flow (Sheet 5 of 5)

When SACK is received in response to the NPG, the processor negates K703 SACK TIMER H and BUS NPG H. K703 DELAYED SACK H, asserted at this time, maintains the assertion of K606 INH BG L and K606 BUS IN USE (1) H, thus inhibiting BR arbitration and processor use of the bus. At the device end, the NPR is negated and BUS BBSY L asserted. This makes the requesting device bus master. At some point after the assertion of BUS BBSY the device drops SACK and, after a delay, K606 INH BG L is negated; BR arbitration can now be resumed. (A BG can be asserted for an accepted BR but the bus is not free until BUS BBSY is negated from the current operation.)

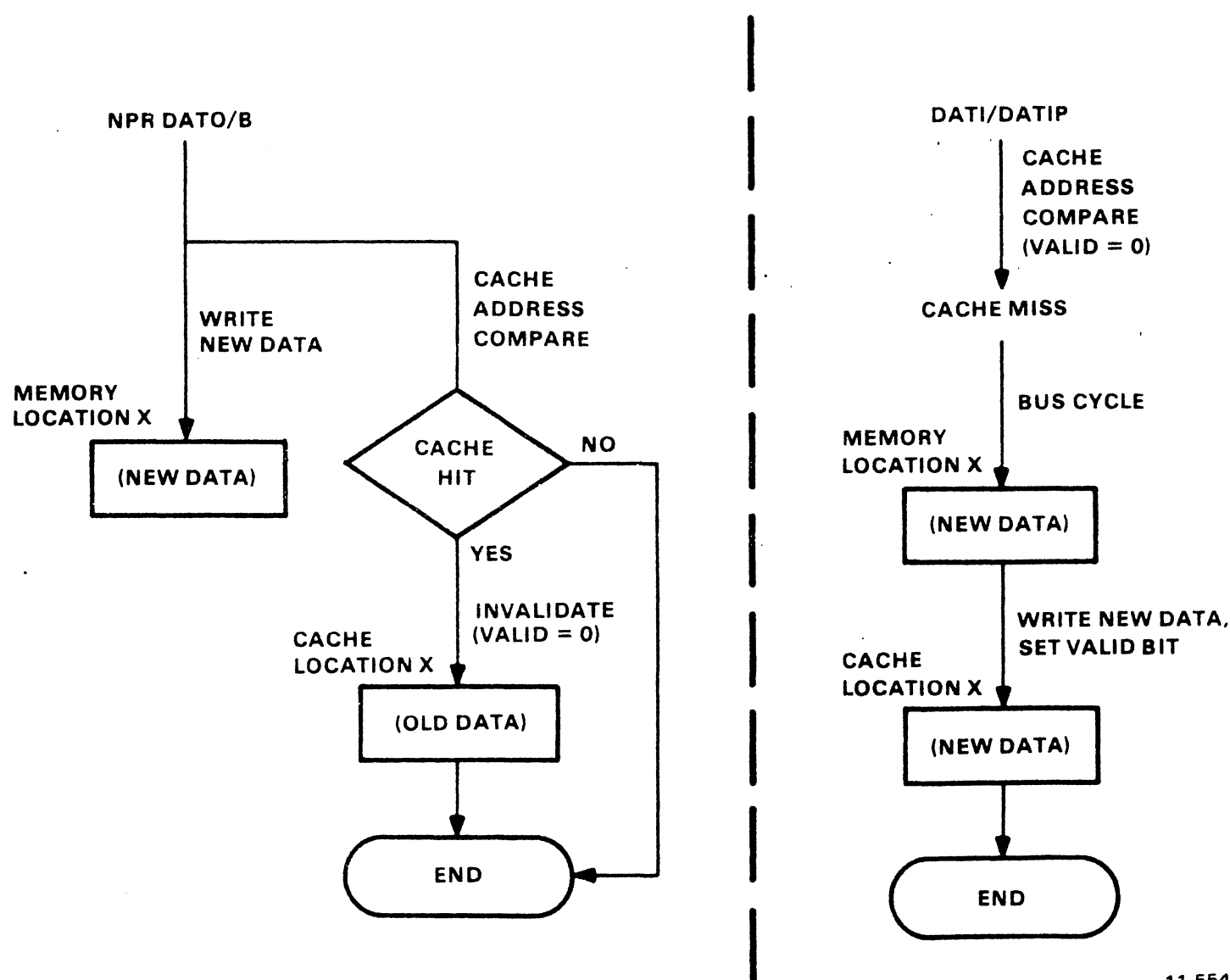
The NPR device places the address, control bits, and data (if DATO/B) on the respective Unibus lines and asserts BUS MSYN L. K607 CLK UNIBUS ADRS H (Figure 1-9), asserted in U CYCLE B, clocks the address into cache [K515 NPR ADRS (17:00) (1) H] and sets the NPR DATO flip-flop if a DATO/B is decoded. The CR (NPR) SYNC flip-flop is set at P3 time. (This is the first clocked event since BUS IN USE was set after the NPR was received.)

IN U CYCLE C, an NPR cache request is generated and the NPR address is compared with the cache tag address. Then, in U CYCLE D, a cache write pulse is produced if a hit is detected. This causes a 0 to be stored in the valid bit for this cache location; the cache location is invalidated. Note that only a DATO/B hit invalidates the cache; a DATO/B miss or a DATI or DATIP operation bypasses (Figure 1-15, off-page connector D) this procedure.

Figure 1-16 shows this in an abbreviated form: the cache location is invalidated if there is a hit when an NPR DATO/B is allowed. If a subsequent DATI or DATIP addresses the same location, a miss occurs because the valid bit equals 0. A bus cycle then goes to the backing store, retrieves the updated data that was the result of the NPR DATO/B, stores it in the cache location, places a copy in the CSP MD register, and sets the valid bit for the cache address.

The memory asserts SSYN to indicate completion of its task and, after the cache location is invalidated (Figure 1-15, U CYCLE D), BUS SSYN L is asserted. In U CYCLE E, various signals are returned to their quiescent state. Finally, the NPR device negates MSYN and BUS BBSY; the bus is again available to the processor and other devices.

If SACK is not returned in response to NPG within 10 μ s, the single shot times out and K703 SACK TIMER, going low, clocks the NPR SACK TIMEOUT flip-flop. K703 NPR SACK TIMEOUT (1) H, input to the BUS DIN MUX (STATUS), asserts BUS DIN 12 H on the DIN lines to indicate the NPR timeout condition. Subsequent microcodes set K209 UCON (07:05) (1) H = 3 to assert K707 CLK NPR TIMEOUT L, which clears the NPR SACK TIMEOUT flip-flop.



11-5540

Figure 1-16 NPR Invalidation

1.7.3 Jam

A jam is a hardware induced transfer of control to microaddress 777, which is the start of the jam subroutine. The conditions that cause a jam are shown in Figure 1-17. With the exception of micro-break and internal address, all the conditions are caused by hardware-detected errors within the machine. Table 1-9 lists where jam conditions are described elsewhere in this manual and in the *WCS Maintenance Manual*.

Each event that causes a jam, with the exception of internal address, sets a bit in the log jam register, which is located in the CSP. This information is later read by the jam routine to determine the highest priority event that caused the event. Control is then passed to the routine that services that event.

Initially, after the microcode goes to address 777, the jam routine determines if a power up operation has taken place. One of three things occurs if it has:

- The machine will be completely initialized and the macroprogram started at the address in the console slide switches. This occurs if there are no grounded wires on BUS DIN.
- The machine will be initialized except for those parts that require a large number of microcycles and then the macroprogram is started at the address set in the console slide switches. This occurs if BUS DIN 14 (only) is grounded.
- The console microdiagnostic program is entered. This occurs if BUS DIN 13 (only) is grounded.

41 Grounding jumpers should be connected to the BUS DIN lines only when power is turned off. This is because BUS DIN is a tristate bus and can have signals actively applied to it; some gates may be damaged if jumpers are connected while power is on.

To prevent this damage, the jam routine disables all tristate drivers on BUS DIN. This is done by selecting NULL UCON, which disconnects all devices that use UCON signals for enable and/or select inputs. The BUS DIN is then read to check if any wires are being pulled down by jumpers. If BUS DIN 13 or 14 is grounded, the program waits until the jumpers are removed before proceeding. (Refer to the overviews of power up, NO SWP, and console diagnostic for details of actions taken.)

If a power up has not taken place, the residual control register (K408) is reset so that the shift register (K408) can be loaded [BUS BIN (13:12) = 00 result in a parallel load from the arithmetic logic unit (ALU)]. CLEAR FLAGS cannot be used to clear the residual control register because it would also clear any error flags that should be logged in the event of an error.

The jam routine now determines if an internal address (K705 INTERNAL ADRS H) caused the jam. The state of K705 OTHER JAM H is checked by BUT (24) to eliminate the possibility that one of the other jam events caused the jam. It can be assumed that an internal address has been referenced if K705 OTHER JAM H is low. If this is the case, control is passed to the internal address reference flow.

An error of some type has occurred (or U Break set) if an internal address has not been referenced. BUT (27) selects the K306 PREFETCH*JAM (1) H input to the U Branch MUX to determine if the error occurred during a macroinstruction execution when a fetch overlap was taking place (Paragraph 4.7). If this is the case, the PC is decremented so that it is pointing at the instruction following the one that caused the error.

The jam routine next checks log WHAMI register bit 1, the error trap in progress bit, to determine if an error trap was in progress when the jam occurred. If WHAMI bit 1 = 1, control is passed to the flow that handles EOE (error on error). If WHAMI bit 1 = 0 when checked, it is set by the program. The shared trap flow clears this bit if it is successfully completed.

The jam microcode next reads the log jam register to determine the highest priority event that caused the jam; events are serviced according to the priority indicated in Figure 1-17, from the highest (U Break) to the lowest (memory parity error). Control is now passed to the appropriate error handling routine. Lower priority events are bypassed (i.e., never serviced) if multiple jam conditions are detected at the same time.

Table 1-9 Jam Condition Descriptions

Jam Condition	Description Location (Paragraph Number)
Microbreak	4.11
Odd Address Error	2.2.3.5
Red Zone Error	2.2.5
MM Abort	2.2.4.4.2
Illegal Address	2.2.3.5
Cache Error	2.2.3.5
SSYN Timeout	1.6.3
Memory Parity Error	1.6.3
Internal Address	1.6.1

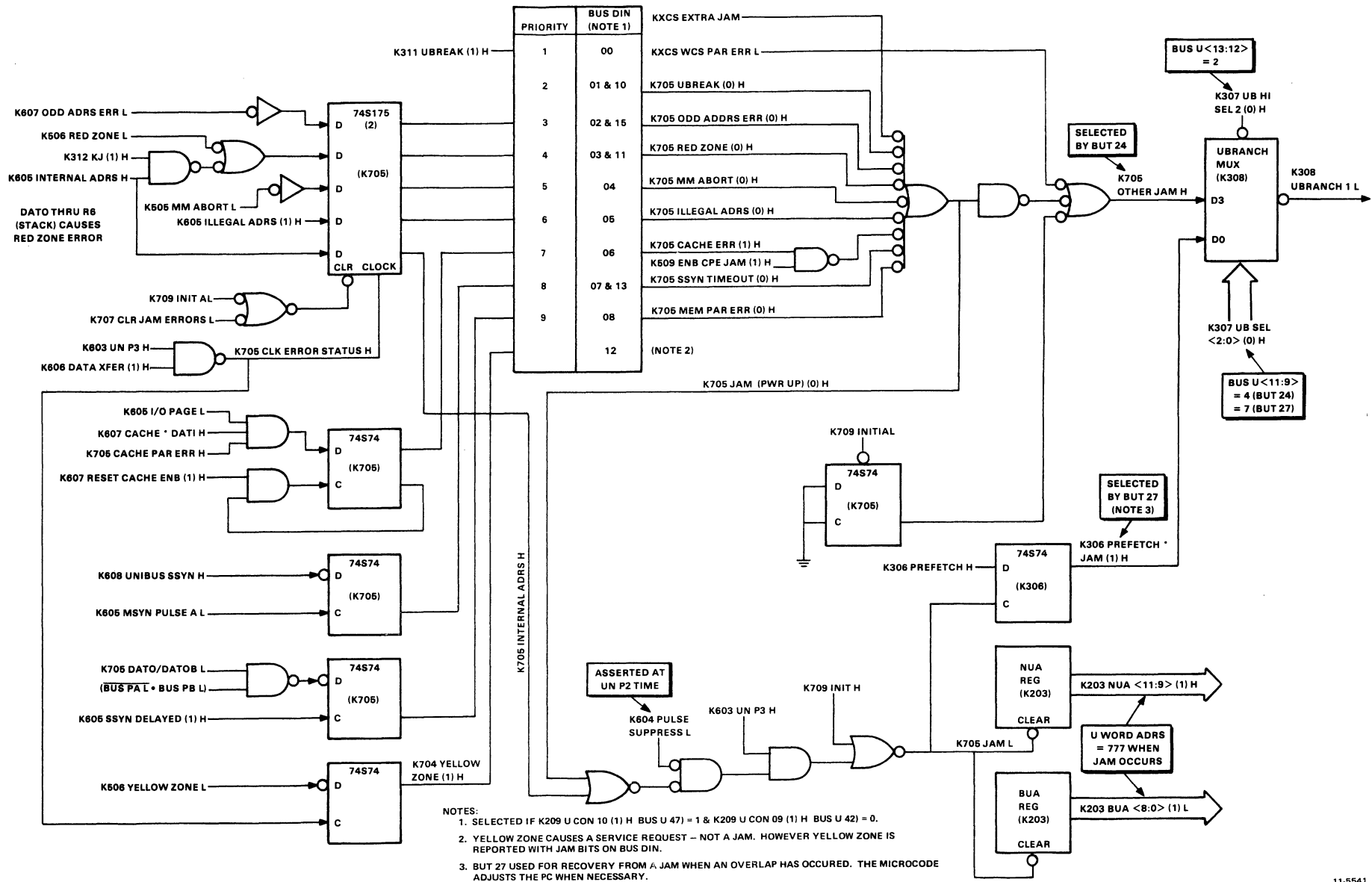


Figure 1-17 Jam Logic

CHAPTER 2

KT CACHE MEMORY

DETAILED LOGIC DESCRIPTION

2.1 CACHE MEMORY CONCEPTS

2.1.1 Scope

This paragraph explains the purpose of cache memory systems and describes various methods used to implement such systems. Parameters and strategies involved in cache memory design are introduced, described, and analyzed to facilitate the reader's understanding of the specific cache implemented in the PDP-11/60 system.

2.1.2 Introduction

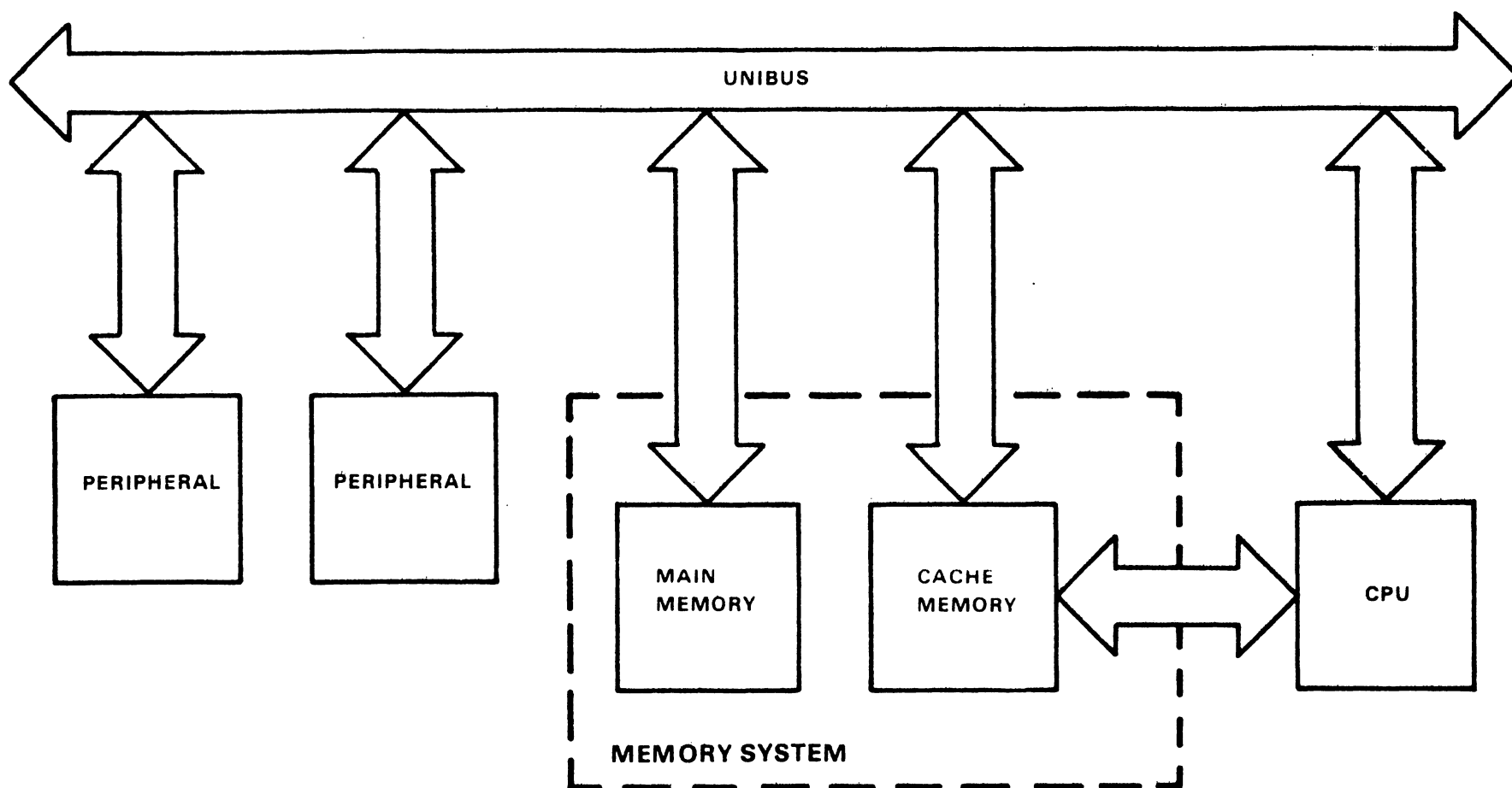
43 A cache memory is a small, high-speed memory that contains a copy of automatically selected portions of main memory data. Its purpose is to provide faster access to instructions and data. In effect, a cache memory simulates a system having a large amount of fast memory. To accomplish this, the system relies on a relatively small fast memory (the cache); a larger, and slower, main memory; and the anticipated behavior of the operational program.

A system using a cache memory has the same appearance to the user (program) as a conventional system containing only slower core memory. Although the cache operation is transparent to the user, program execution is considerably faster. This is the only noticeable difference in the two memory systems.

2.1.3 Cache Memory Organization

Figure 2-1 shows the relationship of a typical cache memory to the other system components.

All of the data is stored in a main memory (called the backing store) and a copy of some of the data is stored in cache memory. If most of the time the needed data is in cache, then the program will execute quickly, slowing down only when it is necessary to access main memory. Conventional mixed semiconductor-core memory systems attempt to achieve this goal by having the programmer guess beforehand which sections of the program should go in each memory. This is often awkward, and usually only moderately successful. The cache memory system tries to achieve the same goal by automatically, dynamically shuffling data between the two memory types in a way that gives a high probability that useful data will be in the fast memory. All of the following discussions of cache organizations and strategies are intended to show implementable methods of shuffling data so that the data most likely to be needed next will be in the fast memory instead of the slower main memory.



11-5553

Figure 2-1 Relationship of Cache to Main Memory and the CPU

Of primary concern is the time required to execute the program. This time is affected by two factors:

1. The relative speeds of fast and slow memory
2. The percentage of times memory references will find the data in cache and therefore allow faster execution.

When the sought for data is found in cache, a hit is said to occur. A miss takes place when the data is not found in cache. The CPU must then access main memory for the data.

2.1.4 Program Locality

A cache memory works because it can usually predict successfully which words a program will require soon. If programs used words completely at random from all of memory, it would be impossible to predict which words would most likely be needed next. Under these circumstances, a cache memory system could perform no better than a conventional mixed memory system with a small amount of semiconductor memory.

Fortunately, programs do not generate random addresses. Instead, programs have a tendency to make most accesses in the neighborhood of locations accessed in the recent past. This is the basis of the principle of program locality. The fact that programs display this type of behavior makes cache memory systems possible.

An understanding of why the principle of program locality is true can be obtained by examining the small scale behavior of typical program data structures. Code execution itself generally proceeds in straight lines or small loops; the next few accesses are most likely to be within a few words ahead or behind. Stacks grow and shrink from one end, with the next few accesses near the current top. Character strings and vectors are often scanned through sequentially.

The principle of program locality is a statement of how most programs tend to behave, not a law which all programs always obey. Jumps in code sequences, seemingly random access of symbol tables by assemblers, and context switching between programs are examples of behavior that can adversely affect the locality of addresses generated by a processor. The process of guessing which words a program will reference next can never be completely successful. The percentage of correct guesses is a statistical measure affected by the size and organization of the cache, the algorithms it uses, and the behavior of the program driving it.

2.1.5 Direct Mapping Cache

The direct mapping cache allows each block from main memory only one possible location in fast memory (Figure 2-2). Consider each incoming address as being made up of three parts. The first part starts at bit 0 and contains enough bits to specify which byte out of a block is being requested. The next field, called the index field, starts where the first field leaves off and contains enough bits to specify any block in fast memory. The third field, called the address field, contains the rest of the bits.

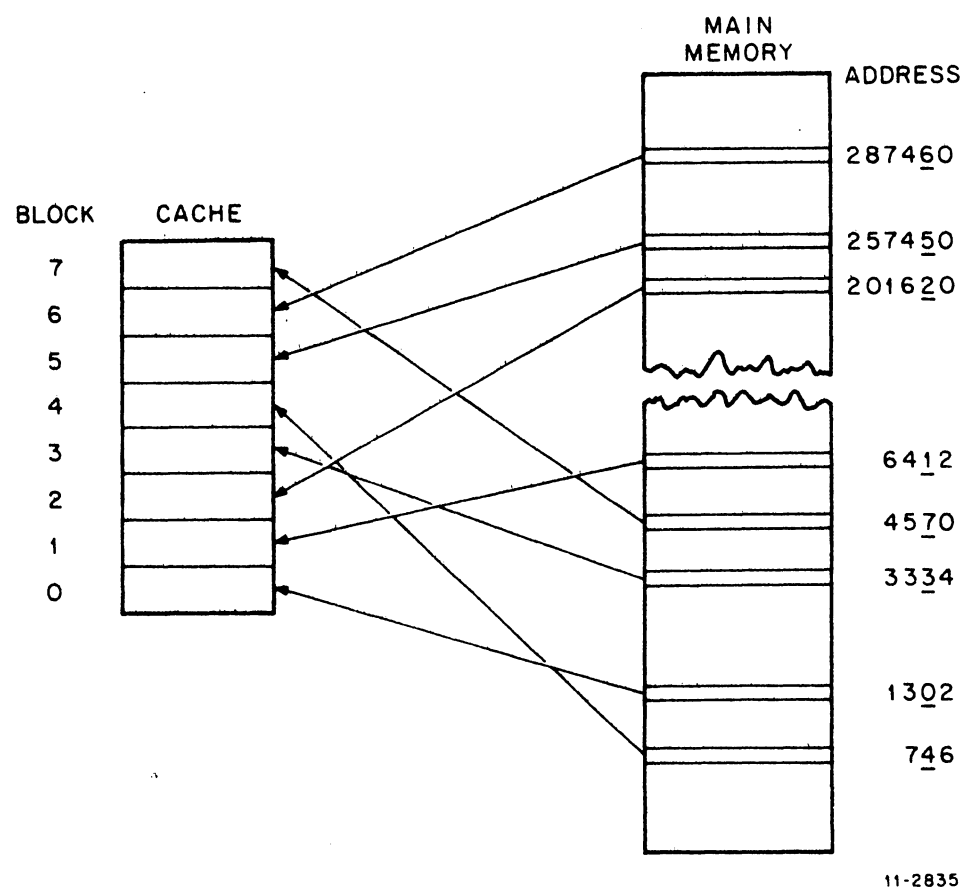


Figure 2-2 Direct Mapping Cache Memory System

As an example, consider an 18-bit PDP-11 byte address as input to a 256-word, 4 word per block, direct mapping cache. (Thus, this cache would be 4 words wide and 64 blocks deep. Assume that four words per block allows us to break down the address conveniently, using octal notation.) As illustrated in Figure 2-3, the word field in this case comprises bits 2, 1, and 0, where bit 0 indicates the byte, and bits 2 and 1 indicate the word. The index field comprises bits 8 through 3 and indicates the block. The address field comprises bits 17 through 9.

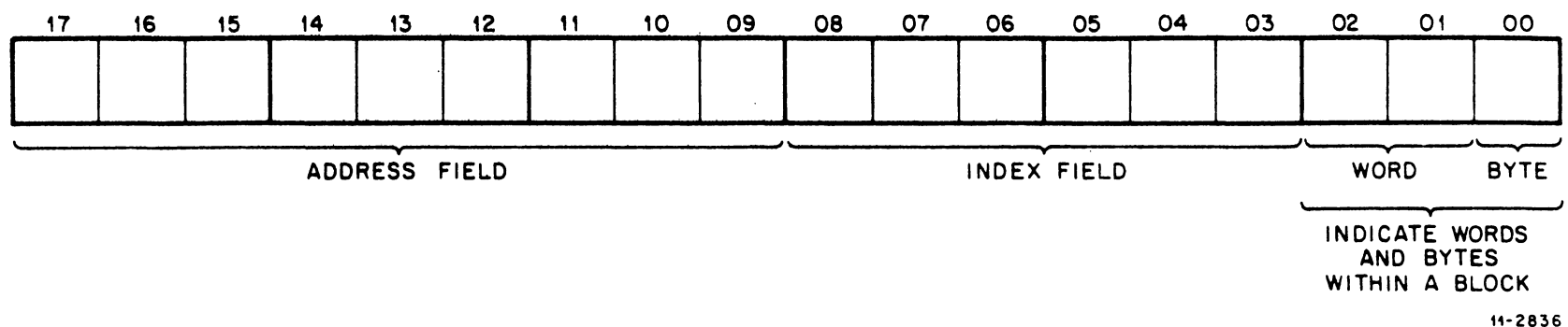


Figure 2-3 18-Bit Byte Address Breakdown (4 Words per Block, 64 Blocks)

If the processor requests word 274356, the cache controller looks at the address that goes with the information currently in block number 35 in fast memory. If this address field is 274, the controller sends the third word in that block to the processor. If the stored address field is not 274, the controller must fetch block 27435 from main memory; transmit the third word in the block to the processor; load the block into block 35 of fast memory, replacing whatever was there previously; and change the address field stored with block 35 to 274.

Any address whose index field is 35 will be loaded into block 35 of fast memory. Therefore, this is the only place the cache controller has to look if the processor requests the data from an address whose index field is 35.

Notice also that only the address field of the address need be stored with each block, because only the address field of the address is required for comparison. The index field need not be compared because anything stored in fast memory block 35 has an index field of 35. The word field need not be compared because if the block is there, every word in the block is there.

This is how the direct mapping cache uses inexpensive direct addressing of fast memory to eliminate almost all comparison operations.

2.1.6 Write-Through and Write-Back

Assume that the following sequence of events occurs. First, the processor does a read of location 200, resulting in the block containing this address being copied into fast memory. Then, the processor writes new data into location 200 updating this location in fast memory. Next, the processor does a reference, which causes the cache controller to over-write the block in fast memory containing location 200. If the processor reads location 200 again, the obsolete data in main memory will be loaded into fast memory. This is unacceptable, and two methods have been devised to deal with the problem. The methods are called write-through and write-back.

With write-through, whenever a write reference occurs, the data is not only stored in fast memory, but is also immediately copied into main memory. This means that the main memory always contains a valid copy of all data. If the controller wants to over-write a block in fast memory, this can be done immediately, without losing any data.

The advantages of write-through are its relative simplicity and the fact that the main memory always has correct data. The primary disadvantage is some reduction of speed due to the need to access the slow memory on every write reference. This is offset somewhat by the fact that write references are a small fraction of all references to memory. In addition, the cache does not have to wait for the main memory to finish before starting the next cycle.

Since a reasonable design would only cycle the memory being written into and not all the parallel memories in main memory, the system should not even be held up by multiple sequential writes. However, some fraction of the time, the system will have a read miss following a write, or two writes to the same memory stack within main memory, and then the system must wait. This causes the system to run slightly slower than first-order estimates would indicate.

2.2 PDP-11/60 CACHE

2.2.1 Scope

This paragraph describes the specific cache memory that has been implemented in the PDP-11/60 system. The reader should be familiar with cache concepts, classifications, and definitions described in the previous paragraph.

2.2.2 PDP-11/60 Cache

The PDP-11/60 memory system consists of 1K (1024_8) cache memory and up to 124K of main memory or backing store. (The upper 4K memory addresses, 124–128K, are reserved for peripheral address.)

The cache memory is a 27-bit 1024 word direct mapping cache (Paragraph 2.1.5). It consists of 27 1K random access memory (RAM) chips, types 93415A and 934150. Each 27-bit cache word contains a tag field and a data field (Figure 2-4). The 9-bit tag field is subdivided into three groups of data: tag address (7 bits), valid bit (1 bit), and tag parity (1 bit). The 18-bit data field consists of two 8-bit data bytes, each with a parity bit.

The tag address bits and a tag parity bit are derived from the seven high-order bits of the physical address, i.e., the address of the bus data in the cache data store. They are stored in the cache tag store and the tag parity store. The valid bit denotes that this cache location contains correct data – data that is an exact copy of the data at the associated address in main memory. The valid bit is cleared on power up, and set when data is loaded into cache (Paragraph 2.2.3.5).

The high and low data bytes are stored in the cache data store. Two parity bits are generated from the two bytes; they are stored in the data parity store.

The ten low-order physical address bits (10:01) serve a different purpose in the cache operation. They are used to select one of 1024 locations in each of the five cache subsections whenever cache is accessed, i.e., when new data and its tag address are written into cache, or when a read operation searches cache for the needed data.

The PDP-11/60 cache memory is a direct mapping cache. However, unlike the direct cache previously described (Paragraph 2.1.5), the PDP-11/60 cache contains only one word blocks. Consequently, it does not possess a look-ahead feature. Figure 2-5 illustrates how the mapping scheme is implemented.

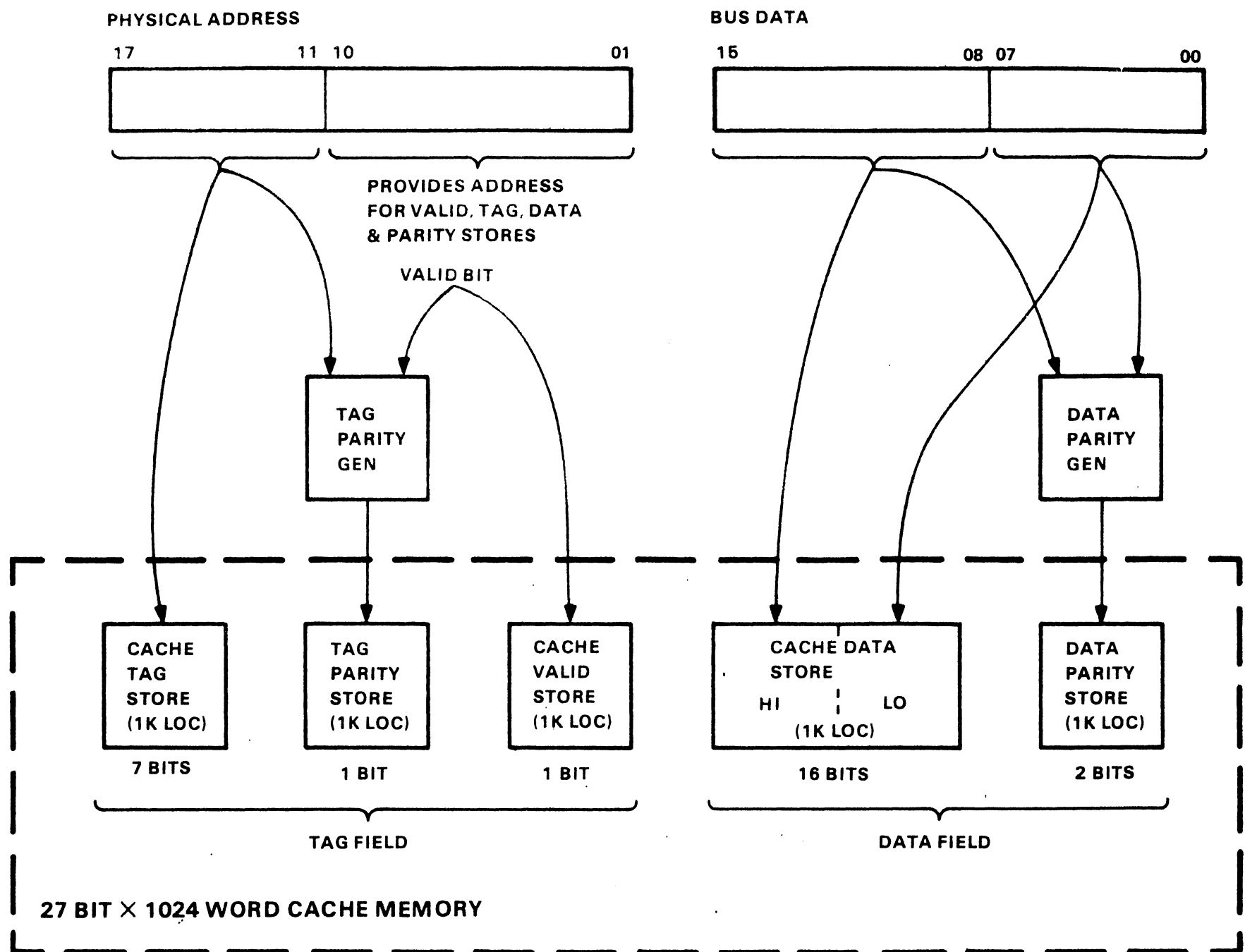
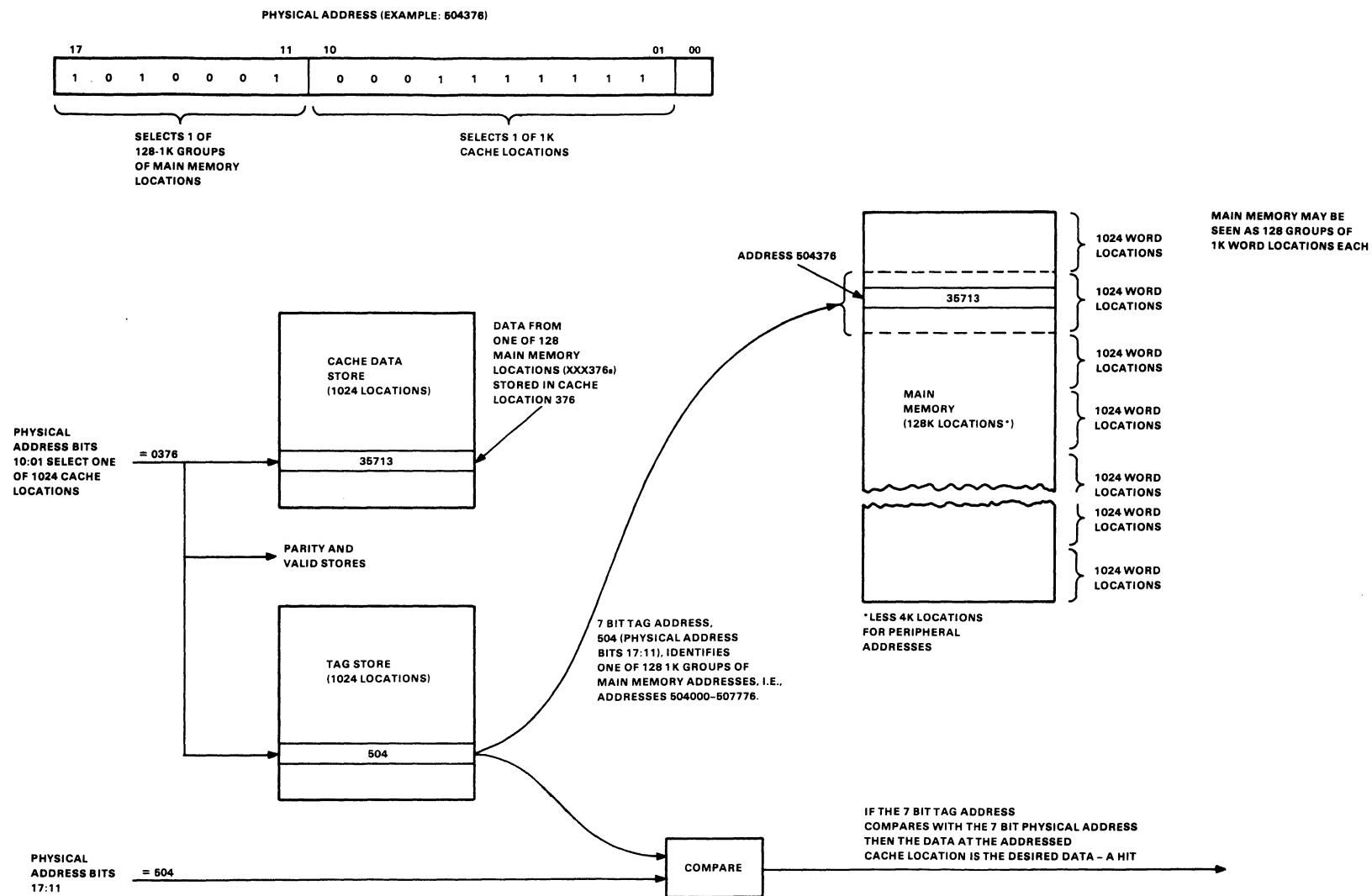


Figure 2-4 Cache Tag and Data Fields

Physical address bits 10:01 are used to access one of the 1024 cache data store locations. The same location in the tag store will contain a 7-bit tag address. The tag address consists of physical address bits 17:11 that were previously loaded into the tag store. Tag addresses define one of 128 1K groups of main memory locations. If the seven high-order physical address bits in the address used to access cache are equal to the 7-bit tag address stored in cache, then the contents of the selected cache location contain the sought for data. This is called a "hit." If the two 7-bit addresses do not compare, a "miss" is said to have occurred; the data in cache is not the data that is needed.

In Figure 2-5 it is assumed that the physical address is 504376. The ten low-order bits (0376) access one of the cache locations, while the seven high-order bits (504) are compared with the tag address. If they compare, as they do in this example, the data (35713₈) at cache location 376 is a copy of the data at main memory address 504376; a hit has occurred.

48A



11-5557

Figure 2-5. Cache Addressing (Simplified)

Cache location 376 could, at other times, contain data from one of the other 1K main memory blocks. For example, a copy of the data at address 266376, or address 470376, could be stored in this same cache location. However, any physical address other than 504376 would cause a miss if the tag address is 504.

Figure 2-6 shows the direct mapping characteristics of the PDP-11/60 cache. Data transfers are similar to that in the typical direct mapping cache (Figure 2-2) previously described. The 1K cache is addressed by physical address bits 10:01, addresses 0000₈ through 3776₈. Each cache location consists of 27 bits, 16 of which are a copy of the data in a main memory location. The address fields for cache and the 1K main memory groups overlap when viewed as a single octal address. For example, the data at main memory location 507774 will be copied in cache location 3774, because cache is accessed by the ten low-order bits of the physical address. The third most significant octal digit of the physical address is shared by both address fields.

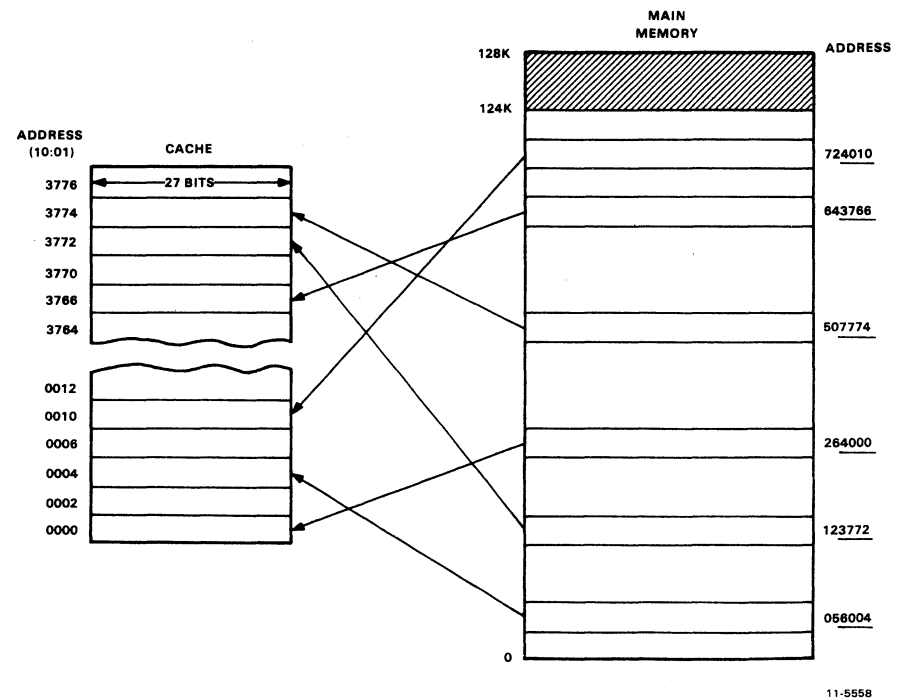


Figure 2-6 Direct Mapping in the PDP-11/60 Memory System

2.2.3 Cache Data and Address Paths

The cache tag and data store subsections, along with the addressing and data paths, are shown in Figure 2-7.

The physical bus address (PBA 17:01) is developed in the ALU, described in Paragraph 3.2. Either this address, or an NPR address if a DATO/B operation is initiated by an NPR device, is applied to the various cache stores via the physical address multiplexer (PA MUX). If K605 SEL PA MUX H is low (not asserted), the PA MUX "A" inputs [NPR ADRS (17:01) (1) H] are selected as the output; if the select signal is high (asserted), the B inputs are present at the PA MUX output.

2.2.3.1 Physical Address Bits 17:11 – The high-order physical address bits, PA MUX (17:11), serve several different functions. In the tag store they provide the data input, i.e., a new tag address, at the time the contents of a cache data store location are changed. At the same time, a new tag parity bit is produced by the tag parity generator. The state of this bit, K510 TAG PAR IN H, is based on the bit configuration of PA MUX (17:11) L and the valid bit, K602 VALID H. To maintain even parity, K510 TAG PAR IN H is asserted if the input bit pattern is odd. Conversely, the signal is low if the input is even. Incorrect (odd) parity can be written for diagnostic purposes through the use of the write wrong parity signal, K514 WT WRONG PAR (1) H.

48c A third use for the seven high-order physical address bits is in the address compare circuit. PA MUX (17:11) L is compared with the tag address, K511 CACHE ADRS (17:11) L, in the ADRS COMPARE chips (K511) to determine if the data in a certain cache location is the data that is needed. (This operation is described in Paragraph 2.2.2.) Two high outputs are produced by ADRS COMPARE if the inputs match, i.e., the sought for data is in cache and the valid bit, K510 CACHE VALID H, is asserted for this address. The valid bit indicates that the cache data is an accurate copy of the current data at the equivalent main memory location. (The valid bit acts as a function select, A=B, in the LOW ADRS COMPARE chip.) K511 HI ADRS COMP H is asserted if the three (17:15) address inputs compare, and K511 LOW ADRS COMP H is asserted if the four (14:11) address inputs compare.

2.2.3.2 Cache HIT – K511 HI ADRS COMP H and K511 LOW ADRS COMP H are ANDed with the data and tag parity check signals (K511 TAG PAR OK H, K513 HI BYTE PAR OK H, and K512 LOW BYTE PAR OK H) to produce the "hit" signal, K514 HIT L. The hit signal indicates to the bus control that the requested data is in cache (address compare), that it is an exact copy of the data at the equivalent main memory location (valid bit), and that the tag and data parity checks are good. A "miss" occurs when the hit signal is not asserted when cache is accessed.

K574 HIT L can be inhibited for diagnostic purposes by K514 FORCE MISS L (Figure 2-8). This might be the case if it were necessary to bypass cache and force all memory requests to access main memory. K514 FORCE MISS L is produced by K514 DISABLE CACHE L (grounding pin F05V2 on the backplane), either one of two force miss inputs from the cache control register (CCR) (Paragraph 2.2.3.7), or K409 INVALIDATE (0) H from the data path module. The K409 INVALIDATE (0) H signal also negates the valid bit, K607 VALID H (Paragraph 2.2.3.5).

The hit signal is sent to the timing module in bus control (Paragraph 1.2) and to the cache hit register (Figure 2-6). This hex D-type flip-flop register records the results of the last six CPU references to cache: were they hits or misses? NPR hits and misses are not registered because the clock signal, K607 CLK HIT H, is not asserted when cache is accessed as the result of an NPR request. A one at a hit register output indicates a read or write hit; a zero indicates a read or write miss. The lower numbered bits are for more recent cycles. All six register outputs can be sent to the read only register multiplexer (K507) (Figure 2-9) and then the DIN lines for diagnostic purposes.

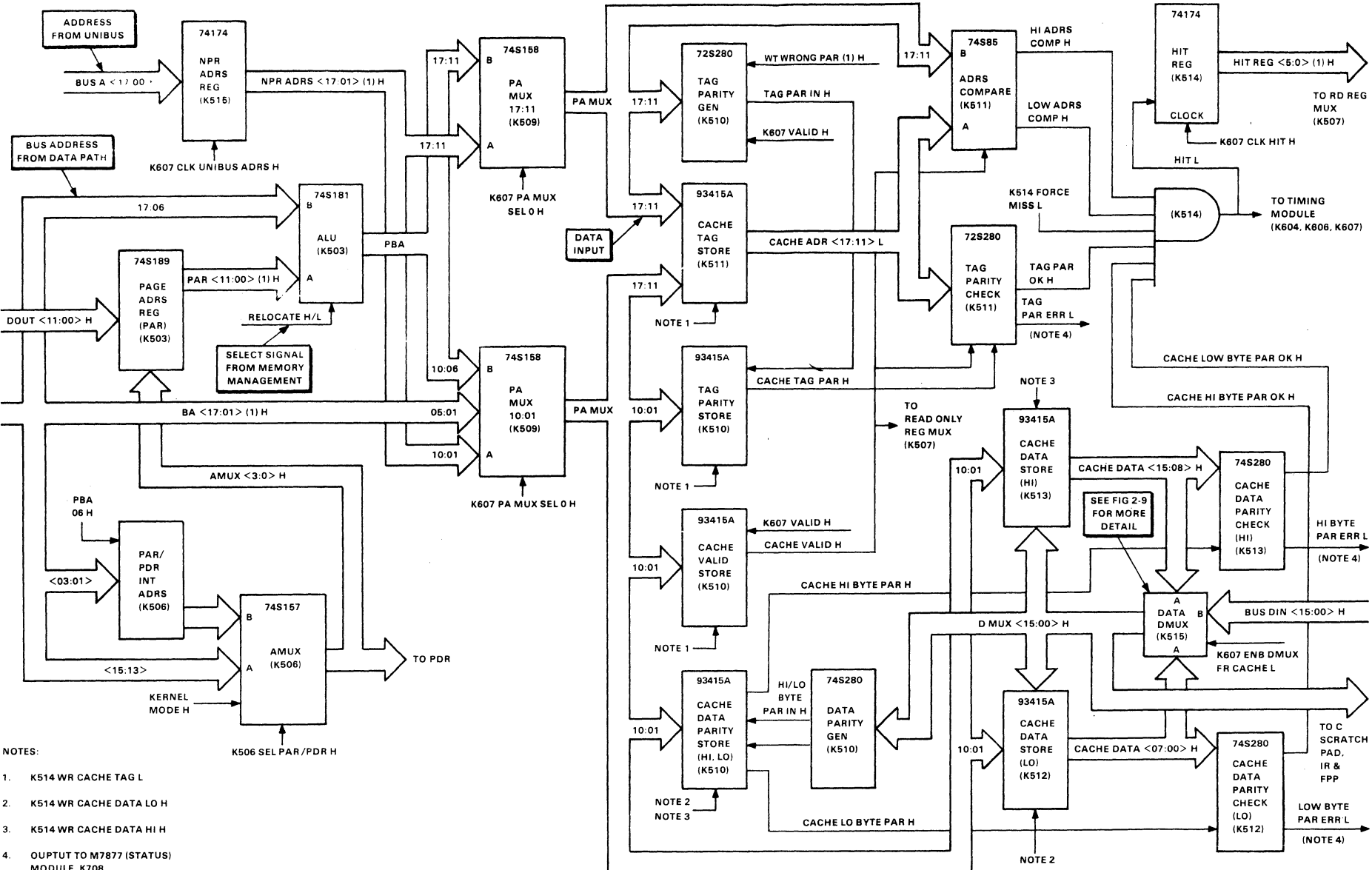


Figure 2-7 Cache Addressing and Data Paths

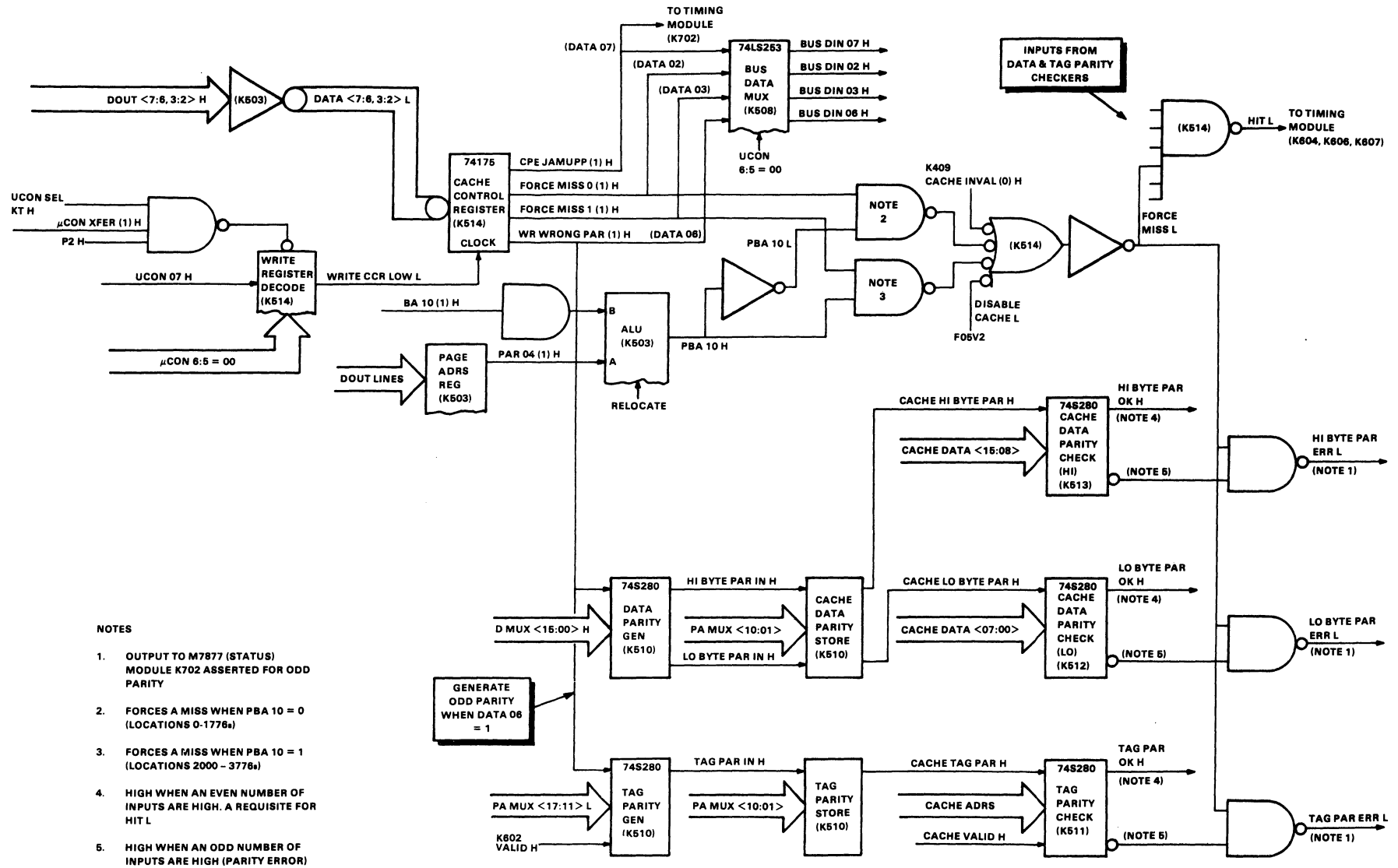


Figure 2-8 Cache Control Register (CCR)

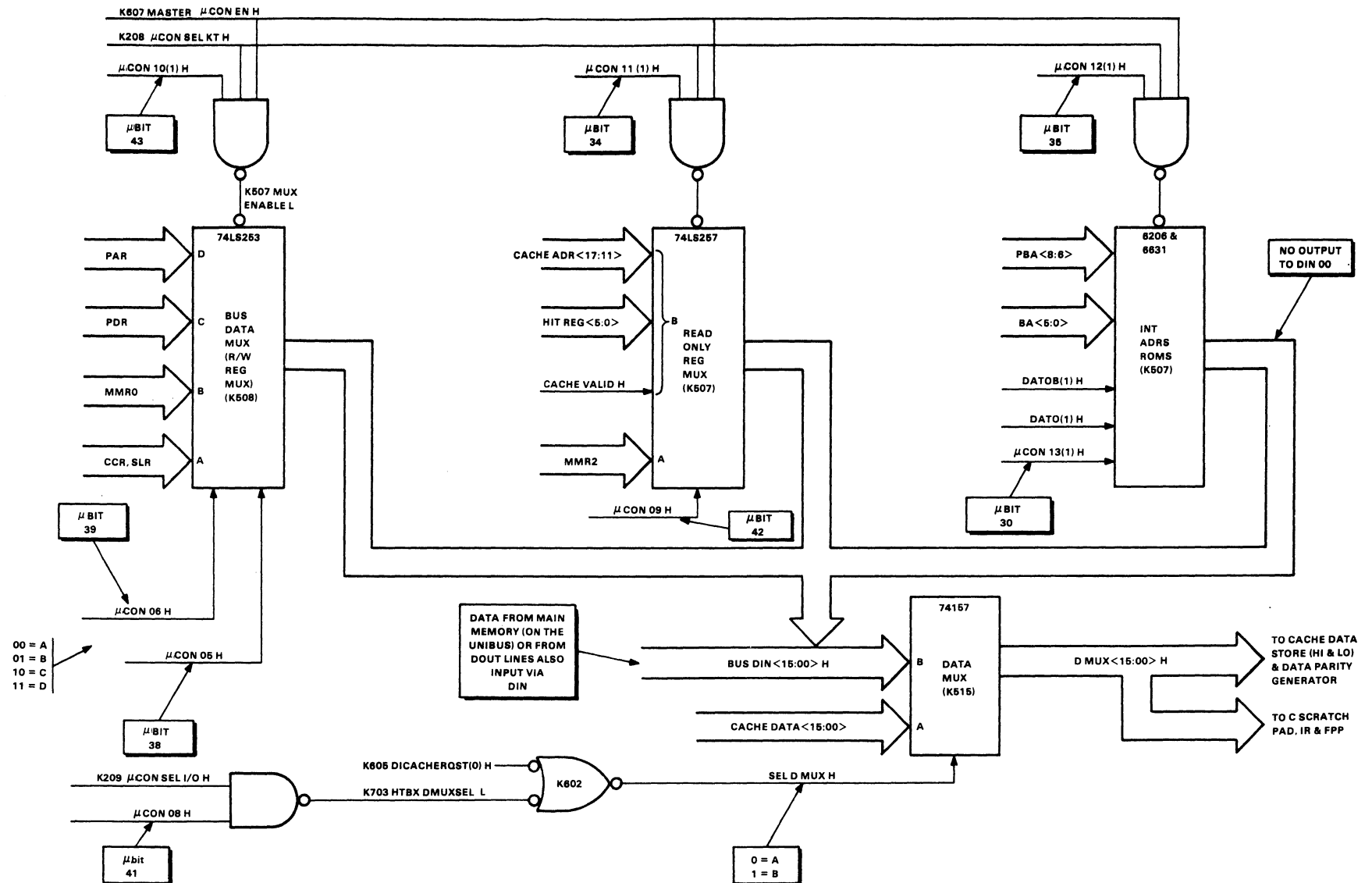


Figure 2-9 Data Multiplexer Inputs

2.2.3.3 Physical Address Bits 10:01 – The low-order physical address bits, PA MUX (10:01) L, are used as the address inputs to five cache stores: data (high and low bytes), data parity (high and low bytes), tag, tag parity, and valid bit (Figure 2-7). Each store has 1024_{10} locations; a total of 27 bits at each location are divided among the five stores (Figure 2-4). The chips used in the stores are always enabled (chip select inputs tied to ground). Consequently, the output is present at the time a location is accessed, i.e., all store outputs are available as soon as the PA MUX address outputs are selected by K607 PA MUX SEL 0 H.

2.2.3.4 Physical Address Outputs to Bus Control – K503 PBA (17:06) and BA (05:01) are routed to the bus control logic. Here they provide the inputs to a ROM (shown on K605) that determines if the physical address is an internal address. The BUS DIN MUX (STATUS) (shown on K708) also receives the physical address – along with error conditions and console switch data. UCON (10:09) H (U word bits 47,42), when equal to 11, select the physical address as the multiplexer output to the DIN lines. Main memory, via the Unibus, also receives the physical address (K608). This is shown in Figure 2-13.

2.2.3.5 Store Outputs – The outputs from the stores can be classified broadly as either internal control bits, such as the parity bits, valid bit, and tag address; or requested data – the 16-bit data word read out to the data path, processor control, or the floating point processor.

49 In the address compare operation (Paragraph 2.2.2), the 7-bit tag store output, K511 CACHE ADRS (17:11) L, is compared with the seven high-order physical address bits to determine if the needed data is in cache. This same output from the tag store, along with the tag parity bit (CACHE TAG PAR H) and the valid bit (CACHE VALID H) for the applicable cache location, is applied to the tag parity checker (Figure 2-7). If an even number of these nine inputs are high, the signal K511 TAG PAR OK H, one of the qualifiers for K514 HIT L, is asserted. K511 TAG PAR ERR L is asserted in the event an odd number of bits are input to the parity checker (Figure 2-8). This error signal, which can be inhibited by K514 FORCE MISS L, is one of the inputs to the jam logic (K705) whose outputs invoke the microcode jam or service routines (Paragraph 1.7.3). (Both K511 LOW ADRS COMP H and K511 TAG PAR OK H are low when K510 CACHE VALID H = 0 for the selected address; both of these signals, when low, inhibit K514 HIT L to cause a cache miss.)

The valid bit, K510 CACHE VALID H, for the selected cache location in the valid store (Figure 2-4), is output to three destinations: address compare (Paragraph 2.2.2), tag parity checker, and the read only register multiplexer (Figure 2-9). The valid bit is derived from K607 VALID H, which is high for DATO or DATOB operations. K409 INVALIDATE (0) H (microword bits 24,22:20 all = 1 and 23 = 0) or 607 CACHE*NPR L forces the valid bit low.

The cache is filled with incorrect data after power is brought up. This requires an invalidation sequence in which all valid bits are negated. In this operation, each of the 1024 cache locations is accessed and a zero is written in the valid store. During cache invalidation, which is initiated by bus control, the invalidate bit [K409 INVALIDATE (0) H] is asserted in the data path module. In the cache, INVALIDATE (0) H produces K514 FORCE MISS L (Figure 2-8) and in bus control it negates K607 VALID H and resets K606 BUS RQST (PROC) (1) H and K606 BBSY (PROC) (1) H so that the cache invalidation sequence does not use the Unibus.

Another need for invalidation occurs when an NPR on the Unibus results in the modification of data at a main memory location. K607 CR (NPR) SYNC (1) H enables K607 CLK UNIBUS ADRS H, which latches the NPR address on the Unibus into the cache address logic on K515 (Figure 2-7). K607 CACHE RQST (NPR) (1) H negates K607 PA MUX SEL 0 H which, in turn, selects the NPR address [NPR ADRS (17:01) (1) H] as the PA MUX input (K509). If the high-order bits of this address match the tag address in the address compare logic, a hit occurs. Bus control must then invalidate that location so that it will force a backing store cycle to get the new NPR data when it is referenced again. To accomplish this, K607 CACHE*NPR L negates K605 VALID H.

Another store output is the data from the low and high byte cache data stores, K512 CACHE DATA (07:00) H and K513 CACHE DATA (15:08) H. These two data bytes are applied to two parity checkers and the DMUX (K515). The two parity checkers, shown on K512 and K513, also receive the two parity bits from the data parity store. The low byte parity bits, K510 CACHE LO BYTE PAR H, and cache data bits 07:00 are evaluated in the low byte checker. Likewise, K510 CACHE HI BYTE PAR H and cache data bits 15:08 are tested in the other parity checker.

Even parity asserts K512 LOW BYTE PAR OK H and K513 HI BYTE PAR OK H, two of the signals necessary to generate a "hit" (Figure 2-7). Odd (bad) parity asserts K512 LO BYTE PAR ERR L and K513 HI BYTE PAR ERR L, unless inhibited by K514 FORCE MISS L, described previously (Paragraph 2.2.3.2). These two error signals are inputs to the jam register CACHE ERR (1) H bit (K705) and to BUS DIN MUX (STATUS) on K708.

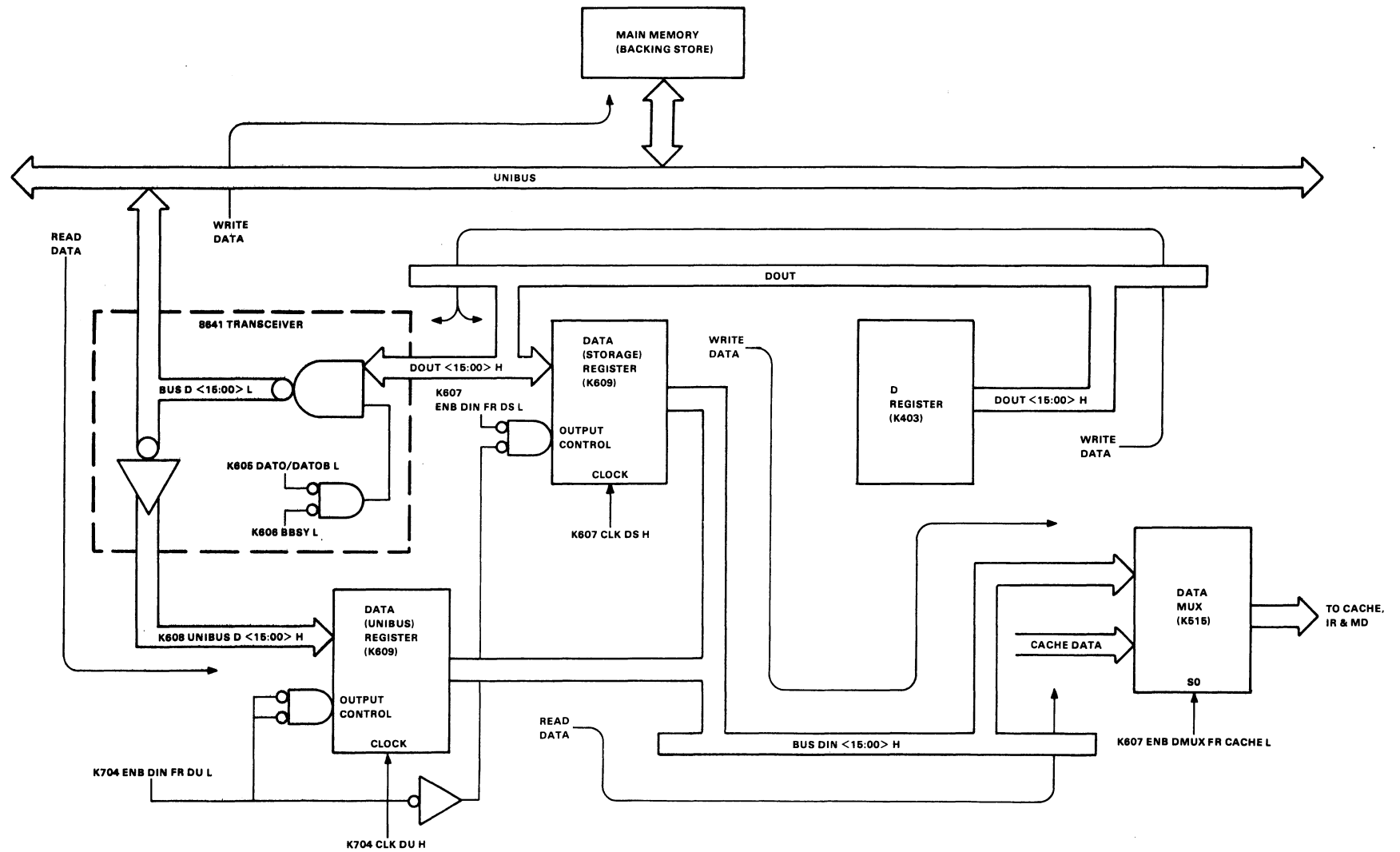
The other destination for the two 8-bit cache data bytes is the A port of the DMUX. The A port is selected by K607 ENB DMUX FR CACHE L, which is low for a DATI cache request, K607 CACHE RQST (DATI) (1) H is high (Figure 2-9). From the DMUX the data, now K515 DMUX (15:00) H, is routed to three possible places: the C scratchpad (CSP) (K407) (normally the MD register), the instruction register (IR) (K303) if the data is an instruction, or the floating point processor if a floating point instruction or data is involved. The MD register receives both data and some instruction.

2.2.3.6 Store Inputs – The inputs to the stores may be considered as either control information (address, valid, and parity bits) or data to be stored. The valid and tag address inputs to the tag parity generator, valid store, and cache tag store have already been discussed (Paragraphs 2.2.2 and 2.2.3.1). Addressing inputs to the stores were also described previously (Paragraph 2.2.3.3).

Data input to the high and low cache data stores is via the DMUX (Figures 2-7 and 2-9). Data read from main memory as the consequence of a DATI miss is routed to cache memory via the Unibus and DIN data lines (Figure 2-10). Identified on the Unibus as BUS D (15:00) L, the read data goes through the receiver portion of the bus transceivers to the Unibus data buffer as K608 UNIBUS D (15:00) H. From the data buffer the data, BUS DIN (15:00) H, is delivered to the DMUX. On the other hand, data written in memory as the result of a DATO (B) follows a slightly different path. The data, which is stored in both cache and main memory, is output from the D register to the DOUT lines as K403 DOUT (15:00) H. Here the path diverges with the data for main memory being gated through the transmitter portion of bus transceivers to the Unibus, while the data for cache is latched into the data storage (DS) register in bus control. From the DS register it goes, via the BUS DIN lines, to the DMUX. Both write and read data use the BUS DIN lines and are input to the same port in the DMUX. However, the two inputs to BUS DIN are never enabled at the same time. The Unibus data buffer output is enabled by K704 ENB DIN FR DU L, while the DS register output is enabled by K607 ENB DIN FR DS L.

Write data enters the DMUX from the DIN lines through port B, which is selected by K607 ENB DMUX FR CACHE L (Figure 2-8). From the DMUX the data, DMUX (15:00) H, is sent to the high and low cache data stores and to the data parity generator (Figure 2-7). Two chips are used for parity generation, one for the high byte and one for the low byte. The outputs, K510 HI BYTE PAR IN H and K510 LO BYTE PAR IN H, are asserted to maintain even parity, i.e., if the input data configuration is odd.

The two data parity bits are input to the addressed cache data parity store locations from where they are available as inputs (K510 CACHE LO BYTE PAR H and K510 CACHE HI BYTE PAR H) to the two data parity checkers (Figure 2-8). Parity checking of the data bytes and the tag is performed every time cache is accessed, i.e., for both read and write cache operations. Even parity conditions are necessary to assert the hit signal; odd parity will assert the parity error signals unless inhibited by K514 FORCE MISS L (Paragraph 2.2.3.2).



11-5562

Figure 2-10 Cache Data Interface

2.2.3.6.1 Cache Write Signals – The source and destination of the signals used to write data into the various cache stores and registers are listed in Table 2-1. Different configurations of the microcontrol (UCON) signals assert some of these signals; they are listed in Table 2-2. (The UCON interface is described in Paragraph 4.16).

2.2.3.6.2 Data Multiplexer (DMUX) – The DMUX, in addition to being a key element in the path for data read from cache and data written to cache (Figures 2-7 and 2-10), also provides the means to transfer secondary information to the data path and processor control modules (Figure 2-9). This information originates in several cache and memory management registers such as the hit register and memory management registers (MMRs) 0 and 2. Two multiplexers, read/write register and read only register, feed the data to the DMUX by way of the BUS DIN lines. Other information, in the form of internal addresses, is derived from two internal address ROMs (Paragraph 1.4). Selection and enable control is developed from UCON signals (Table 2-2).

2.2.3.7 Cache Control Register (CCR) – The CCR (Figure 2-8) is used to modify cache operation for diagnostic purposes. Input data bits 7:6 and 3:2 on the DOUT lines from the D register develop four control signals. The clock input to the CCR is derived and controlled by UCON bits (K514).

DOUT line bit 6 is used to write wrong parity. When set, it will cause incorrect (odd) parity to be generated in the tag and data fields. When read, those locations will cause the tag and data parity to indicate bad parity conditions (K513 HI BYTE PAR OK H, K512 LO BYTE PAR OK H, and K511 TAG PAR OK H), which will inhibit the hit signal. Bits 2 and 3 are used to force a miss in the lower half or upper half of cache. FORCE MISS 0 (1) H (bit 2) will cause misses when physical bus address bit 10 (PBA 10 H) is a zero (cache locations 0 – 1776₈). FORCE MISS 1 (1) H (bit 3) will cause misses if PBA 10 H is a one (cache locations 2000 – 3776₈). Bit 7, when a one, asserts K514 ENB CPE JAM (1) H which, when a cache parity error is sensed, causes a jam (K705) and aborts the cache cycle. This is followed by an immediate trap to location 114. However, when bit 7 is a zero, K514 CPE JAMUPP (1) H causes the processor to first go to main memory for the needed data when a cache parity error is sensed. Then, after the Unibus memory cycle is completed, a jam and trap to location 114 occurs.

2.2.4 Memory Management

An integral picture of the cache memory module is the KT11 memory management logic that provides for the relocation of virtual memory addresses to physical memory addresses, i.e., the transformation from a symbolic to an absolute addressing scheme. Because the KT11 automatically relocates all addresses when enabled, the PDP-11/60 can be considered to be operating in a virtual address space. This means that regardless of where a program is loaded into physical memory, it will not have to be "re-linked;" it always appears to be at the same virtual location in memory.

2.2.4.1 Memory Relocation – The primary KT11 function is to perform memory relocation and thus provide expanded memory addressing capability for systems with more than 28K of physical memory. The KT11-D uses two sets of page address registers (PARs) to relocate virtual addresses to physical addresses in memory. These sets are used as hardware relocation registers that permit several user programs, each starting at virtual address 0, to simultaneously reside in physical memory.

Table 2-1 Cache Write Signals

Signal Name	Destination	Function
K514 WRITE CCR LOW L	K514	CCR DATA 2,3,6,7, from DOUT (Assert JAMUPP, FORCE MISS, WT WRONG PAR bits)
K514 WRITE MMR0 LOW L	K505	MMR0, DATA 0 from DOUT [assert K505 RELOCATE ENB (1) H]
K514 WRITE MMR0 HIGH L	K505	MMR0, DATA 8,13,14,15 from DOUT or RRO, page length, NR, illegal errors
K506 WRITE PAR+PDR L	K504	PDR SEL PAR+PDR H (clears W bit) PDR SEL PAR+PDR H (sets W bit)
K514 WRITE PDR LOW L	K504	PDR DATA (3:1) (expansion direction and access control field bits)
K514 WRITE PDR HIGH L	K504	PDR DATA (14:08) from DOUT (page length field bits)
K514 WRITE PAR LOW L	K503	PAR DATA (7:0) from DOUT
K514 WRITE PAR HIGH L	K503	PAR DATA (11:8) from DOUT
K514 WRITE SLR HIGH L	K506	Stack limit register DOUT (15:8) (lower limit of stack)
K514 WT CACHE TAG L	K510 K510 K511	Valid Store (VALID BIT) Tag parity store (TAG PARITY BIT) Cache tag store PA MUX (17:11) (tag address bits)
K514 WT CACHE DATA LO L	K510 K512	Data parity LO store (Low Byte Parity bit) Cache data store (LO) DMUX (7:0) (low data byte)
K514 WT CACHE DATA HI L	K510 K512	Data parity HI store (High Byte Parity bit) Cache data store (HI) DMUX (15:8) (high data byte)

Table 2-2 UCON KT/Cache Functions

K209 UCON KT (1) H	K607 INH UCON (BUS XFER) L	K209 UCON XFER (1) H	UCON										Resultant Signal/Function
			14	13	12	11	10	9	8	7	6	5	
1			0										Assert RELOCATE L
1			1										Inhibit RELOCATE L
1	0			0	1								DIN ← UNIBUS ADRES Internal to CP*
1	0			1	1								DIN ← CP Internal Adrs
1	0					1		1					DIN ← TAG, HIT REG
1	0					1		0					DIN ← MMR2
1	0						1				0	0	DIN ← CCR, SLR
1	0						1				0	1	DIN ← MMR0
1	0						1				1	0	DIN ← PDR
1	0						1				1	1	DIN ← PAR
									0	0			(Inhibit Writing)
1		1							0	1	0	0	Assert WRITE CCR LOW L
1		1							0	1	0	1	Assert WRITE MMR0 LOW L
1		1							0	1	1	0	Assert WRITE PDR LOW L
1		1							0	1	1	1	Assert WRITE PAR LOW L
1		1							1	0	0	0	Assert WRITE SLR HIGH L
1		1							1	0	0	1	Assert WRITE MMR0 HIGH L
1		1							1	0	1	0	Assert WRITE PDR HIGH L
1		1							1	0	1	1	Assert WRITE PAR HIGH L
1		1							1	1	X	X	(Write high and low bytes)
1											1		Assert SEL PAR+PDR H

*NOTE: Refer to Paragraph 1.4.

2.2.4.2 Program Relocation – The PARs are used to determine the starting address of each relocated program in physical memory. Figure 2-11 shows a simplified example of the relocation concept. A more detailed example of how memory relocation works is shown in Figure 2-13. In Figure 2-11, Program A starting address 0 is relocated by a constant to provide physical address 6400₈.

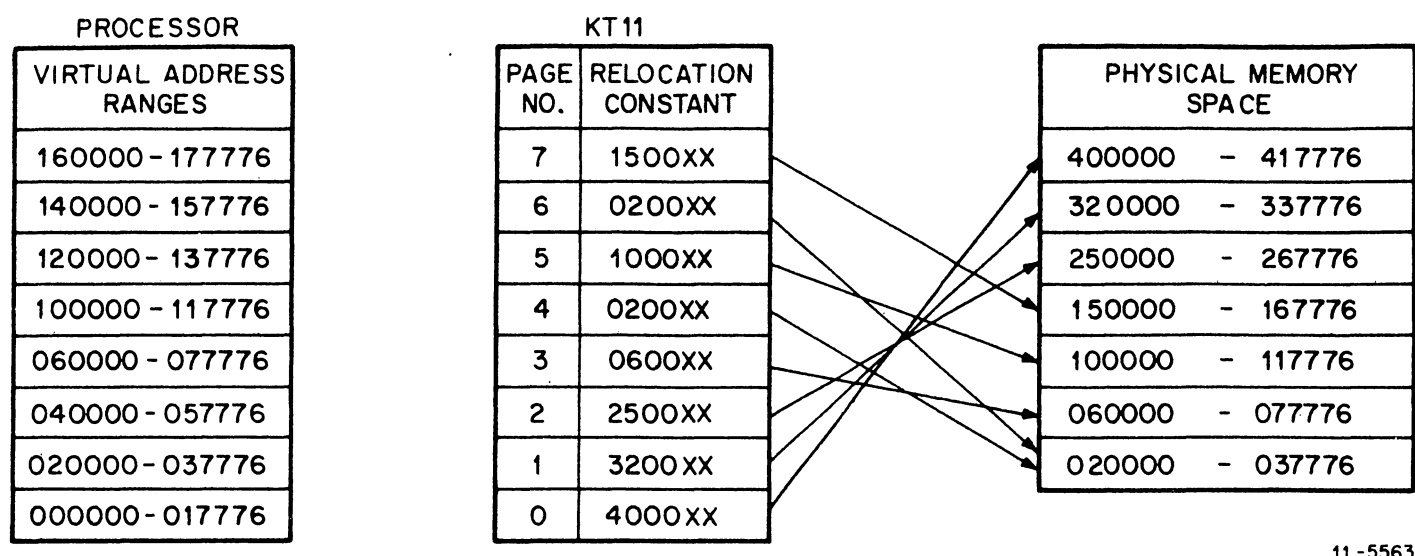


Figure 2-11 Simplified Memory Relocation Example

If the next processor virtual address is 2, the relocation constant will then cause physical address 6402₈, which is the second item of Program A, to be accessed. When Program B is running, the relocation constant is changed to 1000₈. Then, Program B virtual addresses starting at 0 are relocated to access physical addresses starting at 100000₈. Using the active PARs to provide relocation eliminates the need to “re-link” a program each time it is loaded into a different physical memory location. The program always appears to start at the same address.

In the PDP-11/60 systems, a program is relocated in pages. A page can consist of from 1 to 128 blocks. Each block is 32 words in length. Thus, the maximum length of a page is 4096 (128 × 32) words. Using all of the eight available active page registers in a set, a maximum program length of 32,768 words can be accommodated. Each of the eight pages can be relocated anywhere in the physical memory, with each relocated page beginning on a boundary that is a multiple of 32 words. Only the memory actually allocated to the page can be accessed.

The relocation example shown in Figure 2-12 illustrates several points about memory relocation. These points are:

1. Although the program appears to be in contiguous address space to the processor, the 32K-word virtual address space is actually scattered through several separate areas of physical memory. As long as the total available physical memory space is adequate, a program can be loaded. The physical memory space need not be contiguous.
2. Pages may be relocated to higher or lower physical addresses, with respect to their virtual address ranges. In the example of Figure 2-11, page 1 is relocated to a higher range of physical addresses, page 4 is relocated to a lower range, and page 3 is not relocated at all (even though its relocation constant is nonzero).
3. All of the pages shown in the example start on 32-word boundaries.

4. Each page is relocated independently. There is no reason why two or more pages could not be relocated to the same physical memory space. Using more than one PAR in the set to access the same space would be one way of providing different memory access rights to the same data, depending on which part of a program was referencing that data. In the example shown in Figure 2-12, note the relocation constant is assigned to pages 4 and 6. As a result, virtual addresses within both address ranges access the same physical addresses in memory, using separate PARs.

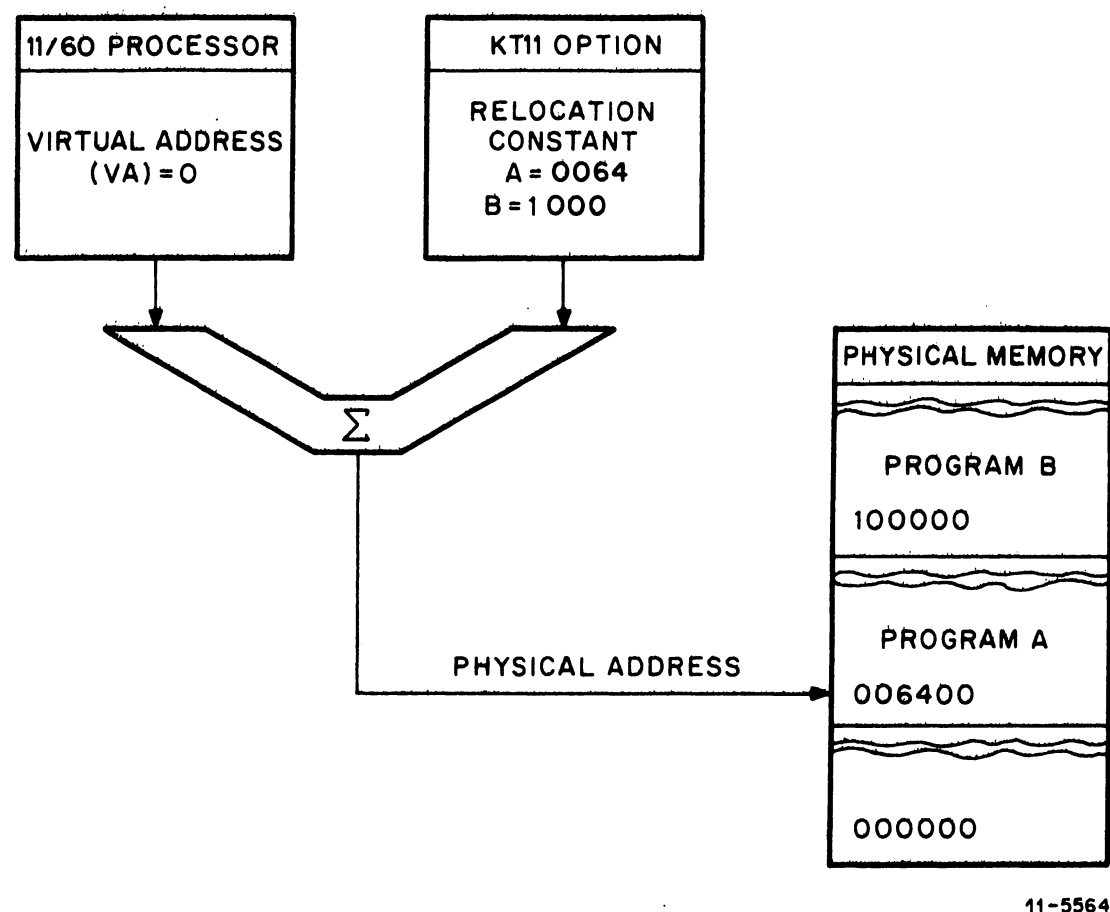


Figure 2-12 Relocation of a 32K Word Program
Into 124K Word Physical Memory

2.2.4.3 Virtual to Physical Address Conversion - With the KT11 memory management logic as a standard component of the PDP-11/60 cache module, the address output from the data path module cannot be considered as the direct physical address of a memory location. Instead, it is viewed as an 18-bit virtual address that contains information to be used by the KT11 to construct a 17-bit physical address. (Bit 0 is not used in the physical address configuration to cache.)

Figure 2-13 shows how a physical address is constructed. Virtual address bits BA (15:13) from the AMUX are interpreted as the page field to select one of eight PARs. There are two sets of eight PARs, one set for use in the kernel mode and one set for use in the user mode. The state of K505 KERNEL MODE H selects one of the two sets. Virtual address bits BA (12:06) provide the block number (0 to 127₁₀) within the page (K504), while BA (05:01) define the displacement within each 32-word block.

S4A

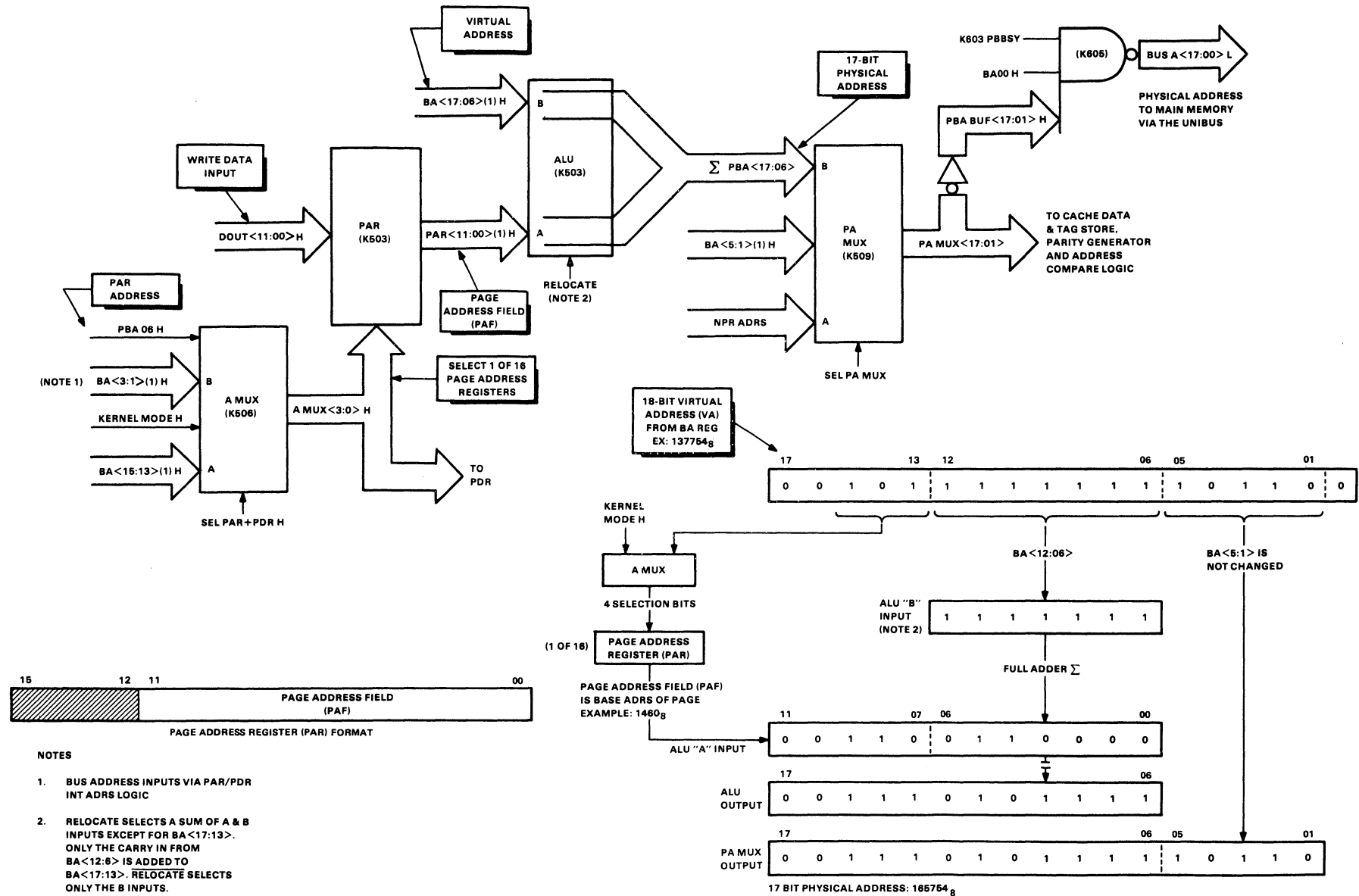


Figure 2-13 Cache Memory Relocation (PAR)

Each PAR contains a page address field (PAF) that is stored in the PAR under program control at the time the complete PAR/PDR (page descriptor register) set is defined for a program page. The PAF can be considered the base address of the page (Figure 2-14). The block number, BA (12:06), is added in the ALU to the base address, PAR (11:00), to form the twelve most significant bits of the physical address. This, plus virtual address bits, BA (05:00), which are unchanged by relocation, make up the 18-bit physical address.

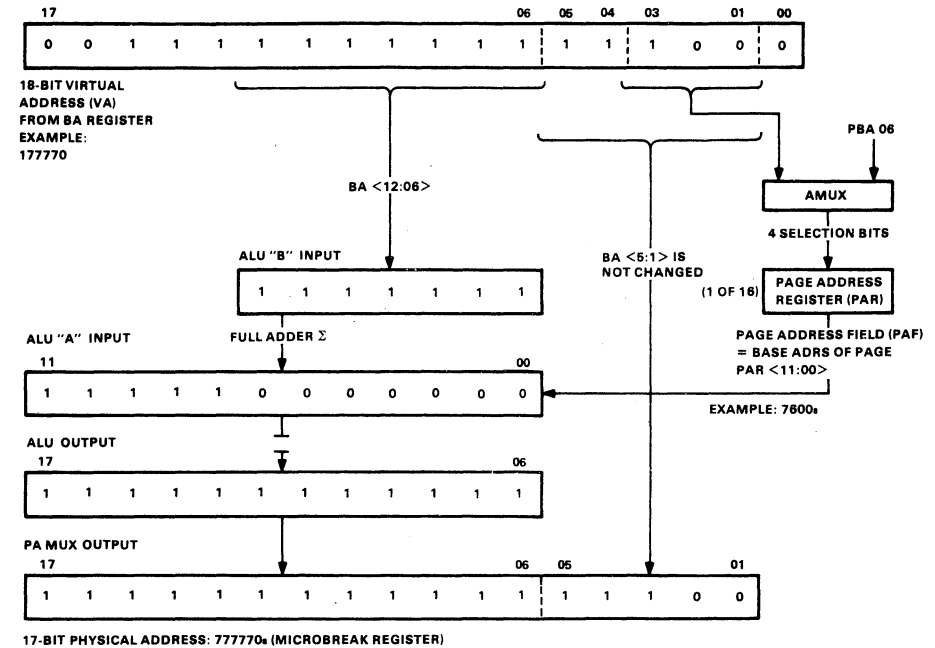


Figure 2-14 Internal Address Relocation

To relocate a virtual address to a physical internal address (760000–77776₈) virtual address bits BA (3:1) and K503 PBA 06 H from the AMUX (K506) select one of 16 PARs to provide the PAF (Figure 2-14). The PAF [PAR (11:00)] is summed with the block number [BA (17:06)] in the ALU. The result, along with the six LSBs that are unchanged, constitute the physical internal address.

The key to which bits are used to pick one of the 16 PARs is K506 SEL PAR + PDR H (Figure 2-13), the AMUX select signal. This signal, when low, selects the "A" inputs to the AMUX, BA (15:13) and KERNEL MODE H. The "B" inputs, BA (3:1) and PBA 06 H, are selected when SEL PAR + PDR H is high.

The cache ALUs (Figures 2-7 and 2-13) consist of three 74S181 ALU chips connected by two 74S182 look-ahead carry generators (K503). K505 RELOCATE L, the function select input to the ALU, is asserted by bit 00=1 and bit 8=1 (for maintenance purposes) on the DOUT lines. If RELOCATE is asserted, the ALU output is the sum of the A and B inputs, except for the high-order bits (17:13). These bits are always equal to the A inputs [PAR (17:13)] plus the carry in from the addition of the lower bits. When RELOCATE is negated, the ALU output is equal to the B input [BA (17:06) (1) H]. Consequently, RELOCATE causes the physical address to be equal to the virtual address; all address relocation is inhibited.

2.2.4.4 PAR/PDR Registers – The KT11-D Memory Management Unit provides two sets of eight PAR/PDR pairs. Each pair consists of a PAR and a PDR. These registers are always used as a pair and contain all information required to locate and describe the current active pages for each mode of operation. One PAR/PDR set is used in kernel mode and the other is used in user mode. The current mode bits (or in some cases, the previous mode bits) of the PSW determine which set will be referenced for each memory access. A program operating in one mode cannot use the PAR/PDR sets of the other mode to access memory. Thus, the two sets are a key feature in providing a full-protected environment for a timeshared multiprogramming system.

A specific processor I/O address is assigned to each PAR and PDR of each set. Table 2-3 is a complete list of address assignments.

NOTE

Unibus devices cannot access PARs and PDRs.

Table 2-3 PAR/PDR Address Assignments

Kernel Active Page Registers			User Active Page Registers		
No.	PAR	PDR	No.	PAR	PDR
0	772340	772300	0	777640	777600
1	772342	772302	1	777642	777602
2	772344	772304	2	777644	777604
3	772346	772306	3	777646	777606
4	772350	772310	4	777650	777610
5	772352	772312	5	777652	777612
6	772354	772314	6	777654	777614
7	772356	772316	7	777656	777616

In a fully protected multiprogramming environment, the implication is that only a program operating in the kernel mode would be allowed to write into the PAR and PDR locations to map user's programs. However, there are no restraints imposed by the KT11 logic that will prevent user mode programs from writing into these registers. The option of implementing such a feature in the operating system, and thus explicitly protecting these locations from user's programs, is available to the system software designer.

2.2.4.4.1 Page Address Registers (PARs) – The PAR, Figure 2-13, contains the 12-bit PAF that specifies the base address of the page.

Bits (15:12) of the PAR are not implemented in the hardware.

The PAR can be alternatively thought of as a relocation constant, or as a base register containing a base address. Either interpretation indicates the basic function of the PAR in the relocation scheme.

2.2.4.4.2 Page Descriptor Registers (PDR) – The PDR, shown in Figure 2-15, contains information relative to page expansion, page length, and access control.

Access Control Field (ACF) – This 2-bit field, ACF (02:01) of the PDR, describes the access rights to this particular page. The access codes or “keys” specify the manner in which a page may be accessed and whether or not a given access should result in an abort of the current operation. A memory reference that causes an abort is not completed and is terminated immediately.

Aborts (K505 MM ABORT L) are caused by attempts to access nonresident pages, page length errors, or access violations, such as attempting to write into a read-only page. Traps are used as an aid in gathering memory management information. Abort conditions are transmitted to the data path via MMR0 (Paragraph 2.2.4.5.1).

In the context of access control, the term “write” is used to indicate the action of any instruction that modifies the contents of any addressable word. A “write” is synonymous with what is usually called a “store” or “modify” in many computer systems. Table 2-4 lists the ACF keys and their functions. The ACF is written into the PDR under program control.

Expansion Direction (ED) – The ED bit located in PDR bit position 03 indicates the authorized direction in which the page can expand. A logic 0 in this bit (ED=0) indicates the page can expand upward from relative zero. A logic 1 in this bit (ED=1) indicates the page can expand downward toward relative zero. The ED bit is written into the PDR under program control. When the expansion direction is upward (ED=0), the page length is increased by adding blocks with higher relative addresses. Upward expansion is usually specified for program or data pages to add more program or table space. An example of page expansion upward is shown in Figure 2-16.

When the expansion direction is downward (ED=1), the page length is increased by adding blocks with lower relative addresses. Downward expansion is specified for stack pages so that more stack space can be added. An example of page expansion downward is shown in Figure 2-17.

Written Into (W) – The W bit located in PDR bit position 06 indicates whether the page has been written into since it was loaded into memory. W=1 is affirmative. The W bit is automatically cleared when the PAR or PDR is written into (UCON 6=1) (Table 2-2). K506 SEL PAR+PDR H is the signal that performs this function (Figure 2-15). K506 SEL PAR+PDR H, is also the select signal for the AMUX (Figure 2-13).

In disk swapping and memory overlay applications, the W bit (bit 6) can be used to determine which pages in memory have been modified by a user. Those that have been written into must be saved in their current form. Those that have not been written into (W=0) need not be saved and can be overlaid with new pages, if necessary.

NOTE

The W bit cannot be set by a memory access of a KT11 internal register (MMR0) or a memory access that causes an abort.

Page Length Field (PLF) – The 7-bit PLF located in PDR bits (14:08) specifies the authorized length of the pages, in 32-word blocks. The PLF holds block numbers from 0 to 177₈, thus allowing any page length from 1 to 128₁₀ blocks. The PLF is written in the PDR under program control.

56A

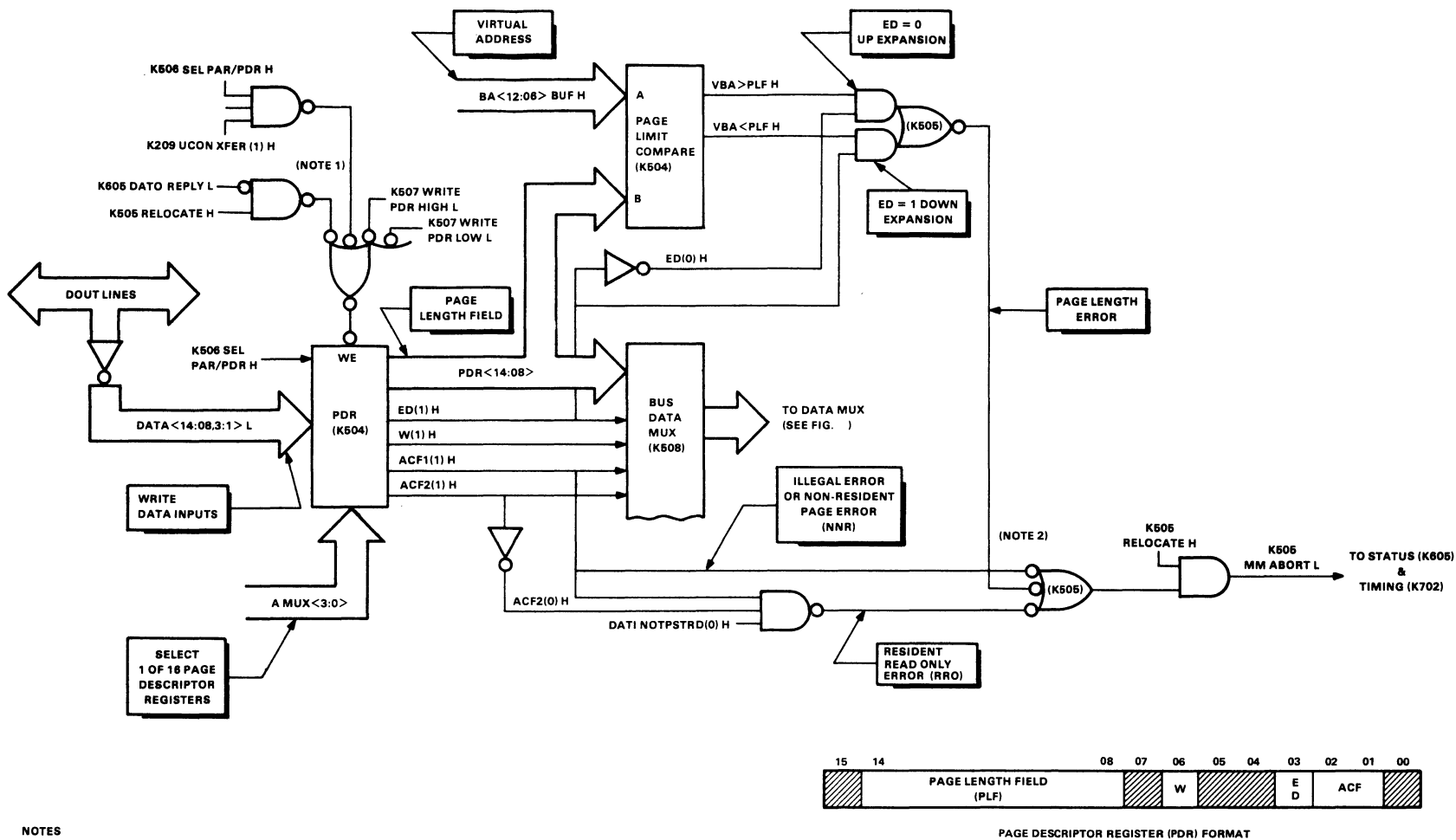


Figure 2-15 Page Status and Control (PDR)

Table 2-4 ACF Keys

ACF	Key	Description	Function
00	0	Nonresident (NR)	Abort any attempt to access this NR page.
01	2	Resident read-only (RRO)	Abort any attempt to write into this page.
10	4	Illegal	Abort all accesses.
11	6	Resident read/write (RRW)	Read or write allowed. No trap or abort occurs.

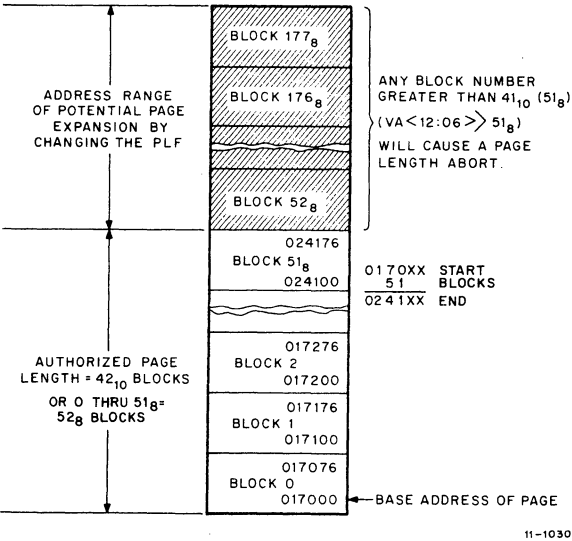
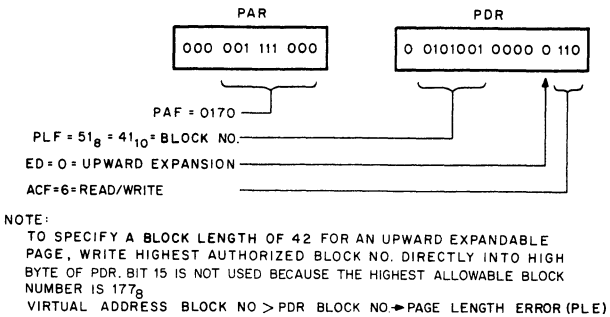
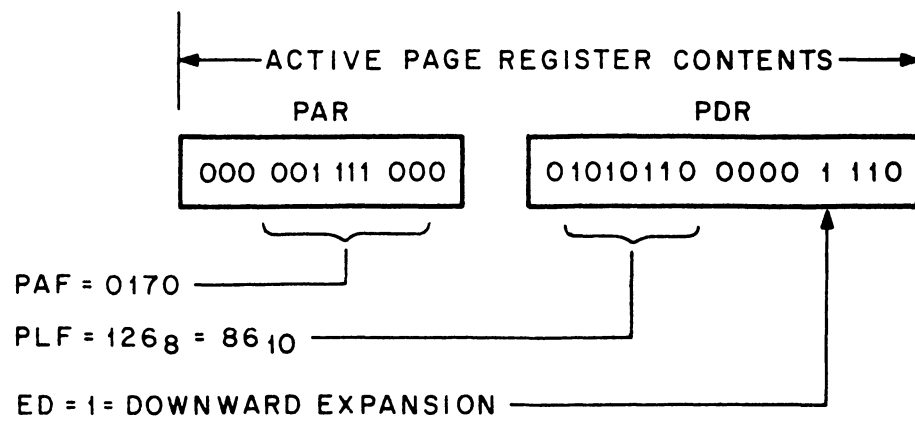


Figure 2-16 Example of an Upward Expandable Page



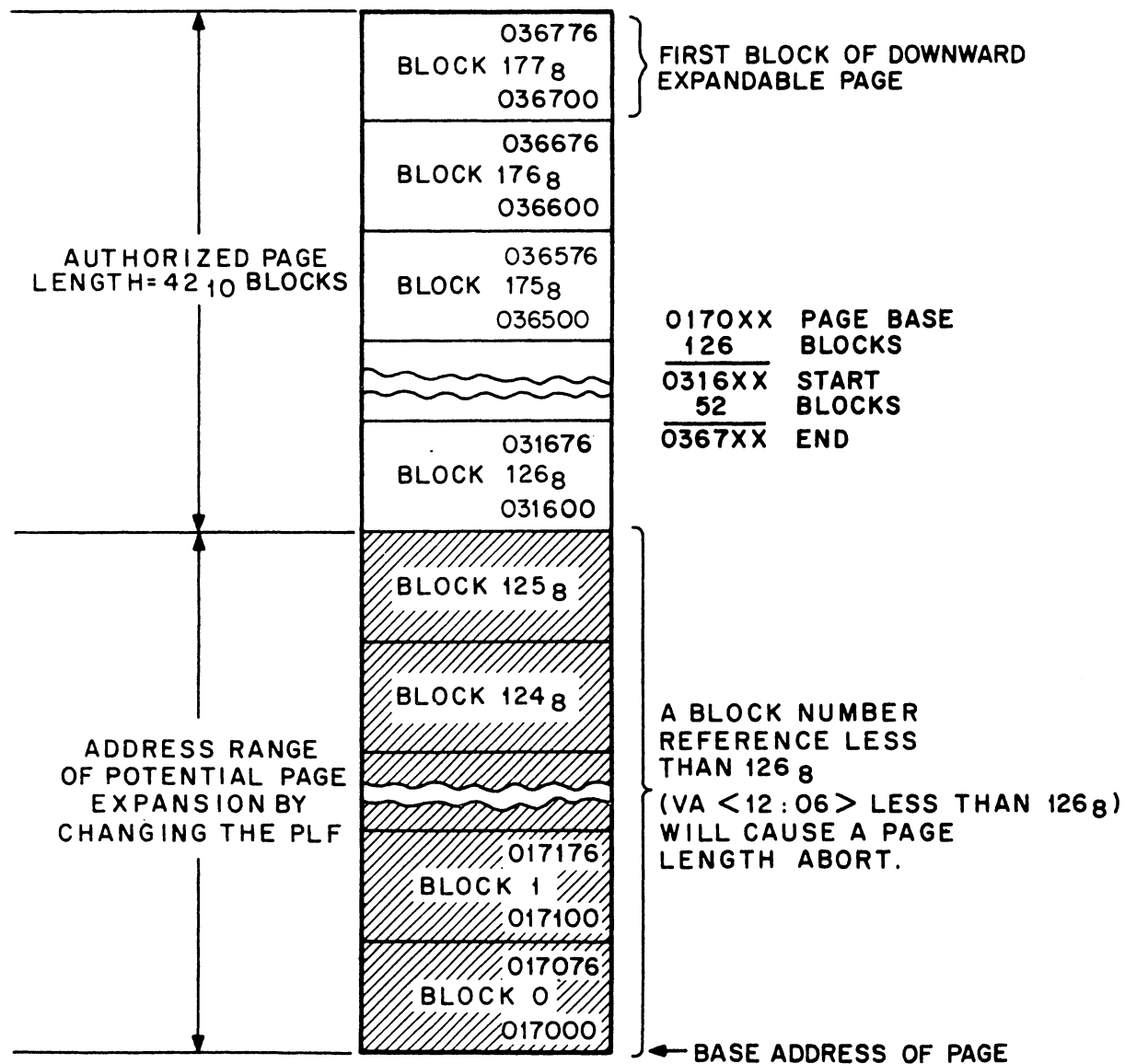
TO SPECIFY PAGE LENGTH FOR A DOWNWARD EXPANDABLE PAGE,
WRITE COMPLEMENT OF BLOCKS REQUIRED INTO HIGH BYTE OF PDR.

IN THIS EXAMPLE, A 42-BLOCK PAGE IS REQUIRED.

PLF IS DERIVED AS FOLLOWS:

42₁₀ = 52₈; TWO'S COMPLEMENT = 126₈

VIRTUAL ADDRESS BLOCK NO. <PDR BLOCK NO. → PAGE LENGTH ERROR (PLE)



11-1031

Figure 2-17 Example of a Downward Expandable Page

2.2.4.4.2.1 PLF for an Upward Expandable Page – When the page expands upward, the PLF must be set to one less than the intended number of blocks authorized for that page. For example, if 52_8 (42_{10}) blocks are authorized, the PLF is set to 51_8 (41_{10}) (Figure 2-16), block 0 being the page boundary and the first block of that page. The KT11 hardware compares the virtual address block number, VA (12:06), with the PLF to determine if the virtual address is within the authorized page length (Figure 2-15).

When the virtual address block number is less than or equal to the PLF, the virtual address is within the authorized page length. If the virtual address is greater than the PLF, a page length fault (address too high) is detected by the hardware and an abort occurs. In this case, the virtual address space legal to the program is noncontiguous because the three most significant bits of the virtual address are used to select the PAR/PDR set.

2.2.4.4.2.2 PLF for a Downward Expandable Page – The capability of providing downward expansion for a page is intended specifically for those pages that are to be used as stacks. In the PDP-11, a stack starts at the highest location reserved for it and expands downward toward the lowest address as items are added to the stack. The first block of the downward expandable page being block 177_8 .

When the page is to be downward expandable, the PLF must be set to authorize a page length, in blocks, that starts at the highest address of the page. That is always block 177_8 . Refer to Figure 2-17, which shows an example of a downward expandable page. A page length of 42_{10} blocks is arbitrarily chosen so that the example can be compared with the upward expandable example shown in Figure 2-16.

NOTE

The same PAF is used in both examples. This is done to emphasize that the PAF, as the base address, always determines the lowest address of the page, whether it is upward or downward expandable.

The rationale for complementing the number of blocks required to obtain the PLF is as follows:

Maximum Block No.	Minus	Required Length	Equals	PLF
177_8	–	52_8	=	125_8

2.2.4.5 Memory Management Register – Two memory management registers, MMR0 and MMR2, are used to convey cache status information, address bits, and error (abort) conditions to the CSP in the data path (Figure 2-18).

2.2.4.5.1 Memory Management Register 0 (MMR0) – Because different clock inputs are used, MMR0 can be considered as three registers. Abort conditions from the PDR logic (Paragraph 2.2.2.4.2) enter MMR0 through a multiplexer (on K505) whose other inputs are four bits from the DOUT lines. These inputs allow the abort conditions to be processor set for maintenance purposes. Also input to MMR0 are the three high-order virtual address bits, BA (15:13) (1) H, that select the page to be accessed. Once latched into MMR0, they are available to the processor to identify the page in the event of an abort or miss condition.

K307 USER MODE (1) H and K503 DATA 00 L are two more inputs to MMR0. The latter signal is the origin of K505 RELOCATE ENB (1) H which, when ANDed with K505 INH REL L, is one of the two ways to assert K505 RELOCATE L and RELOCATE H.

Outputs from MMR0 are sent to the read/write register multiplexer and then to the CSP via the DIN lines and the DMUX. Any one of the three error output signals will assert K505 MMR LOCK UP L. This signal locks up both the respective parts of MMR0, and also all of MMR2, by inhibiting two clock inputs. The effect is to lock up the identity of the error condition, the mode (user or kernel), and the page address in MMR0. In addition, the entire virtual address is locked up in MMR2.

K503 DATA 08 L from the DOUT lines, when equal to one, asserts K505 MAINT (1) H, which, when ANDed with K307 ENB MAINT (1) H, is the other way to assert K505 RELOCATE H. The MMR0 (1) L output for K505 MAINT (1) H is connected, in a lock-up loop, to the A port multiplexer input of DATA 08. Consequently, once MAINT (1) H is set, it will remain set until cleared by DATA 08 = 0. In a maintenance or diagnostic environment, MAINT (1) H is always output from MMR0 in parallel with any (or all) of the three error signals.

2.2.4.5.2 Memory Management Register 2 (MMR2) – MMR2 is a 16-bit register that is used to latch the virtual address, BA (15:00) (1) H, for diagnostic purposes. The output, MMR2 (15:00) (1) H, is transmitted to the CSP by way of the read only register multiplexer, the DIN lines, and the DMUX.

2.2.5 Stack Limit Register

The PDP-11/60 processor is capable of performing hardware stack operations. Because the number of locations occupied by a stack is unpredictable, some form of protection must be provided to prevent the stack from expanding into locations containing other information. Loading the stack limit register (SLR) with a number other than zero provides a programmable boundary. If the SLR is not loaded, then a default boundary of 400₈ (SLR=0) is used.

58A

The SLR (Figure 2-19) consists of a single addressable register, accessible to both the console and the processor, that is used to change the stack limit and to provide warning (yellow zone violation) and error (red zone violation) indications for the stack. The SLR is an 8-bit register (high-order byte) that can be addressed either as a high-order byte (777775) or as a full word (777774).

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

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

If the space between the bus address and the SLR is greater than 16 words, a red zone violation occurs and the operation causing the error is aborted. The stack is repositioned and a bus error trap occurs: i.e., the old PSW and program counter (PC) are pushed into locations 2 and 0 and the new PC and PSW are taken from locations 4 and 6. A red zone violation is a fatal stack error. Note that these two stack error conditions exist even when the SLR has not been loaded; however, in this case, the stack limit is fixed at memory location 400₈. Other fatal stack errors are odd stack or nonexistent stack.

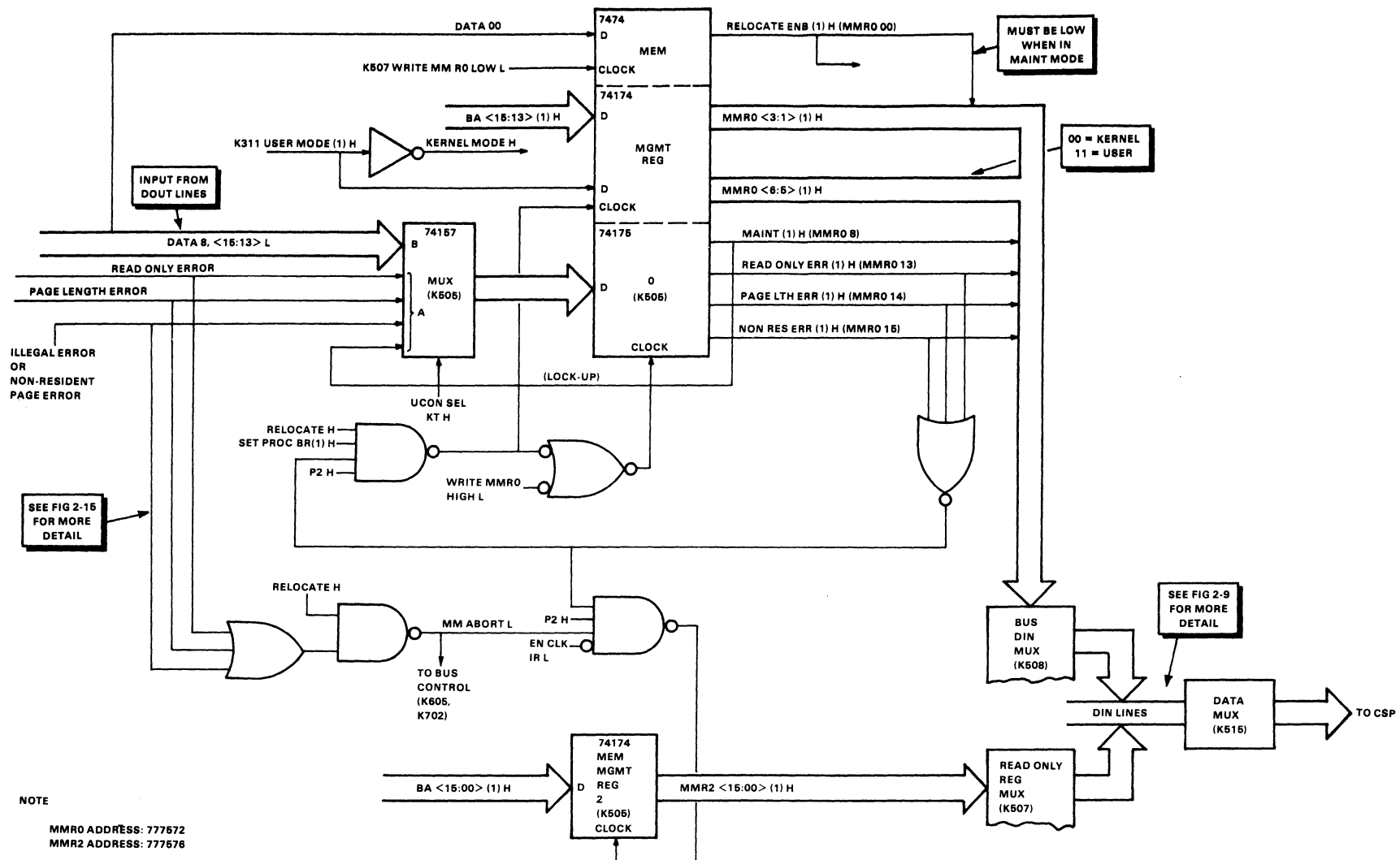
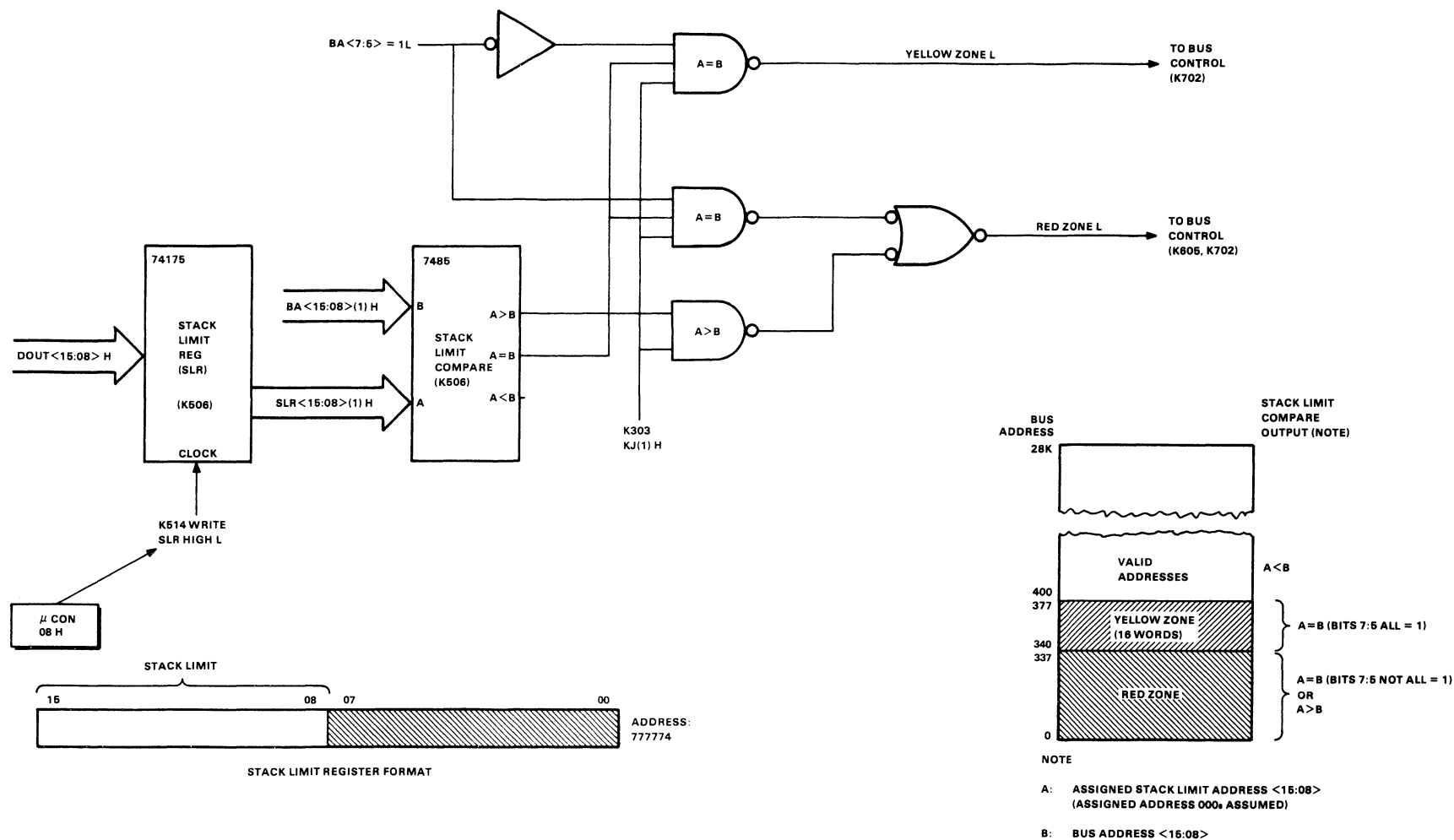


Figure 2-18 Memory Management Registers

SSC



11-5571

Figure 2-19 Stack Limit Register (SLR)

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

The comparison of the high-order byte of the address and the contents of the SLR is shown in Table 2-5. For the situation where the upper bytes of the SLR and the bus address are equal, it is necessary only to monitor the value of BA (07:05) to determine if a red or yellow zone violation has occurred. If all three of these bits are set, then the value of the low-order byte must be somewhere in the range of 340 to 377 (20 octal or 16 decimal word locations), which is a yellow zone violation. If any one of the bits is not set, then the highest possible address would be 337, which is the upper limit of the red zone.

Table 2-6 summarizes the method of monitoring the low-order byte to determine whether a red or yellow zone violation is present.

Table 2-5 Comparison of Address and SLR

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

584

Table 2-6 Detecting Type of Violation

Bus Address	High-Order Byte								Low-Order Byte							
	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
421								1	0	0	0	1	0	0	0	1
.												.				
.												.				
400								1	0	0	0	0	0	0	0	0
377								0	1	1	1	1	1	1	1	1
.												.				
.												.				
.												.				
340								0	1	1	1	0	0	0	0	0
337								0	1	1	0	1	1	1	1	1
.												.				
.												.				
.												.				
.												.				
000								0	0	0	0	0	0	0	0	0

More than SLR

Equal to SLR;
bits 7,6,5 are
all set

Equal to SLR;
bits 7,6,5 are
not all set

VALID

YELLOW

RED

- NOTES: 1. In above example, SLR is loaded with 000.
2. In *all* cases, highest yellow zone address must end in either 377 or 777.
3. In *all* cases, highest red zone address must end in either 337 or 737.

60

CHAPTER 3

DATA PATHS DETAILED LOGIC DESCRIPTION

3.1 SCOPE

This chapter describes the data path module and its functions in the PDP-11/60 processor. The data path module is made of logical elements that perform the data handling operations required by the processor. In the process, the data path logic alters, stores, and directs the data to specific points in the processor. Data path components, and the functions they perform, are controlled by the 48-bit microword from the ROM control store or the writable control store; different fields in these words control different parts of the data path.

The main elements in the data path module are along the data flow loops shown in Figure 3-1. Data storage is provided by the A, B, and C scratchpads, which are tied to the ALU by BUS AIN and BUS BIN. The ALU performs arithmetic and logical operations on the data; the result is stored in either the shift register (SR) or a holding register (the D register), applied to the shift tree, or routed directly back to the scratchpads.

3.2 THE ARITHMETIC AND LOGIC UNIT

A key circuit in the data path is the ALU. The ALU performs one of 16 operations on two, 16-bit words received from the A and B busses. Control for these operations is derived from the 4-bit ALU microword field BUS U(47:44) H. Table 3-1 lists the 16 ALU operations, input and output bit configurations, and the carry bit source. Figure 3-2 shows the ALU and related logic.

BUS U(47:44)H are input to the 4-bit UALU register located on the decode module. Outputs from this register provide the inputs to both ports of the ALU control multiplexer and the select signals for the carry-in multiplexer (CINMUX). The ALU mode select signal, K306 ALUM H, is also decoded from the ALU field of the microword. This signal, when low, selects the A port inputs to the ALU control multiplexer (K313) and an arithmetic function in the 74S181 ALU generators. When high, ALUM H selects the multiplexer B port and a logic function in the ALU (K404). Three bits of the ALU field, Bus U(47:45) H, are used to select which input to the CINMUX asserts K313 CIN 00 L, the carry-in bit to the ALU and the carry-out [D(C)] select register (Paragraphs 3.2.1 and 3.2.2). K313 CIN 00 L is also asserted if the UALU decodes a divide instruction and there is a carry from the last operation [D(C)B H].

The output of the ALU control multiplexer goes to a group of OR gates whose other inputs are asserted (low) if there is a divide and a carry (the same signal combination that asserts CIN 00 L). The four ALU generator chips use the OR gate outputs, K313 ALU SEL (3:0)H to select one of 16 functions to perform. Data input to the ALU is from the B or C scratchpads (BSP or CSP) via BUS BIN and the A scratchpad (ASP), the XMUX, or the shift tree via BUS AIN.

Outputs from the ALU are sent to several destinations. The primary 16-bit result of the operation on the A and B bus data [ALU(15:00) H] is clocked into, if selected, either the D register or the SR (Paragraphs 3.3 and 3.4). Three of these 16 bits, ALU 00 H, ALU 07 H, and ALU 15 H, provide three sources for the carry-out bit from the D(C) MUX. The two remaining ALU outputs, COUT 07 L and COUT 15 L, are the ALU word and byte carry bits. They are two more sources for the carry-out bit.

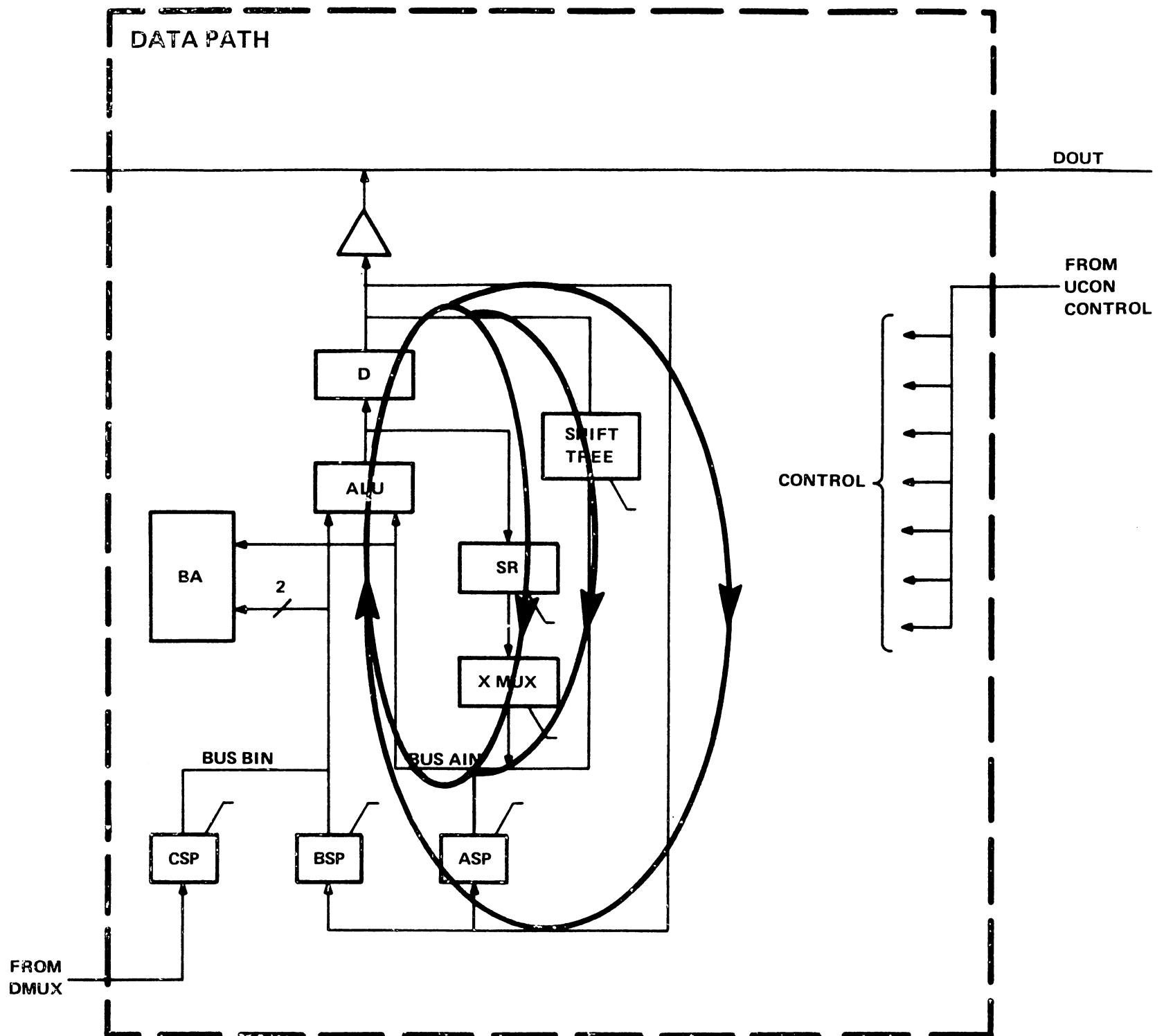


Figure 3-1 Data Flow Loops

Table 3-1 ALU Control Multiplexer

ALU OPERATION	ALU Field (Note 3) BUS U(47:44)	ALUM SEL H (Note 1)	ALU SEL (03:00) H	CIN 00 L Equal to:
A	0	H	0	1
ADD A+B+PS(C)	1	L	11	PS(C)
A * B	2	H	2	PS(C)
GENERATE 0	3	H	3	PS(C)
ADD A+B+D(C)	4	L	11	D(C)B
SUB W/CARRY (A+B+D(C))	5	L	6	D(C)B
XOR A WITH B	6	H	6	D(C)B
A * B	7	H	7	D(C)B
DIVIDE	10	L	11 (Note 2)	0
ADD A+B	11	L	11	0
SELECT B	12	H	12	0
A * B	13	H	13	0
ADD A+B+1	14	L	11	1
SUB B FROM A (A-B)	15	L	6	1
A OR B	16	H	16	1
SELECT A	17	H	17	1

- Notes:
1. H = Logic Constant, L = Arithmetic Constant.
 2. If divide and D(C) carry. [Integer division forces a subtract if D(C) present. This allows a one word inner divide loop.] Assert CINMUX L.
 3. The ALU field bits do not correspond one-to-one to the ALU select bits in the vendor's catalog but the ALUM SEL and ALU SEL (3:0) bits do match the vendor's catalog.

Table 3-1 ALU Control Multiplexer

ALU OPERATION	ALU Field (Note 3) BUS U(47:44)	ALUM SEL H (Note 1)	ALU SEL (03:00) H	CIN 00 L Equal to:
A	0	H	0	1
ADD A+B+PS(C)	1	L	11	PS(C)
A * B	2	H	2	PS(C)
GENERATE 0	3	H	3	PS(C)
ADD A+B+D(C)	4	L	11	D(C)B
SUB W/CARRY (A+B+D(C))	5	L	6	D(C)B
XOR A WITH B	6	H	6	D(C)B
A * B	7	H	7	D(C)B
DIVIDE	10	L	11 (Note 2)	0
ADD A+B	11	L	11	0
SELECT B	12	H	12	0
A * B	13	H	13	0
ADD A+B+1	14	L	11	1
SUB B FROM A (A-B)	15	L	6	1
A OR B	16	H	16	1
SELECT A	17	H	17	1

- Notes:
1. H = Logic Constant, L = Arithmetic Constant.
 2. If divide and D(C) carry. [Integer division forces a subtract if D(C) present. This allows a one word inner divide loop.] Assert CINMUX L.
 3. The ALU field bits do not correspond one-to-one to the ALU select bits in the vendor's catalog but the ALUM SEL and ALU SEL (3:0) bits do match the vendor's catalog.

3.2.1 The Carry-In Multiplexer

The CINMUX, located on the decode module, generates the carry-in bit for ALU operations. The logic consists of a 74S151 data selector/multiplexer and an associated OR gate. Inputs are: K404 D(C)B H (the carry-out bit from the previous ALU operation); K306 PS(C) (1) H (the PSW C bit); two constants, 1 and 0; and BUS U44H, which is ORed with PS(C) (1) H.

Multiplexer input selection is affected by the states of BUS U(47:45) H in the UALU register (Table 3-2). Note that the input selection pins do not line up with the three pertinent bits of the ALU field.

The output of the 74S151 is the complement of the input. It is ORed with the output of a NAND gate, which is qualified by a divide instruction [BUS U(47:44) = 10] and a carry-out [D(C)B H] from the D(C) MUX. The output of the OR gate is K313 CIN 00 L, the carry-in bit to the ALU, and the carry-out select register (COUT SEL).

Table 3-2 CINMUX Input Selection

UBUS			Selected Input
47	46	45	
MUX SELECT			
(S1)	(S0)	(S2)	
0	0	0	Bus U44 H or PS(C) (1) H
0	0	1	PS(C) (1) H
0	1	0	D(C)B H
0	1	1	D(C)B H
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

3.2.2 The D(C) Multiplexer

The D(C) multiplexer produces the carry-out bit from several sources (Figure 3-2 and Table 3-3). Most of these sources are the result of an ALU operation, e.g., the byte and word sign bits and the byte and word carry (overflow) bits. Two registers, D(C) and COUT SEL, are associated with the D(C) MUX. All the inputs to the D(C) MUX except one, PS(C) (1) H, the PSW C bit, are temporarily held in these registers. Both registers are clocked at the same time as the D register by K403 CLK D H (Paragraph 3.3). The three COUT field bits in the microword [BUS U(32:30)], are used to select the D(C) MUX input. These three bits are also stored in the COUT HOLD and COUT SEL registers.

Table 3-3 COUT Field Definition of Data Carry [D(C)]

COUT Field BUS U(32:30)	D(C) Source	Description
0	CIN 00	Carry-in bit from the CINMUX
1	PS(C)	Carry (C) bit of the PSW
2	ALU00	Odd or even result of the ALU operation
3	ALU07	Byte sign bit. Result of the ALU operation
4	ALU15	Word sign bit. Result of the ALU operation
5	COUT 07	Byte carry from the ALU operation
6	COUT 15	Word carry from the ALU operation
7	D(C)	Carry-out from previous operation

The D(C) MUX outputs consist of different forms of the carry-out bit, D(C); they are routed to several places. As K404 D(C) H, the carry-out bit is looped back to the D(C) register, where it is retained for a future operation. K404 D(C)B H is input to the low-order position of the SR and K404 D(C)A H goes to the high-order position of the XMUX for floating point operations. K404 D(C)B H is looped back to the CINMUX where it can be used in a subsequent ALU operation. Another destination for D(C)B H is a ROM in the condition code logic (K303). The two outputs from this ROM are K303 CC (C) H, the source of the PSW C bit, and K303 CC (V) H, the source of the PSW overflow bit (Paragraphs 4.6.3.3 and 4.8).

3.3 THE D REGISTER

The 16-bit data or D register (Figure 3-2) holds the result of the ALU operation for transmission to the Unibus (via DOUT), the A and B scratchpads, or the shift tree. The register consists of four, 4-bit 74S175 D-type flip-flop chips, one for each of the ALU chips. The D inputs receive K404 ALU(15:00) H, the ALU output, which is clocked in on the trailing (rising) edge of K403 CLK D H.

K403 CLK D H can be asserted at either P2 or P3 time, with the latter being the usual assertion time. Any carry or sign bit from the ALU operation (or carry-in from the CINMUX or carry-out from the COUT MUX) is clocked into the D(C) and COUT SEL registers at the same time by CLKD. Two 1-bit fields control when and if K403 CLK D H clocks the three registers. The CLK D field, microword bit 28, determines whether or not K403 CLK D H is asserted while the ENB P3 CLK field, microword bit 29, specifies the time (P2 or P3) the clock signal occurs.

Once the result of the ALU operation is in the D register, it is available to a number of places in the system. As K403 DOUT (15:00) H, the data can be directed to cache memory via the DOUT lines and to the Unibus and main memory, also via DOUT. The D register data input to the A and B scratchpads is termed K403 D(15:00) (0) H, while its high version, K403 D(15:00) (1) H, goes to the shift tree. (6)

3.4 THE SHIFT REGISTER (SR)

The SR (Figure 3-3) functions not only as a conventional shift register but also as a holding register for data from the ALU. In this capacity, the SR is used to hold data, such as addresses destined for BUS AIN that will not be written back to a scratchpad register or, because of insufficient time, data that is not written back during this timing cycle. The SR can also be used to form and hold a mask constant. The mask would then be loaded, in parallel with BUS BIN data, into the ALU during a subsequent operation. As a shift register, the SR can shift data one bit position to the left or one bit position to the right, once per microcycle.

The SR is composed of four, 4-bit 74S194 shift register chips. Mode control bits, the clear bit, and the clock bit go to each chip. Each chip also receives four data bits from the ALU. K410 BMUX 00 H, the LSB from the BMUX (Paragraph 3.5.2), provides the data input to the SR most-significant position during a shift right operation. The data source for the SR least-significant position in a shift left is either the carry-out bit, K404 D(C)B H, or K408 GUARD 03 (1) H. The state of the Guard Enable flip-flop determines which input is selected.

Associated with the SR is another 74S194 chip, the SR guard. The SR guard receives, i.e., saves, the SR LSB during a right shift. The 2-bit output, K408 GUARD (3:2) (1) H, is used only in the UBR-ANCH MUX (Paragraph 4.5) for a 2-bit BUT; it is not readable via BUS AIN.

Selection of the SR mode is determined by two inputs from the residual control (RES CNTL) register. A LH combination of these signals, derived from BUS B13 H and BUS B12 H, at the trailing edge of K408 CLK SHIFT H, causes the SR to shift one position to the right. A shift left (one position) results from an HL select signal combination (BUS B13 H and BUS B12 H) when the clock signal is asserted. An HH select signal combination (BUS B13 H and BUS B12 H) causes a parallel load of 16 bits from the ALU at clock time. All SR functions are inhibited if both select signals are low (BUS B13 H and BUS B12 H). Table 3-4 summarizes these functions.

Table 3-4 Shift Register (74S194)

SRS1 (0) H	SRS0 (0) H	Shift Left Input (pin 7)	Shift Right Input (pin 2)	SR Clock Signal Ubit 27 = 1 (See Note)
L	L	X	X	Ignore clock (no function performed)
L	H	X	H/L	Shift right (bit 15 from BMUX 00 H)
H	L	H/L	X	Shift left (bit 00 from D(C) B H or GUARD 03 (1) H)
H	H	X	X	Parallel load from ALU

Note: Clock on the rising edge of CLK SHIFT H at P2 if bit 29 (ENB P3 CLK) = 0 or at P3 if bit 29 (ENB P3 CLK) = 1. Data is present at the outputs after the positive transition.

649

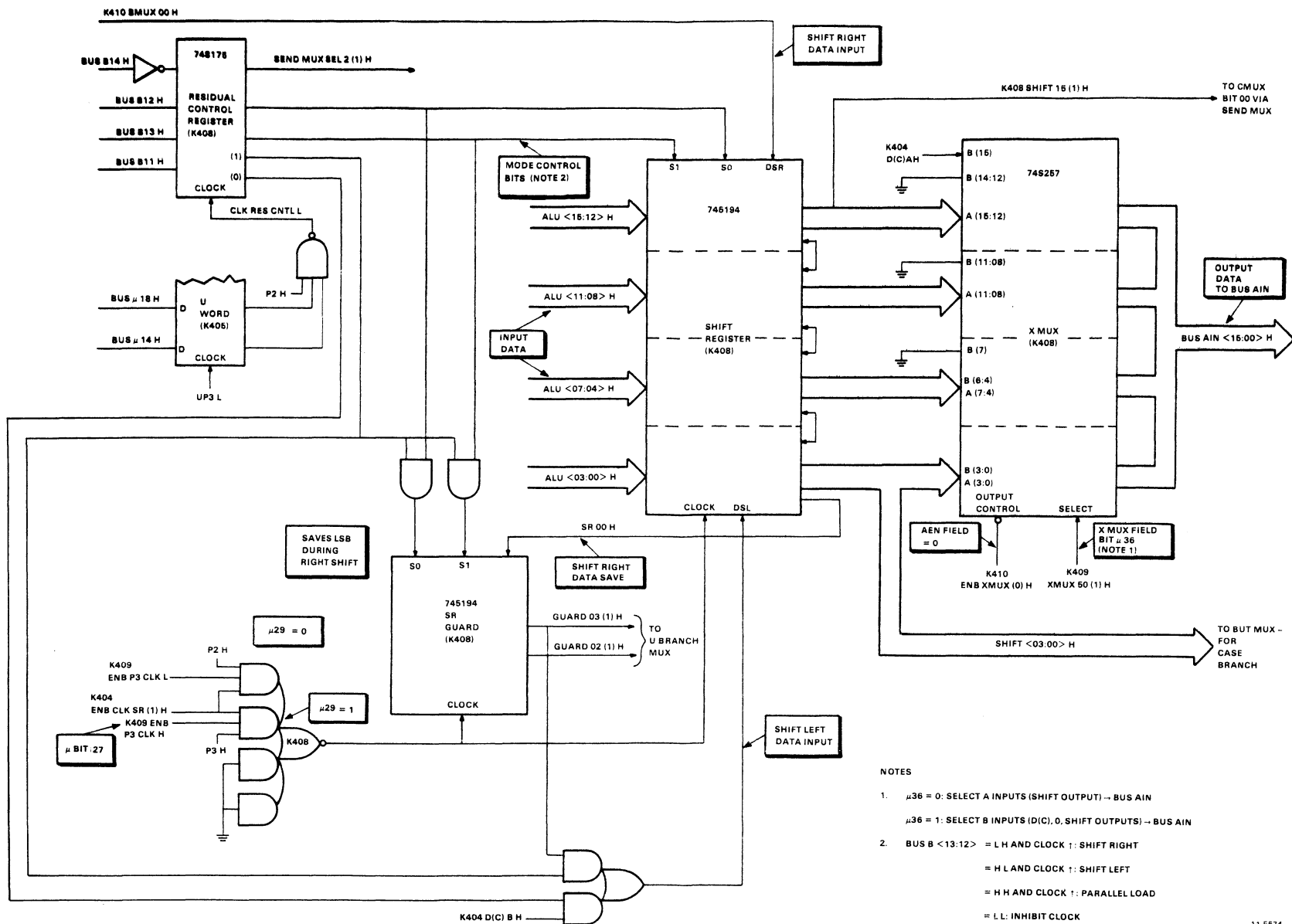


Figure 3-3 Shift Register (SR)
and XMUX

Two 1-bit microword fields control the SR clock signal (Figure 3-3). The CLKSR field, microword bit 27, determines whether or not the SR is clocked while the ENB P3 CLK field, microword bit U29, determines the time the SR is clocked; signal origins are shown on Figure 3-2. If bit 29 = 0, the SR is clocked on the trailing edge of P2 H. However, if microword bit 29 = 1, then the SR is clocked later, on the trailing edge of P3 H. If both the D register and the SR are clocked in the same microinstruction, they are clocked at the same time.

All 16 bits of the SR output go to the XMUX (Paragraph 3.4.2), while four bits, K408 SHIFT (03:00) (1) H, are delivered to the BUT MUX (Paragraph 4.5) for use in a 16-way Case branch.

3.4.1 Residual Control Register

The RES CNTL register, which is shown on Figure 3-3, serves as an extension of the microword. Control signals are developed in the RES CNTL from data on BUS BIN; the data is clocked in when Ubit 14 = 1 and Ubit 18 = 1 at P2 time. Two output signals provide mode control for the SR (Paragraph 3.4). Two other outputs control the shift left data input (bit 0) to the SR and the clock and function select signals for the SR guard. Another input, BUS B14 H, asserts K408 SEND MUX SEL 2 (1) H, one of the select signals for the shift end multiplexer in the shift tree (Paragraph 3.5). The RES CNTL register is cleared at the end of an instruction by K207 BUT (CLR FLAGS) H. As a result of this, the SR is returned to the parallel load mode.

3.4.2 The XMUX

The XMUX consists of four, 2-to-1, 74S257 data selector/multiplexer chips that transfer either SR or floating point results to BUS AIN (Figure 3-3). Control is provided by the microword AEN field [microword bits (39:38) = 0] and the XMUX field, microword bit 36. When microword bit 36 = 0 [K409 XMUX S0 (1) H], the 16-bit SR output is selected at the A port. If bit 36 = 1, K409 XMUX SEL 0 (1) H selects the B port. Used for floating point operations, the B port inputs contain a combination of D(C), 0, and SR bits:

Bit position	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	D(C)		0	0	0	0	0	0	0	0						SHIFT (06:00)

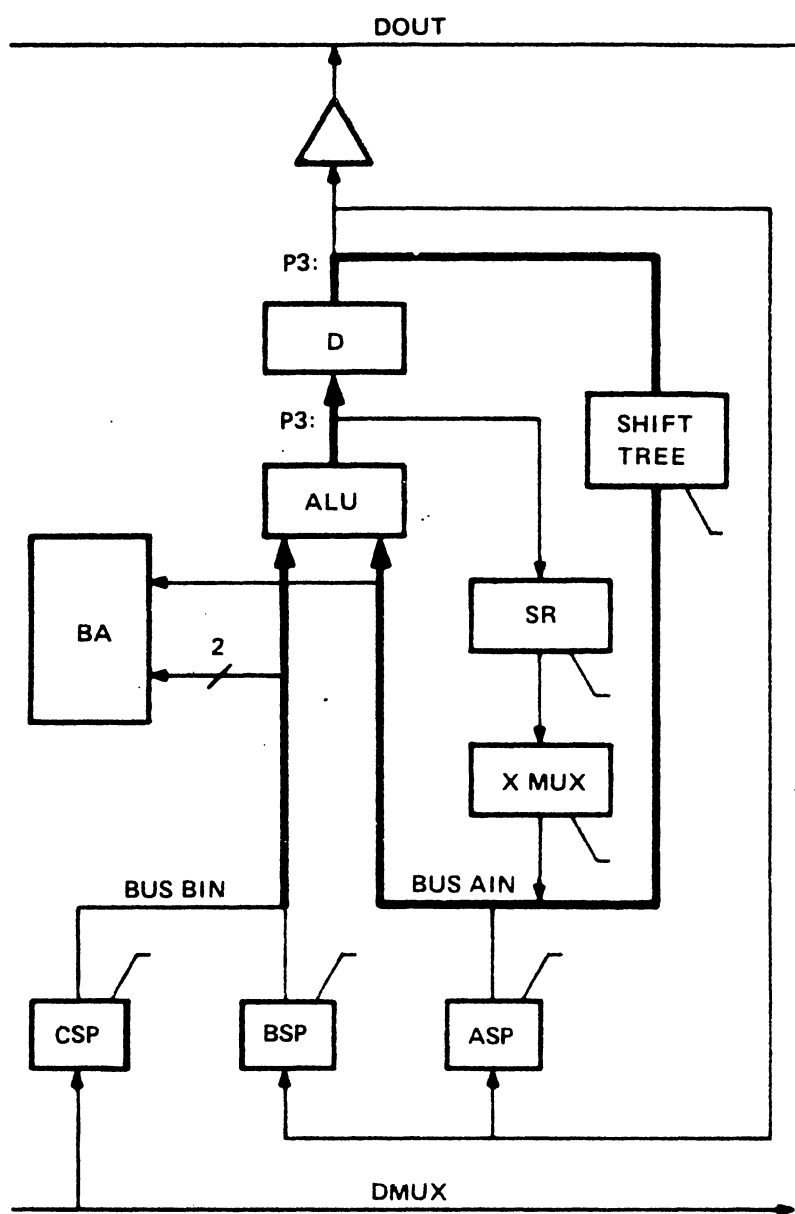
K405 ENB XMUX (0) H, derived from microword bit (39:38) = 0, is the output control signal for the XMUX; it must be low to enable the tristate output to BUS AIN. (A high on this line results in a high impedance output.) Table 3-5 lists the XMUX control and selection combinations.

Table 3-5 XMUX (74S257)

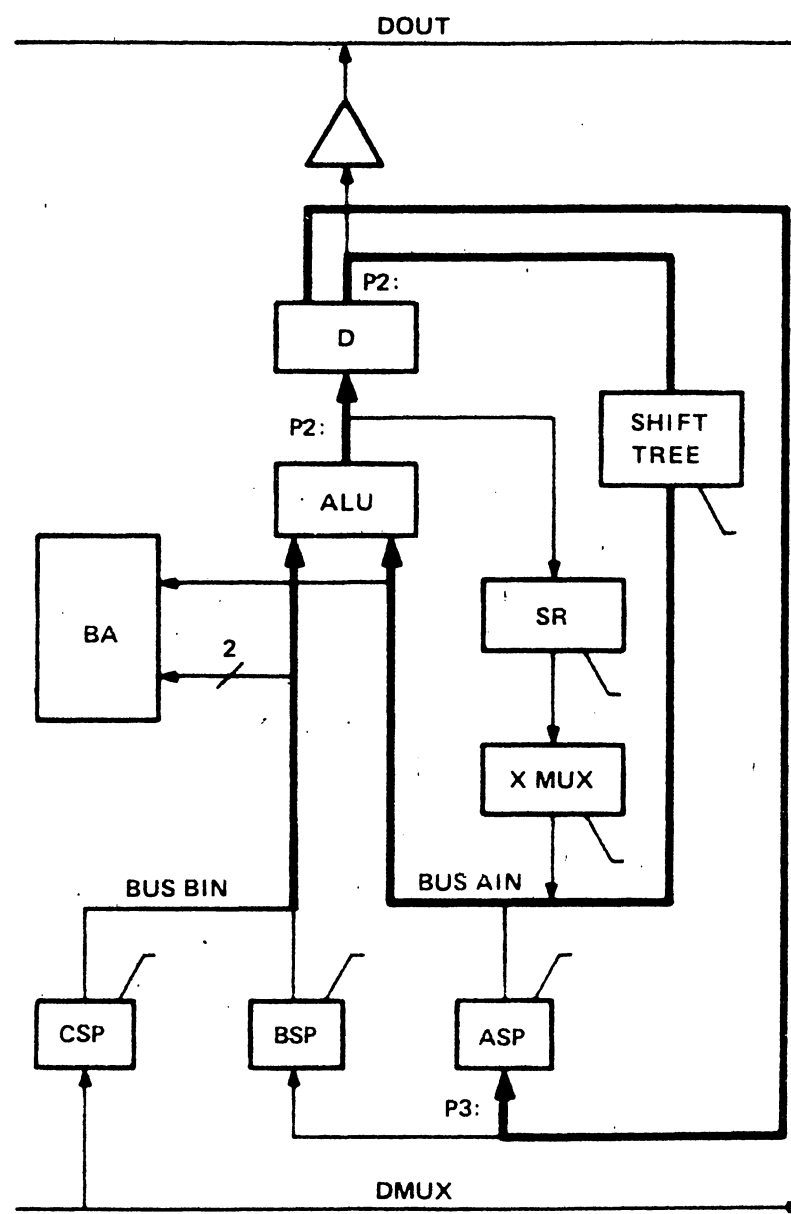
K405 ENB XMUX (0) H (Output Control)	K409 XMUX SEL 0 (1) H (Select)	Port A Data	Port B Data	Output Data BUS AIN (15:00) H
H	X	X	X	High Z
L	L	L	X	L
L	L	H	X	H
L	H	X	L	L
L	H	X	H	H

3.5 THE SHIFT TREE

In the data path, between the D register and BUS AIN, there are a series of multiplexers called the shift tree (Figure 3-1). The shift tree is a path for repositioning data within one microcycle. The two examples in Figure 3-4 illustrate this. In example A, the data is moved at P3 time from the D register through the shift tree, where it is repositioned, and then, via BUS AIN, to the ALU where additional data manipulation takes place with data from the B bus. At P3 time, the data is returned to the D register. A similar path is followed in example B. A complete loop back to the ALU can be made at P2 time and the data can then be written into either the A or B scratchpad at P3 time of the same microcycle. (Refer to the notes on Figure 3-4.)



A. D Register to D Register.



B. D Register to D Register to A or B Scratch Pad.

NOTES:

1. IF THE D REGISTER IS CLOCKED AT P2 THEN ANY SHIFT TREE FUNCTION AND ANY ALU FUNCTION CAN BE CLOCKED INTO THE D REGISTER AT THE NEXT P2.
2. IF THE D REGISTER IS CLOCKED AT P3 THEN ANY SHIFT TREE FUNCTION AND ANY ALU FUNCTION CAN BE CLOCKED INTO THE D REGISTER AT THE NEXT P3.

11-5575

Figure 3-4 Microcycle Shift Tree Operations

Three stages or levels of hard-wired multiplexers make up the shift tree: AMUX (not to be confused with the cache module AMUX), BMUX, and CMUX. Also included is the single chip shift end multiplexer (SENDMUX). The counter register, although connected, is not considered as part of the shift tree (Figure 3-5). D register data, K403 D(15:00) (1) H, is input to the AMUX (high and low), passed on to the BMUX, and then into the CMUX from where it is available to BUS AIN. Data manipulation takes place in each multiplexer stage. Together, the three stages of the shift tree can perform the following operations:

- Sign extend
- Byte swap
- Left shift by 1 place
- Right shift by 1 through 14 places

Three microword fields select the operations performed by each multiplexer stage in the shift tree: AMUX [microword bits (22:00)], BMUX (microword bit 23), and CMUX [microword bits (37:36)]. Table 3-6 lists these microword bits, the signals they develop in the data path module, and the functions they cause in the multiplexers.

3.5.1 The AMUX

The AMUX is divided into two parts: AMUXHI for the high byte and AMUXLO for the low byte. Both high and low data bytes from the D register go to both AMUXHI and AMUXLO. This allows duplicating either byte, swapping bytes, and shifting eight bits to the right. [The latter operation is the result of the mnemonic 8 D(C)#D(HI).] Note that two multiplexer types are employed: AMUXHI consists of four, 4-to-1, 74S153 multiplexer chips, while AMUXLO is constructed of two, 2-to-1 74S157 multiplexer chips.

3.5.1.1 The Counter Register – The D port of AMUXHI receives the 7-bit output of the counter register, which consists of two, 93S16 counters connected in cascade. Operating as an iteration counter, the counter register is loaded from BUS BIN with the complement of the desired count and clocked (P2 L) if K207 ENB CNTR H, the count enable input, is asserted. When the counter is used in a loop it can be at the beginning of the loop before the counted operation (use 1's complement) or at the end of the loop after the operation to be counted (use 2's complement to load the counter).

Example:

	Desired count = 14_8
	364_8 (2's complement of 14_8)
	+ 13_8 (count – plus 1 for the load operation)
(CNTR=ONES)	$1\ 377$

K207 ENB CNTR H is asserted if BUT (COUNT) (Paragraph 4.5.2) is decoded. At a full count, the carry-out signal from the counter register, K409 CNTR = ONES H, is asserted on the last clock pulse, i.e., when the counter equals 377_8 and P2 L occurs. Input to the UBRANCH MUX (Paragraph 4.5), K409 CNTR = ONES H is used in BUT (COUNT) operations to exit from a program loop.

3.5.2 The BMUX

The BMUX contains four, 2-to-1, 74S157 multiplexer chips. The two functions are: (Port A), straight pass through of the data from the AMUX, and (Port B), shift right four places with D(C) filling in the high four bits.

3.5.3 The CMUX

The CMUX can perform four functions that include three shift operations and a straight pass through of the data from the BMUX (Table 3-6). Consisting of eight, 4-to-1, 74S253 multiplexer chips, the CMUX output is placed on BUS AIN for input to the ALU or the BA register.

Table 3-6 Shift Tree Selection Control

AMUX Microword Bits (22:20)	Mnemonic	Function
0	Direct	Input data not changed
1	LO#LO	D(7:0) goes to both bytes
2	Extend	D(C) fills high byte, D(7:0) goes to low byte
3	Counter	Counter register to high byte, D(7:0) goes to low byte
4	Hi#Hi	D(15:8) goes to both bytes
5	Swap	Swap bytes
6	8 D(C)#D(HI)	D(C) fills high byte, D(15:8) goes to low byte
7	Count#D(HI)	Counter register to high byte, D(15:8) goes to low byte

BMUX

K404 BMUX S0 (1) H
(Microword bit 23)

0	Direct	Input data not changed
1	Right Four	Shift input right four, D(C) fills high bits

CMUX

K407 CMUX (S1:S0) (1) H
Microword bits (37:36)

0	Left One	Shift input left one, SENDMUX bit to low bit
1	Direct	Input data not changed
2	Right One	Shift input right one, D(C) goes to high bit
3	Right Two	Shift input right two, D(C) goes to 2 high bits

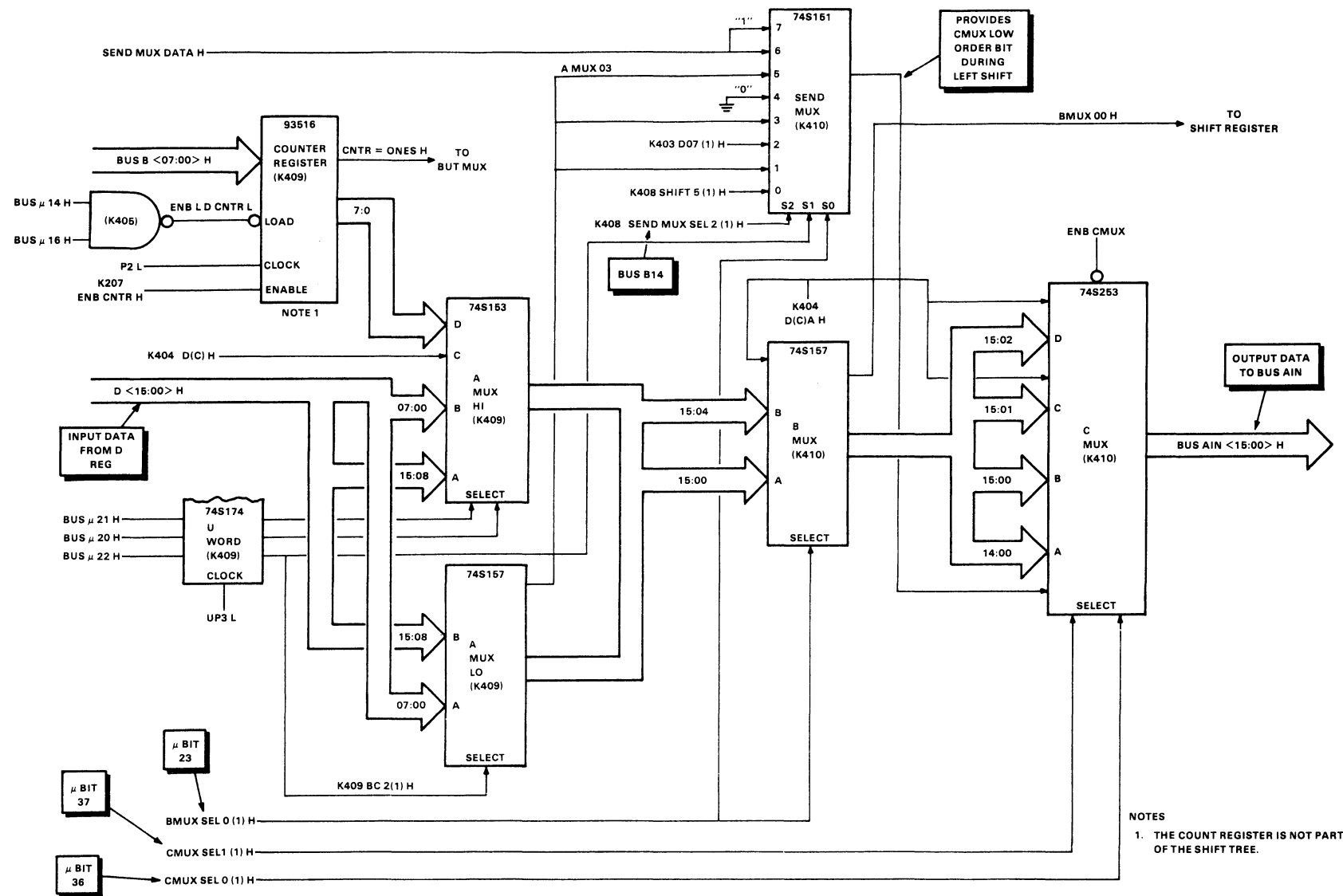


Figure 3-5 Counter Register and Shift Tree

3.5.4 The SENDMUX

The SENDMUX provides the low-order input bit to the CMUX when the CMUX performs a shift left. An example of this is seen in the shift right seven operation represented in Figure 3-6. To accomplish this, the shift tree must first shift right eight and then shift left one. On completion, input bit 7 should be the output's LSB. However, bit 7 is lost in the process of storing the shift right eight in the AMUX. This is where the SENDMUX performs its function. The SENDMUX directs bit 7 from the DOUT lines to the CMUX, where it becomes the LSB.

Consisting of a single 8-to-1, 74S151, multiplexer, the SENDMUX input is selected by microword bits (23:22) and K410 SEND MUX S2 (1) H from the RES CNTL register (Paragraph 3.4.1 and Figure 3-3). Table 3-7 lists the outputs for the different selection input configurations.

3.6 A AND B SCRATCHPADS

The principle storage elements in the data path are two 32-word scratchpads, ASP and BSP (Figure 3-7). Each scratchpad is divided into two, 16-word sections, high and low. The scratchpads may be considered as a series of 16-bit, addressable registers for storing information that is circulated in the data path.

Scratchpad assignments are shown in Figure 3-8. (See K402 for a more complete SP allocation chart and the MED codes to access each location.) The eight PDP-11 general-purpose registers, R0-R7, are duplicated in both scratchpads. This facilitates movement of data in registers to register operation. Data from either or both scratchpads is delivered to the ALU via BUS AIN and BUS BIN, clocked into the D register, and returned to either or both scratchpads - all within one microcycle. Both scratchpads also contain identical copies of the user stack pointer (R16). Two locations in each scratchpad, WCS(0) and WCS(1), are dedicated to the writable control store. The remaining scratchpad locations are for use as temporary storage for floating point, WCS, and base machine operations by the microcode.

Table 3-7 SENDMUX Output

Selection (Note)				Output
S2	S1	S0		
BUS B14=1	0	0	0	K408 SHIFT 15 (1) H
	0	0	1	K409 AMUX 03 H
	0	1	0	K407 D07 (1) H
	0	1	1	K409 AUX 03 H
	1	0	0	0
	1	0	1	K409 AMUX 03 H
	1	1	0	SEND MUX DATA H
	1	1	1	SEND MUX DATA H

Note: S2: K408 SEND MUX SEL 2 (1) H (BUS B14 H)
S1: K409 BC 2 (1) H (BUS U22 H)
S0: K404 BMUX SEL 0 (1) H (BUS U23 H)

3-15

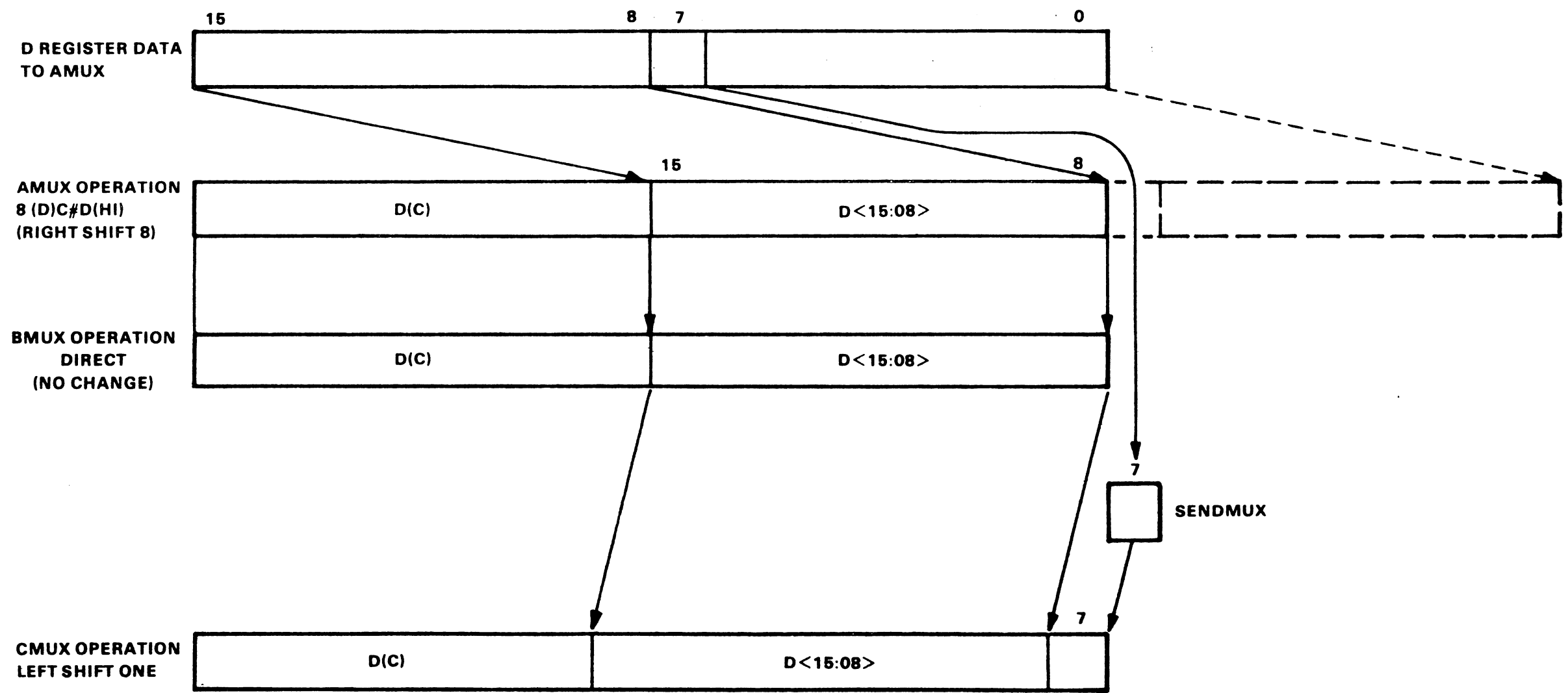
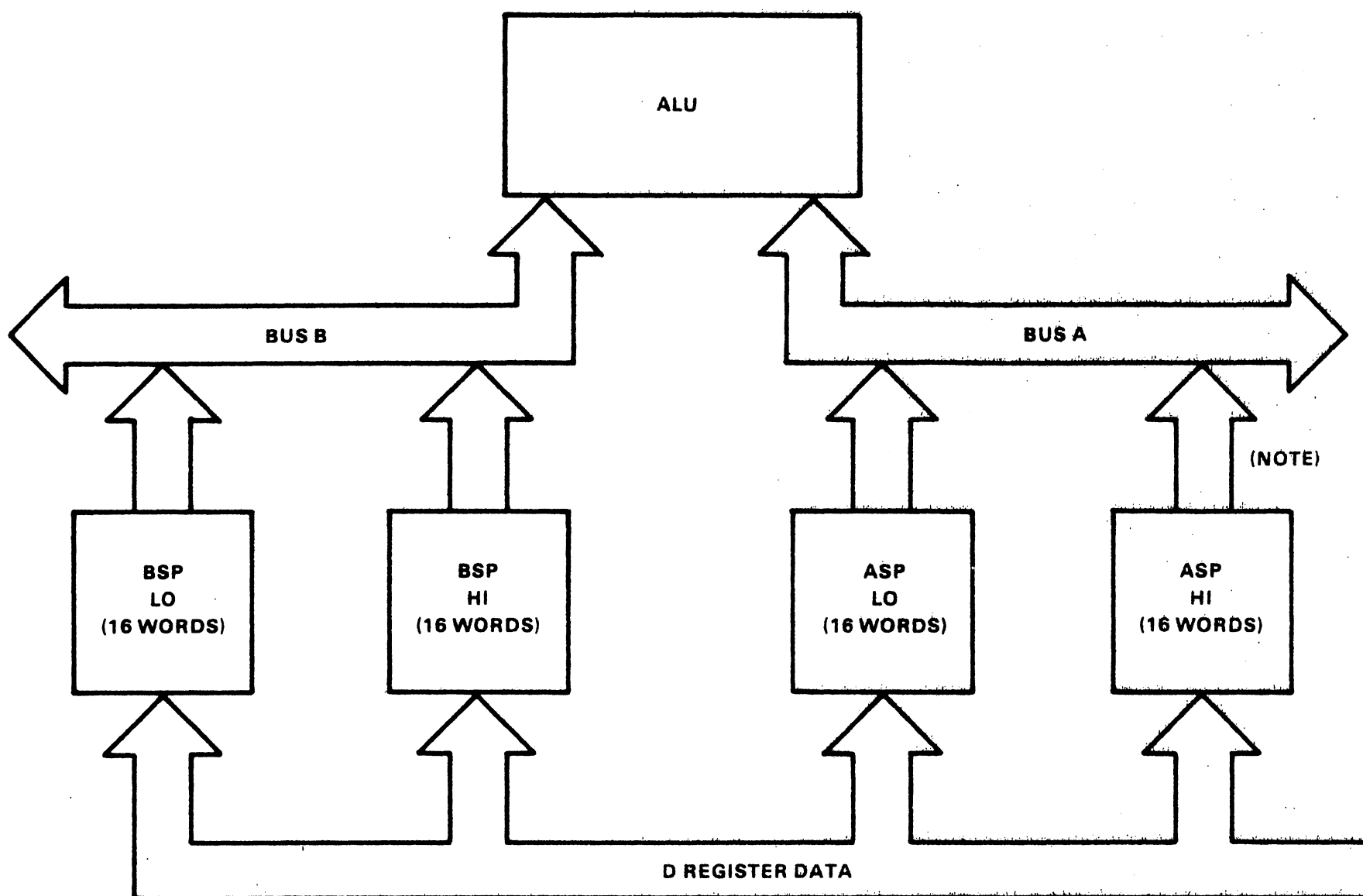


Figure 3-6 Shift End Multiplexer Operation
(Example: Shift Right Seven)



NOTE:
OUTPUTS ARE TRI-STATE

11-5578

Figure 3-7 A and B Scratchpads

	BSPLO	BSPHI	ASPLO	ASPHI
0	R0	WCSB[0]	R0	WCSA[0]
1	R1	WCSB[1]	R1	WCSA[1]
2	R2	WCSADR	R2	GEN-WHAMI
3	R3	R(ZERO)	R3	CNSL.TMPSW
4	R4	R(T1B)	R4	R(T1A)
5	R5	R(T2B)	R5	R(T2A)
6	R6(SP)	R(VECTSAV)	R6(SP)	CNSL.SW
7	R7(PC)	CNSL.CNTL	R7(PC)	CNSL.CDR
10	FAC2[0]	FAC0[0]	FAC3[0]	FAC1[0]
11	FAC2[1]	FAC0[1]	FAC3[1]	FAC1[1]
12	FAC2[2]	FAC0[2]	FAC3[2]	FAC1[2]
13	FAC2[3]	FAC0[3]	FAC3[3]	FAC1[3]
14	FAC2[4]	FAC0[4]	FAC3[4]	FAC1[4]
15	FAC2[5]	FAC0[5]	FAC3[5]	FAC1[5]
16	USER R6	FEA	USER R6	FPSHI-FEC
17	FDSD2	FDST0	FDST3	FDST1

11 5579

Figure 3-8 Scratchpad Assignments

3.6.1 Scratchpad Address Selection

Figure 3-9 illustrates the major elements in the ASP addressing scheme. (The following discussion pertains to the ASP; the BSP is similar.)

A 16-bit location in the ASP is accessed through the use of two, 2-bit microword fields, A enable (AEN) and A select (ASEL). The AEN field [microword bits (39:38)] selects either ASPLO or the ASPHI, while the ASEL field [microword bits (37:36)] controls the ASPMUX and selects one of 16 locations in either the ASPLO or the ASPHI.

The 4-bit ASP address is the product of two, 4-to-1, 74S153 multiplexer chips called the ASP address multiplexer. Table 3-8 lists the ASEL field bit combinations that select the different multiplexer input ports. Ports A and B allow a 4-bit address, developed from the microword register immediate field (R17), bits (35:33), and bit 36 of the ASEL field, to select a scratchpad location other than a general register. This is defined as direct addressing. The other two ports are for indirect addressing, where a pointer is used to specify a general register address. Port C accepts the IR destination field of the macrocode instruction (register mode), and port D the source field, for use as the scratchpad address. The address sources for ports C and D contain address modifiers for use in cache and floating point operations.

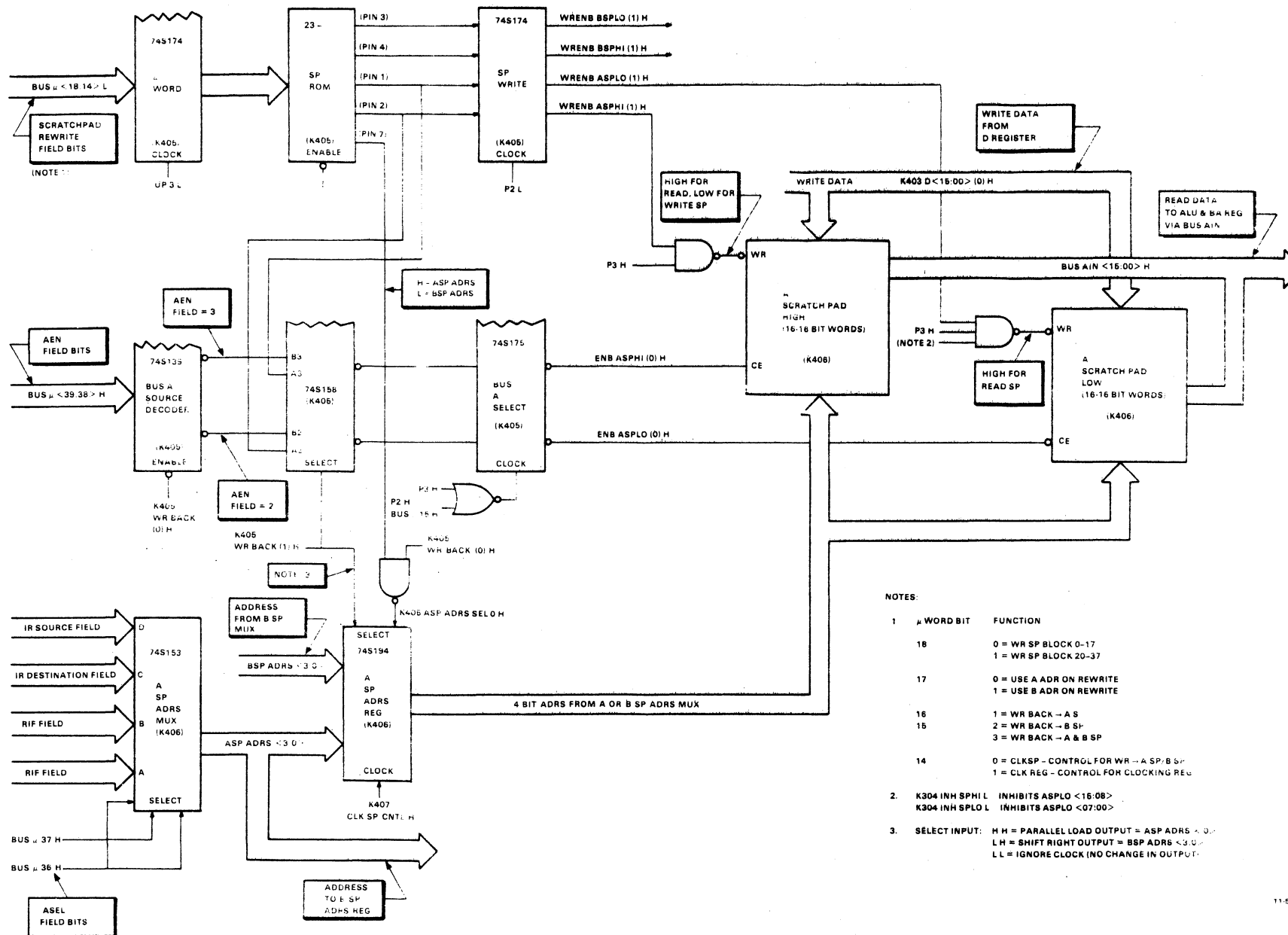


Figure 3-9 Scratchpad Addressing (ASP)

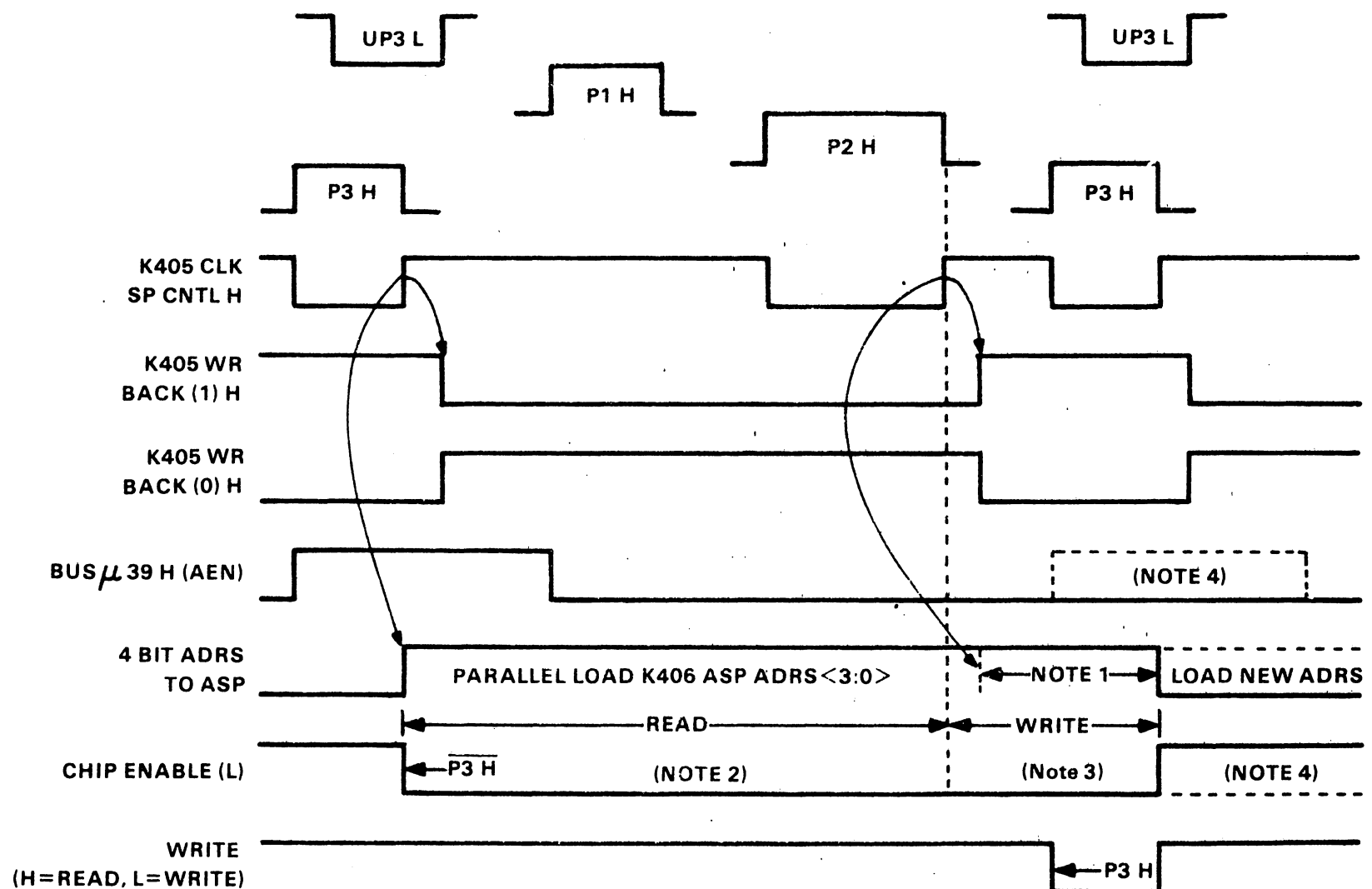
Table 3-8 ASP Address Sources

ASEL Field BUS U (37:36)	ASP ADDR MUX Port Selected	Address Source			
		ASP ADRS A3	ASP ADRS A2	ASP ADRS A1	ASP ADRS A0
0 0	A	(See Note 1)	BUS U34 H	BUS U33 H	BUS U36 = 0 (Note 2)
0 1	B	BUS U35 H	BUS U34 H	BUS U33 H	BUS U36 = 1 (Note 3)
1 0	C	DST ADRS 3 H	IR 02 (1) H	IR 01 (1) H	IR 00 (1) H
1 1	D	SRC ADRS 3 H	IR 08/FLTPT INH H	IR 07 (1) H	IR 06/SPADRS+1 H

- Notes: 1. Derived from [U(KT STACK)L + PS(15:14) (1) H] AND BUS U35 H (although the received signal is BUS U35 H, it is asserted as a low when used in the RIF field, i.e., RIF=6 is U35, U34, U33.)
2. Select even numbered locations: 0, 2, 4, 6, 16.
3. Select odd numbered locations: 1, 3, 5, 7, 17.

The 4-bit scratchpad address from the ASP address multiplexer is connected to four of the D inputs on two 74S194 shift register chips, the ASP address register. [The other four D inputs are the BSP address, K407 BSP ADRS (A3:A0) H. They may be used on a rewrite to the ASP (Paragraph 3.6.12)]. If both ASP address register select inputs [K405 WR BACK (1) H and K405 ASP ADRS SEL 0 H are high] the ASP address is parallel loaded and available to ASPLO and ASPHI after the positive transition of K405 CLK SP CNTL H (P3 time) (Figure 3-10). To access the scratchpad location, the address must be accompanied with a low to the chip enable (CE) input. For a write to the scratchpad the write (WR) input must also be low.

Table 3-9 lists the BSEL field bit combinations that select the sources for BSP addresses.



NOTES:

1. ROM PIN 7 HIGH = IGNORE CLOCK - USE ASP ADRS FOR WRITE
ROM PIN 7 LOW = SHIFT RIGHT - USE BSP ADRS FOR WRITE
2. CHIP ENABLE DERIVED FROM BUS A SOURCE DECODER
3. CHIP ENABLE DERIVED FROM SP ROM
4. CHIP ENABLE REMAINS LOW IF THE NEXT MICROWORD HAS BUS μ 39 H = '1

11-5581

Figure 3-10 A Scratchpad (ASP) Timing

72

Table 3-9 BSP Address Sources

BSEL Field BUS U (41:40)	BSP ADDR MUX Port Selected	Address Source			
		BSP ADRS A3 H	BSP ADRS A2 H	BSP ADRS A1 H	BSP ADRS A0 H
0 0	A	DST ADRS 3 H	IR 02 (1) H	IR 01 (1) H	IR 00 (1) H
0 1	B	SCR ADRS 3 H	IR 08/FLTPT INH H	IR 07 (1) H	IR 06/SPADRS+1 H
1 0	C	U35 H	U34 H	U33 H	U40 = 0 (Note 1)
1 1	D	U35 H	U34 H	U33 H	U40 = 1 (Note 2)

Notes: 1. Select even numbered locations: 0,2,4,6, 16.

2. Select odd numbered locations: 1,3,5,7, 17.

3.6.1.1 Chip Enable – One of two CE signals must be asserted (low) to read or write to the scratchpad. K405 ENB ASPHI (0) H and K405 ENB ASPLO (0) H are the CE signals for ASPHI and ASPLO, respectively. These signals, from the ASP enable register, originate at the BUS AIN source decoder for a read SP and at the SP ROM for a write SP. (The CE signal is usually kept asserted during the entire microcycle because a read SP is normally followed by a write SP.)

The BUS AIN source decoder occupies half of a double, two-line-to-four-line, 74S139 decoder chip (the other half of the chip decodes the BUS BIN CE signals). Inputs to the decoder consist of the AEN field bits, BUS U(39:38) H, and the enable input, K405 WR BACK (0) H. When this latter signal goes low (after the trailing edge of P2 H) one of the four output pins is low (Table 3-10). The low from the decoder is clocked into the BUS A select register at P3 time. This is for a read SP.

In a rewrite, when the data is written back into the scratchpad after being read out to the D register, the CE signal is developed from one of the SP ROM outputs (Paragraph 3.6.1.2). At the trailing edge of P2 H, the ROM asserts either K405 ENB ASPHI (0) H (low) to enable the ASPHI or K405 ENB ASPLO (0) H (low) to enable the ASPLO.

3.6.1.2 Read/Write Input – The logic level to the read/write (WR) inputs on the 74S189 scratchpad chips determines which internal function, read or write, is performed. A read occurs when the WR input is high; a write occurs when the WR input is low. (The CE input must also be low for a read or a write.)

The logic levels that cause a read or a write are developed from signals whose origin is the scratchpad rewrite field, BUS U(19:14). Five of these microword bits, BUS U(18:14) control the A and B scratchpad read/write function. (BUS U19 controls the CSP, Paragraph 3.7. Figure 3-10 shows these five bits as input to a U word register on K405. ROM receives a configuration of these bits and asserts one or more write enable and select signals (Table 3-11). There are two destinations for the SP ROM outputs:

1. The BUS A and BUS B select registers, where they provide the CE input(s) on a rewrite,
2. The SP write register, whose outputs enable one or more WR gates at P3 time.

In reference to the ASP, a write to ASPHI is effected when K405 WR ENB ASPHI (1) H enables the WR input at the same time the SP ROM output asserts K405 ENB ASPHI (0) H (low) to enable the CE input. Three ASP write options are available:

1. Write to ASPHI
2. Write to ASPLO (as a word operation)
3. Write to only the low byte at the ASPLO location.

This is the result of inhibiting the WR inputs to the high byte of the ALSPLO. K306 INH SPHI L, asserted for instructions whose destination is a register (R0–R7), inhibits the WR inputs on ASPLO (15:08). Thus, the data in the high bytes of the registers is unaffected at this time. Another signal, K306 INH SPLO L, is asserted (along with K306 INH SPHI L) during a macroinstruction fetch when K207 BUT (INST1) H is asserted in the FET03 microword and there is no fetch overlap (K306 OVERLAP L). Both inhibit signals, asserted together, prevent writing to both high and low bytes in ASPLO. This prevents the PC from being updated at this time.

Table 3-10 AEN and BEN Field Signals

AEN Field BUS U (39:38)	BUS A+B Source Decoder Output Pin (Low)	BUS A+B Select Register Output Asserted (Note)	Function
0 0	12	K405 ENB XMUX (0) H	XMUX Output Control
0 1	11	K405 ENB CMUX (0) H	CMUX Output Control
1 0	10	K405 ENB ASPLO (0) H	ASPLO Chip Enable
1 1	9	K405 ENB ASPHI (0) H	ASPHI Chip Enable
BEN Field BUS U (43:42)			
0 0	4	K405 ENB BSPLO (0) H	ESPLO Chip Enable
0 1	5	K405 ENB BSPHI (0) H	BSPHI Chip Enable
1 0	6	No Output	Select arbitrary CSP Location (K407)
1 1	7	No Output	Select base constants in CSP (K407)

Note: Signals are low when asserted, i.e., (1) L.

Table 3-11 A and B Scratchpad ROM Patterns (K405)

Inputs						Outputs								Resultant Signals	Operation (Note 1)
Scratchpad Rewrite Field Mnemonic: MOD WRSP WR HI/LO						BSP ADRS S0 H	ASP ADRS S0 H	Not Used	Not Used	WRENB BSPHI	WRENB BSPLO	WRENB ASPHI	WRENB ASPLO		
BUS U	14	15	16	17	18										
		SEL													
		0=SP	WRT	WRT	Q=A	Q=LO									
		1=BC	A	B	1=B	1=HI									
Chip Input:	A4	A3	A2	A1	A0	M7	M6	M5	M4	M3	M2	M1	M0		
	0	0	0	0	0	1	1	0	0	0	0	0	0	Not Used	
	0	0	0	1	1	1	1	0	0	0	0	0	0		
	0	0	1	0	0	0	1	0	0	0	1	0	0	K405 BSP ADRS S0 H, K405 WRENB BSPLO (1) H, K405 ENB BSPLO (0) H	Write BSPLO (ASP ADRS)
	0	0	1	0	1	0	1	0	0	1	0	0	0	K405 BSP ADRS S0 H, K405 WRENB BSPHI (1) H, K405 ENB BSPHI (0) H	Write BSPHI (ASP ADRS)
	0	0	1	1	0	1	1	0	0	0	1	0	0	K405 WRENB BSPLO (1) H, K405 ENB BSPLO (0) H	Write BSPLO (BSP ADRS)
	0	0	1	1	1	1	1	0	0	1	0	0	0	K405 WRENB BSPHI (1) H, K405 ENB BSPHI (0) H	Write BSPHI (BSP ADRS)
	0	1	0	0	0	1	1	0	0	0	0	0	1	K405 WRENB ASPLO (1) H, K405 ENB ASPLO (0) H	Write ASPLO (ASP ADRS)
	0	1	0	0	1	1	1	0	0	0	0	1	0	K405 WRENB ASPHI (1) H, K405 ENB ASPHI (0) H	Write ASPHI (ASP ADRS)
	0	1	0	1	0	1	1	0	0	0	0	0	1	K405 ASP ADRS S0 H, K405 WRENB ASPLO (1) H, K405 ENB ASPLO (0) H	Write ASPLO (BSP ADRS)
	0	1	0	1	1	1	1	0	0	0	0	1	0	K405 ASP ADRS S0 H, K405 WRENB ASPHI (1) H, K405 ENB ASPHI (0) H	Write ASPHI (BSP ADRS)
	0	1	1	0	0	0	1	0	0	0	1	0	1	{ K405 BSP ADRS S0 H, K405 WRENB ASPLO (1) H, K405 ENB ASPLO (0) H } K405 WRENB BSPLO (1) H, K405 ENB BSPLO (0) H	Write ASPLO and BSPLO (ASP ADRS)
	0	1	1	0	1	0	1	0	0	1	0	1	0	{ K405 BSP ADRS S0 H, K405 WRENB ASPHI (1) H, K405 ENB ASPHI (0) H } K405 WRENB BSPHI (1) H, K405 ENB BSPHI (0) H	Write ASPHI and BSPHI (ASP ADRS)
	0	1	1	1	0	1	0	0	0	0	1	0	1	{ K405 ASP ADRS S0 H, K405 WRENB ASPLO (1) H, K405 ENB ASPLO (0) H } K405 WRENB BSPLO (1) H, K405 ENB BSPLO (0) H	Write ASPLO and BSPLO (BSP ADRS)
	0	1	1	1	1	1	0	0	0	1	0	1	0	{ K405 ASP ADRS S0 H, K405 WRENB ASPHI (1) H, K405 ENB ASPHI (0) H } K405 WRENB BSPHI (1) H, K405 ENB BSPHI (0) H	Write ASPHI and BSPHI (BSP ADRS)
	1	0	0	0	0	1	1	0	0	0	0	0	0	Not Used	
	1	1	1	1	1	1	1	0	0	0	0	0	0		

Note 2

Notes: 1. The high or high and low bytes of the registers in ASPLO and BSPLO can be inhibited (Paragraph 3.6.1.2)

2. Must be used when a general register (R0-R7) is included in an instruction.

74B

3.6.2 BUS AIN and BUS BIN

These two busses, internal to the data path module, provide the connecting link between the ALU and certain registers and multiplexers (Table 3-12). The relationship of these components to BUS AIN and BUS BIN is seen in Figure 3-1. All inputs to the busses are from tristate devices. Devices are selected, i.e., enabled onto the busses, by the AEN [BUS U(39:38)] and BEN [BUS U(43:42)] microcode fields (Table 3-10).

Table 3-12 BUS AIN and BUS BIN Connections

Devices Connected To BUS AIN	Relation To BUS AIN	Devices Connected To BUS BIN	Relation To BUS B
ALU	Output from	ALU	Output from
BA Register	Output from	CSP (Note 1)	Input to
XMUX (Note 1)	Input to	BSPHI (Note 1)	Input to
CMUX (Notes 1 and 2)	Input to	BSPLO (Note 1)	Input to
ASPHI (Note 1)	Input to	BA Register	Output from
ASPLO (Note 1)	Input to	Counter Register	Output from
		Residual Control Register	Output from

Notes: 1. Tristate device.

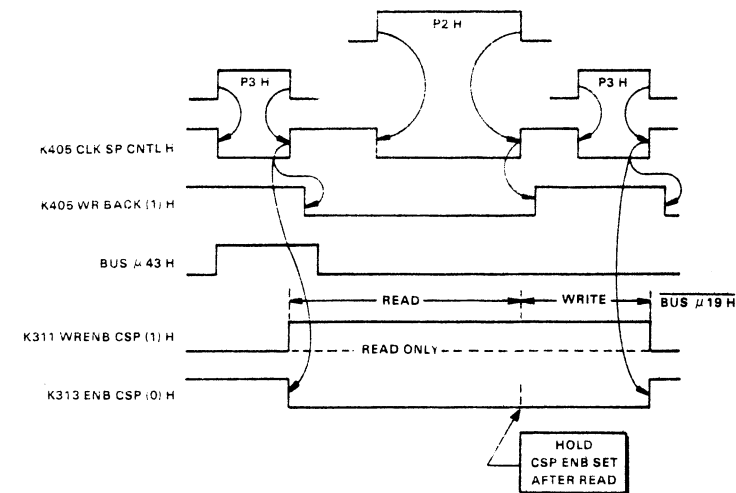
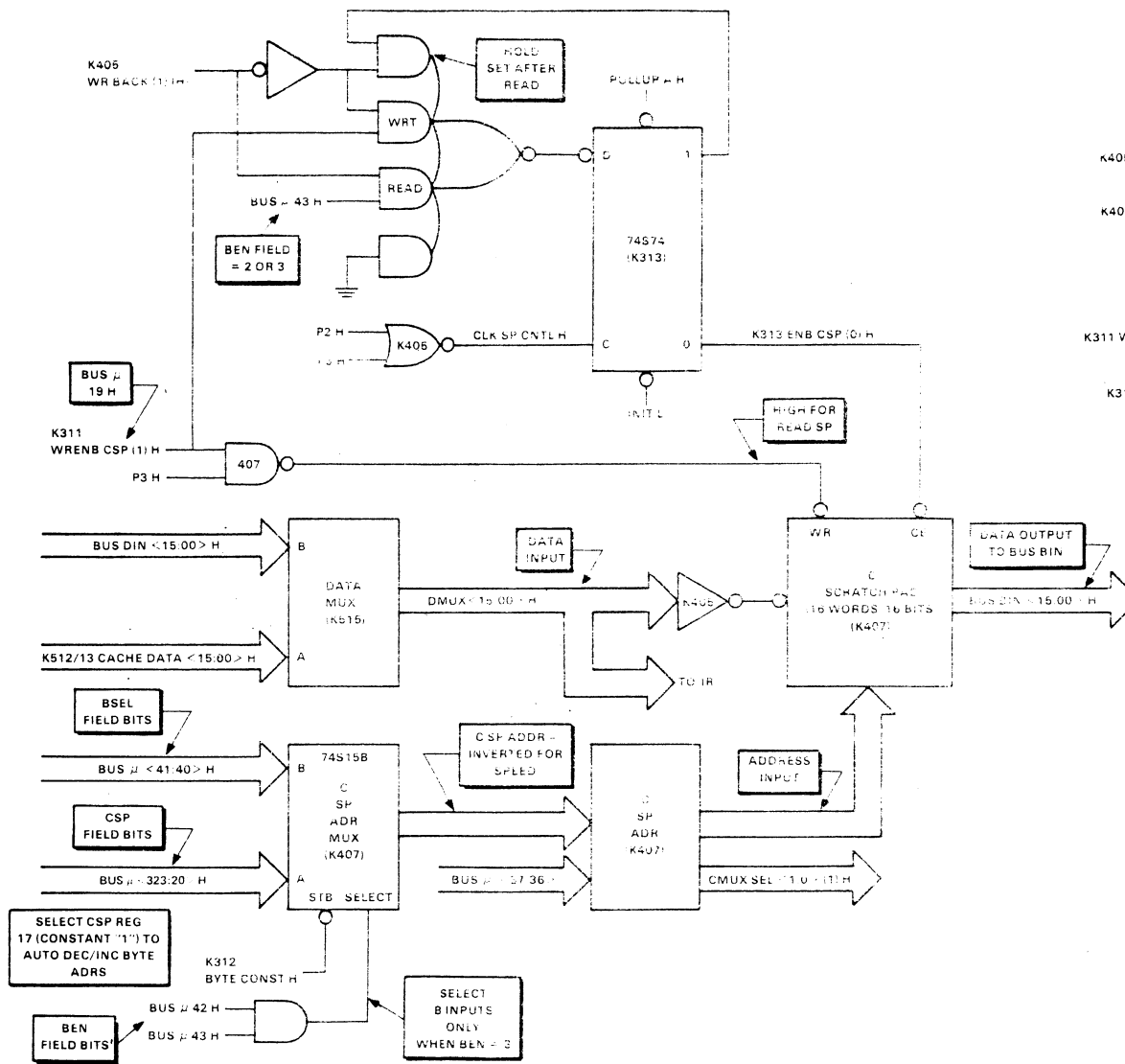
2. The CMUX is the output multiplexer in the shift tree.

3.7 THE CONSTANT SCRATCHPAD (CSP)

The CSP provides the entry point for data from areas external to the data path module: cache memory, main memory (via the Unibus), and the floating point processor and WCS via the DIN lines. This data is routed through the DMUX (Figure 3-11) and temporarily stored in the CSP, where normally it is assigned to the memory data (MD) register. The CSP contains the base constants (0, 1, and 2) that are used in the processor control logic. Other constants, used in the floating point and console logic, also reside in the CSP. The locations used for floating point constants function at other times as log registers. (Only the base constants are inviolate; they should never be changed.)

The CSP consists of four, 4-bit $\times$ 16-word 74S189 RAM chips with tristate outputs (the same chips used in the A and B scratchpads). Address control is provided by the BEN, CSP, and BSEL microword fields. Any one of the 16 CSP locations can be accessed when BEN $\neq$ 3; the CSP field bits, U(23:20), supply the addresses at this time. However, when BEN = 3, the BSEL field bits, M(43:42), furnish the two low-order address bits. The two high-order bits are tied to ground and always equal 1 after inversion. Consequently, when BEN = 3, the only possible CSP addresses are (17:14), the addresses for the base constants and the MD register.

Refer to the address selection table on Figure 3-11. The address developed from the CSP and BSEL field bits is inverted (complemented) and the result is connected to the address inputs of the four CSP chips. Locations 0 through 13 are dual-purpose registers. They are used to store, at different times, floating point constants, error log information, and console registers. Locations 14, 16, and 17 contain the base constants. Location 15 is the MD register.

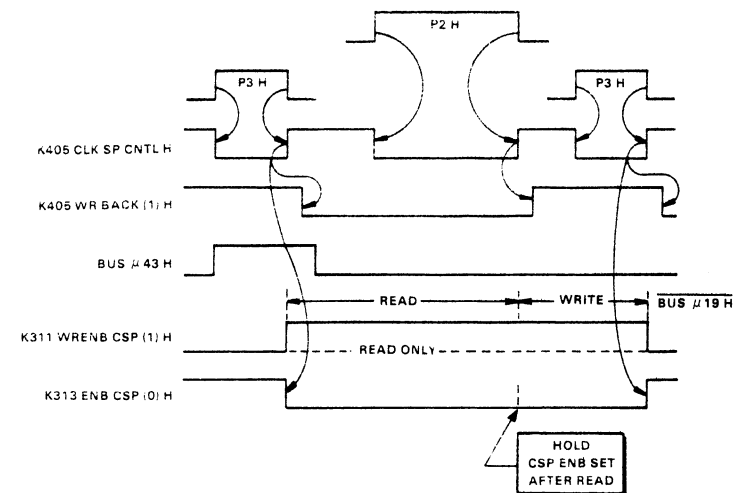
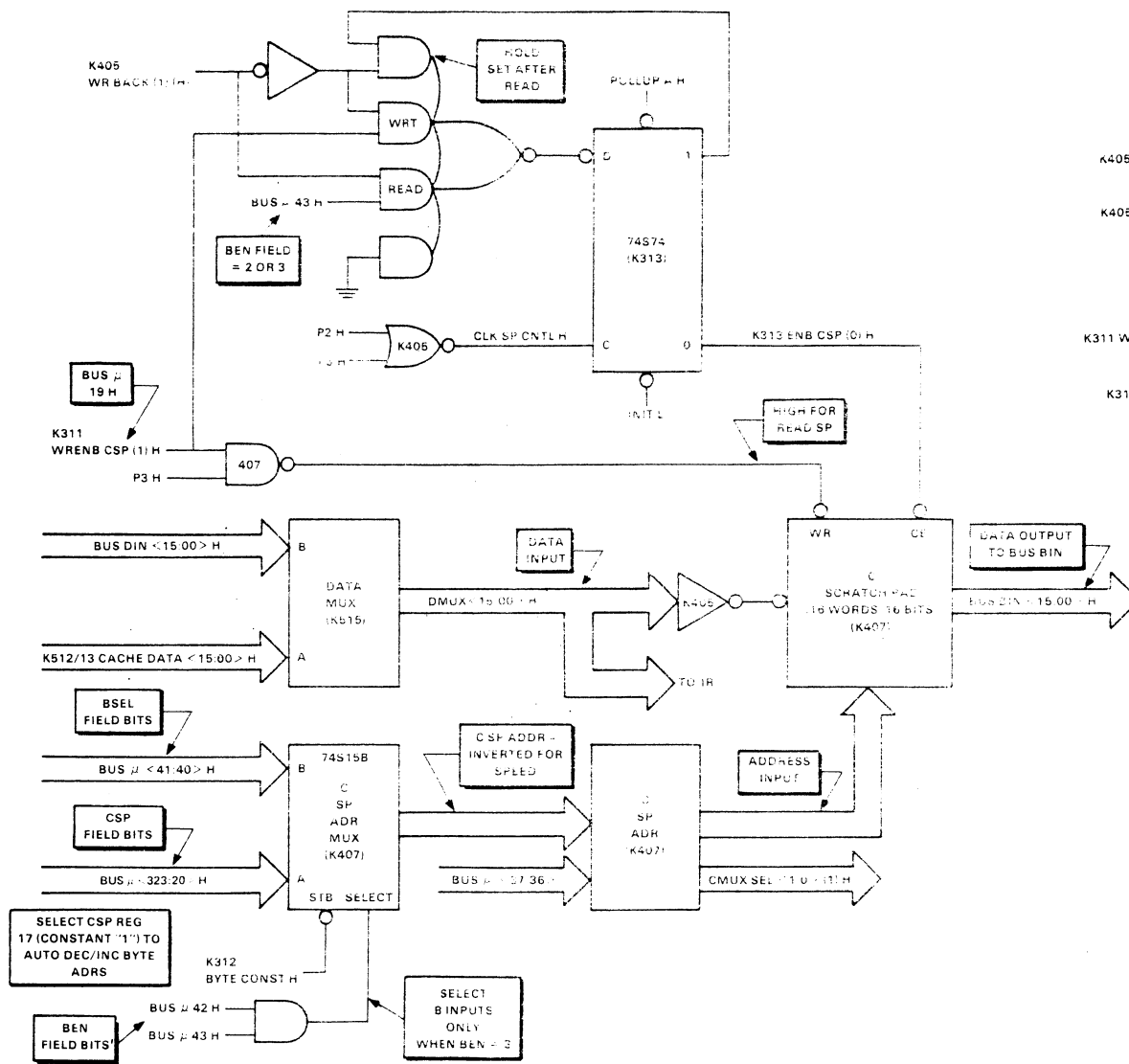


CSP ADDRESS SELECTION

BEN = 2	CSP FIELD VALUE (BUS μ 3:20 H)	CSP ADRS MUX OUTPUT	INITIAL CONTENTS
0	0000	1111 17	1
1	0001	1110 16	0
2	0010	1101 15	MD
3	0011	1100 14	2
4	0100	1011 13	} CONSOLE CONSTANTS
5	0101	1010 12	
6	0110	1001 11	
7	0111	1000 10	
10	1000	0111 7	FP. LOG TAG/CPU
11	1001	0110 6	FP. LOG CACHE DATA
12	1010	0101 5	FP. LOG WHAM I
13	1011	0100 4	FP. LOG FLAG/INTR
14	1100	0011 3	FP. LOG CUA
15	1101	0010 2	FP. LOG PBA
16	1110	0001 1	FP. LOG SERVICE
17	1111	0000 0	FP. LOG JAM
BEN = 3	BSEL FIELD VALUE (BUS μ 41:40 H)		
0	0000	1111 17	1
1	0001	1110 16	0
2	0010	1101 15	MD
3	0011	1100 14	2

11-5582

Figure 3-11 C Scratchpad (CSP)



CSP ADDRESS SELECTION

BEN =2	CSP FIELD VALUE (BUS μ<3:20> H)	CSP ADRS MUX OUTPUT	INITIAL CONTENTS
0	0000	1111 17	1
1	0001	1110 16	0
2	0010	1101 15	MD
3	0011	1100 14	2
4	0100	1011 13	} CONSOLE CONSTANTS
5	0101	1010 12	
6	0110	1001 11	
7	0111	1000 10	
10	1000	0111 7	FP. LOG TAG/CPU
11	1001	0110 6	FP. LOG CACHE DATA
12	1010	0101 5	FP. LOG WHAM I
13	1011	0100 4	FP. LOG FLAG/INTR
14	1100	0011 3	FP. LOG CUA
15	1101	0010 2	FP. LOG PBA
16	1110	0001 1	FP. LOG SERVICE
17	1111	0000 0	FP. LOG JAM

BEN =3	BSEL FIELD VALUE (BUS μ<41:40> H)	CSP ADRS MUX OUTPUT	INITIAL CONTENTS
0	0000	1111 17	1
1	0001	1110 16	0
2	0010	1101 15	MD
3	0011	1100 14	2

11-5582

Figure 3-11 C Scratchpad (CSP)

Chip enable and write enable timing are also shown on Figure 3-11. A low to the CE input, necessary to access the CSP, occurs when the CSP ENB flip-flop is set. A three-input 74S64 AND-OR-INVERT gate supplies the means to set the flip-flop. Two of these inputs, labeled WRT and READ, are controlled by the state of K405 WR BACK (1) H, which is clocked by K405 CLK SP CNTL H at P2 and P3 time. An addressed scratchpad location is read between the trailing edge of P3 and the trailing edge of P2. A write then occurs, providing K311 WRENB CSP (1) H (BUS U19 H) is high, before the trailing edge of the next P3 pulse. After the ENB CSP flip-flop is set by the first P3 pulse, it is held set by a third input to the 74S64. To reset ENB CSP, and conclude access to the scratchpad, the CSP is deselected (BUS U43 H) prior to the trailing edge of the second P3. With all inputs to the 74S64 inhibited (WR BACK is high), ENB CSP is reset at P3 time.

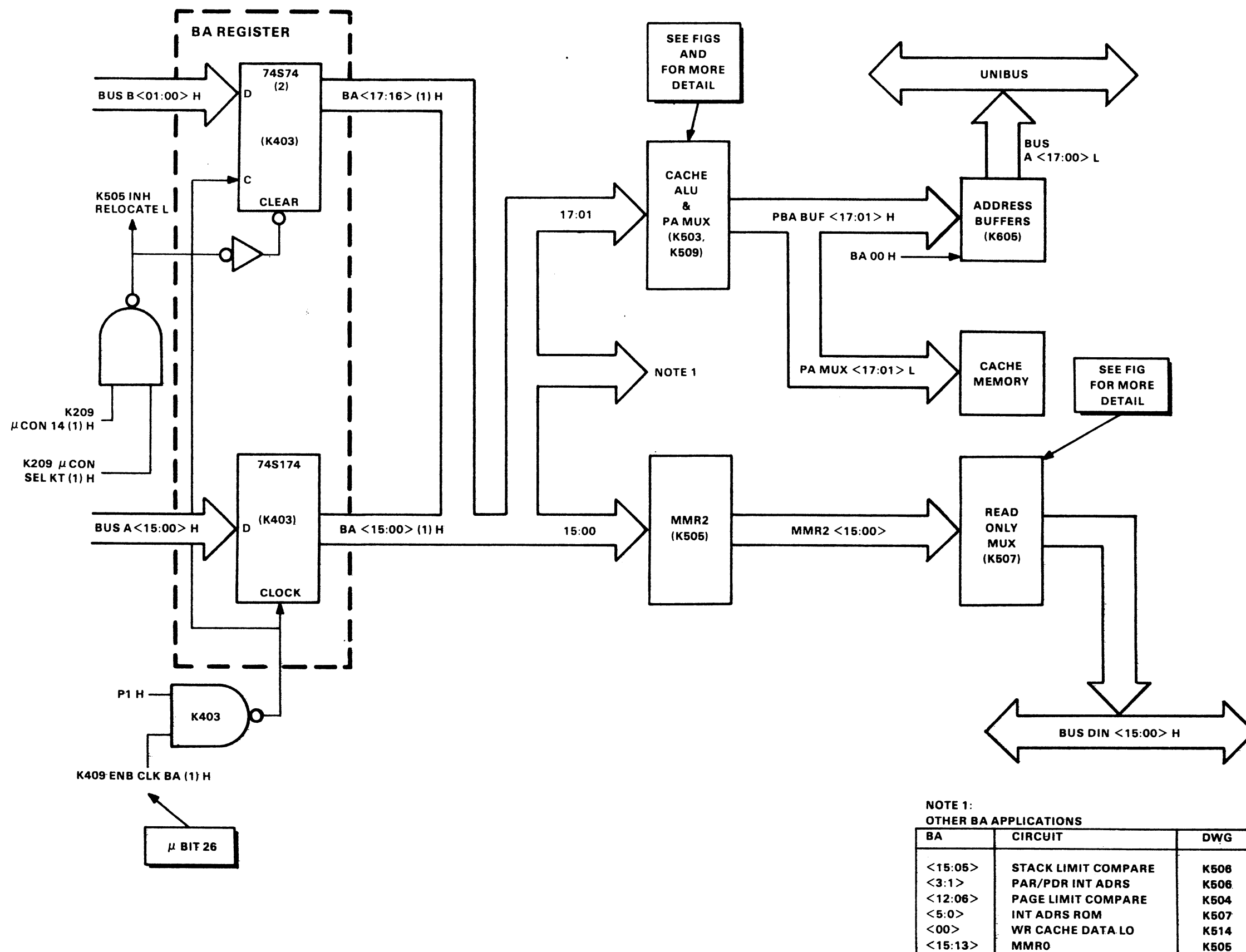
If the macroinstruction in the IR specifies the autoincrement or autodecrement of a byte address, the signal K312 BYTE CONST H is asserted (Paragraph 4.6.3.4). Connected to the strobe input of the CSP MUX, K312 BYTE CONST H disables the multiplexer. This causes all the outputs to go high, i.e., = 17₈, the address of the CSP location that contains a constant of 1. Consequently, with an instruction of this type, the LSB of the data (in BUS BIN) to the ALU is equal to 1, so that the register on BUS AIN will be incremented by one (byte operation) versus two (word operation).

3.8 THE BUS ADDRESS (BA) REGISTER

The BA register provides addresses for read and write operations conducted in cache memory and devices, such as main memory, on the Unibus (Figure 3-12). The BA consists of three, 74S174, hex, D-type registers for address bits (15:00) and two, 74S74, D-type flip-flops for address bits (17:16), all on the data path module.

The inputs to the BA register from the A and B busses are the virtual address bits of the location to be read or written to. This address is routed from the BA register to two cache module circuits: the ALU and MMR2. From MMR2, the address is passed on to the read only multiplexer. Here it is available to the processor for diagnostic purposes during maintenance. The ALU converts the virtual address to a physical memory address (Paragraph 2.2.4.3). As K509 PA MUX (17:06) H (the physical address) is sent to a number of circuits, K509 PA MUX (17:01) L is used to access the various cache stores (tag, parity, valid, and data). K509 PBA (17:06) H and BA (05:00) H are output, via the bus control logic, to the Unibus, where they are used to access a location in a device connected to the Unibus, e.g., memory or a peripheral register.

Control of the BA register is determined by the state of K409 ENB CLK BA (1) H (the single bit microword BA field, U26) which, with P1 H, provides the clock pulse. The two high-order outputs, K403 BA (17:16) (1) H, are not used when the cache memory management logic is functioning, i.e., relocating the virtual address from the BA to a physical address. At this time, the cache ALU produces the two high-order physical address bits, PBA (17:16) H, from the page address register (PAR) bits 10 and 11; the states of BA (17:16) (1) H are not considered. To relocate, K209 UCON 14 (1) H must equal 0. With UCON 14 = 0, K505 INH RELOCATE L is inhibited and BA (17:16) (1) H are held low. When the KT is off, BA (17:16) are used to generate the address. An example of this would be a console examine or deposit.



11-5583

Figure 3-12 BA Register

CHAPTER 4

PROCESSOR CONTROL DETAILED LOGIC DESCRIPTION

4.1 SCOPE

This chapter describes the logic elements and functions of the PDP-11/60 processor control. The processor control consists of two hex boards, M7872 microword and M7873 decode. From a functional standpoint, the boards are considered as one unit; hardware for a particular aspect of processor control is often located on both boards.

4.2 INTRODUCTION

The major elements of the processor control are shown in Figure 4-1 and listed, along with their uses, in Table 4-1. Of these circuits, the focus is on the control store, the heart of the processor control. The other logic either supports the control store operation or uses its output to carry out various functions.

Table 4-1 Processor Control Circuits

Major Circuits	Functions
Control Store Auxiliary Control Store	Contains the 48 bit $\times$ 2560 word microprogram and the 8 bit $\times$ 512 word auxiliary control store extension.
Control Store Address Microbranch MUX Return Register NUA Register CUA	Provides addresses to the control store under various conditions. (branching, return from subroutines, etc.)
Instruction Decode Instruction Register Condition Codes	Instruction decoding and processor status word condition code monitoring and updating.
UCON Interface	Provides for the transfer of control, flag and secondary information within the base machine.

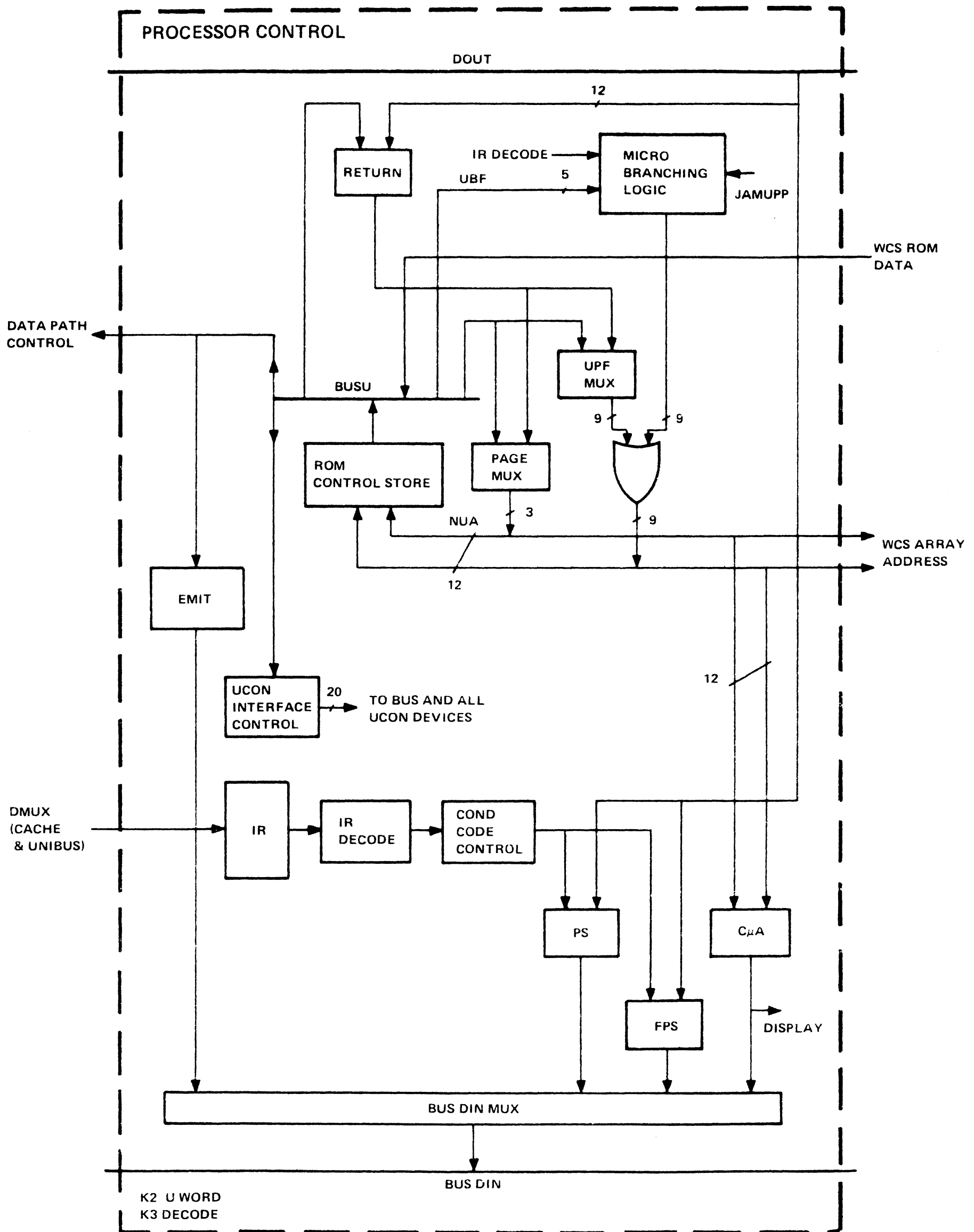


Figure 4-1 Processor Control Block Diagram

4.3 MICROWORD CONTROL STORE

Each instruction in the macroprogram is implemented by one or more microinstructions, or microwords. Microwords, when decoded, supply the basic controls for the operation of the PDP-11/60. However, a microword is not normally viewed as a single, 48-bit word, but as a group of control fields each containing a number of bits with a common function. Table 4-2 lists the print set source and destination of each of the microbits and the control field to which it is assigned. Note that some of the control fields overlap; the same bits may be interpreted in different ways. This is shown in more detail in Figure 4-33.

The microinstructions are stored in a ROM, a storage device whose contents are predetermined and cannot be altered. In the PDP-11/60 this ROM is called the control store. It consists of an array of 60, type 6206 ROM chips depicted on drawings K204, K205, K206, and K309 of the print set. Divided into five pages of 512 words each, the control store contains 2560 48-bit microwords.

4.4 SEQUENCING

Sequencing is the process of retrieving instructions in a prescribed manner. In a computer that executes machine language instructions, sequencing is controlled by a program counter (PC) that contains the address of the next instruction to be brought out from memory. Since the program is usually stored in and executed from contiguous memory locations, the PC is incremented once every time an instruction is read from memory. Branches (both conditional and unconditional) to another part of the program can be made if the normal PC incrementation is stopped and a nonsequential address is inserted into the PC. The processor now retrieves and executes the instruction at this address; instructions at sequentially addressed locations will then be executed until another branch is required.

In the PDP-11/60, instead of using a PC to control sequencing, the address of the next microinstruction is included in the microinstruction presently being executed.* The order in which microinstructions are read from the control store is determined by the micropointer field (UPF) of the present U WORD, which may be in sequence from sequential addresses, from nonsequential addresses, or from addresses that are modified by current machine states or conditions (conditional branches).

The sequencing method employed in the PDP-11/60 is called chained sequencing. It is UPF, BUS U(8:0), of the microinstruction that determines the control store address of the next microinstruction. The associated microbranch field (UBF), BUS U(13:09), is used to select certain machine conditions, or information decoded from the macroinstruction, to modify the address defined by the UPF. The end result is the next microaddress (NUA), which is used to access a control store location.

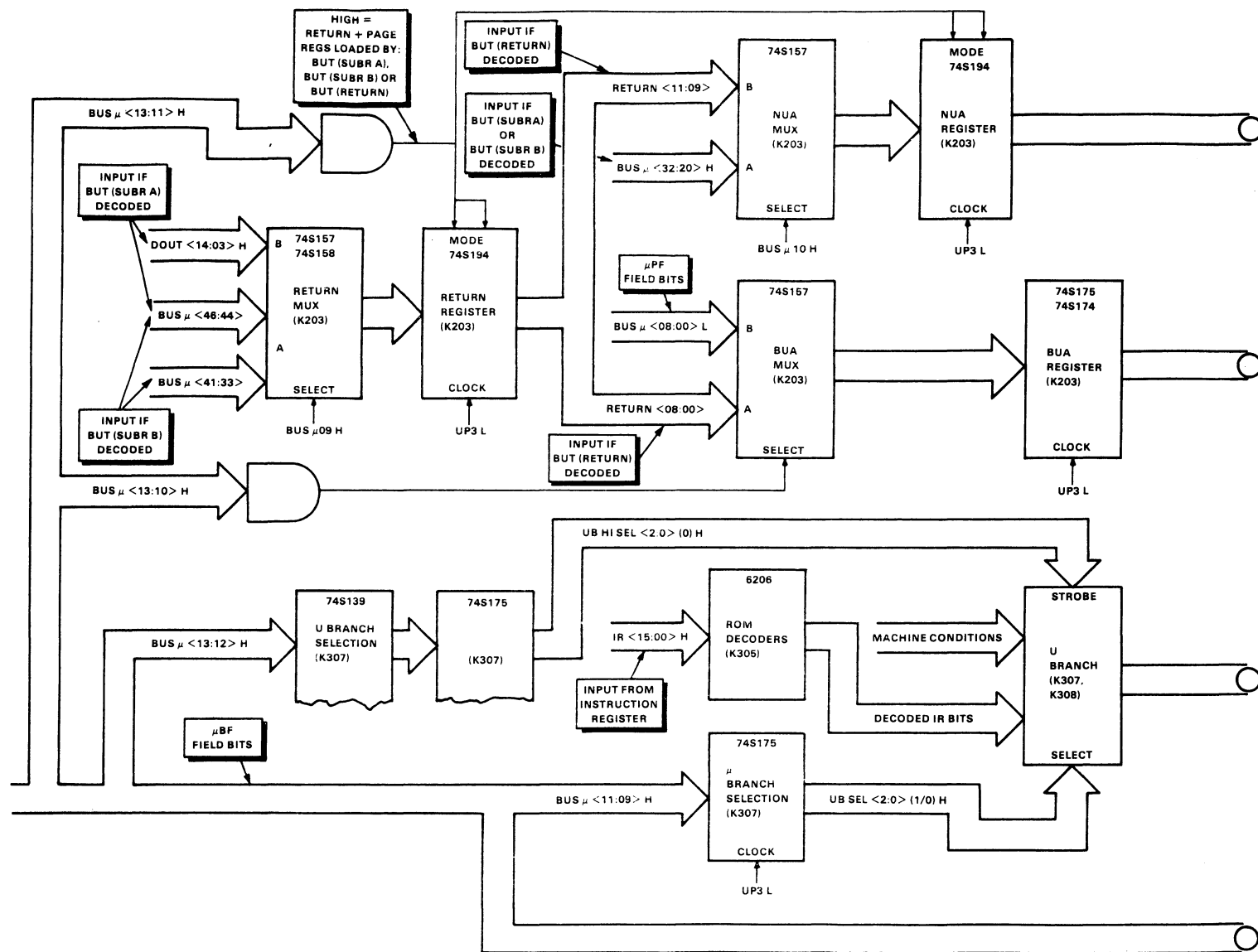
Figure 4-2 shows the control store and the logic used to determine the NUA. Included are the return multiplexer, the NUA multiplexer, the microbranch multiplexer, and other associated circuits. They are described in the following paragraphs.

A control store location is accessed by NUA (11:00). The three high-order address bits, NUA (11:09), are used for page selection. Five different combinations of these bits enable five NAND gates whose outputs are connected to the CE inputs on the control store chips. Each NAND gate provides the CE for 12 of the 60 chips in the store, i.e., one page. (Refer to the table in Figure 4-2.) The nine low-order address bits to the control store [in effect NUA (8:0)] are the outputs from a series of 74S37 NAND gates. These gates, shown in the print set as negative input OR gates, have inputs from the microbranch multiplexer [U BRANCH (7:0)] and the basic microaddress register [BUA (8:0)].

*The only sequence is the one by which the code was written. For example, the code may sequence from 702 to 0 to 700 only because those were the words assigned by the U assembler.

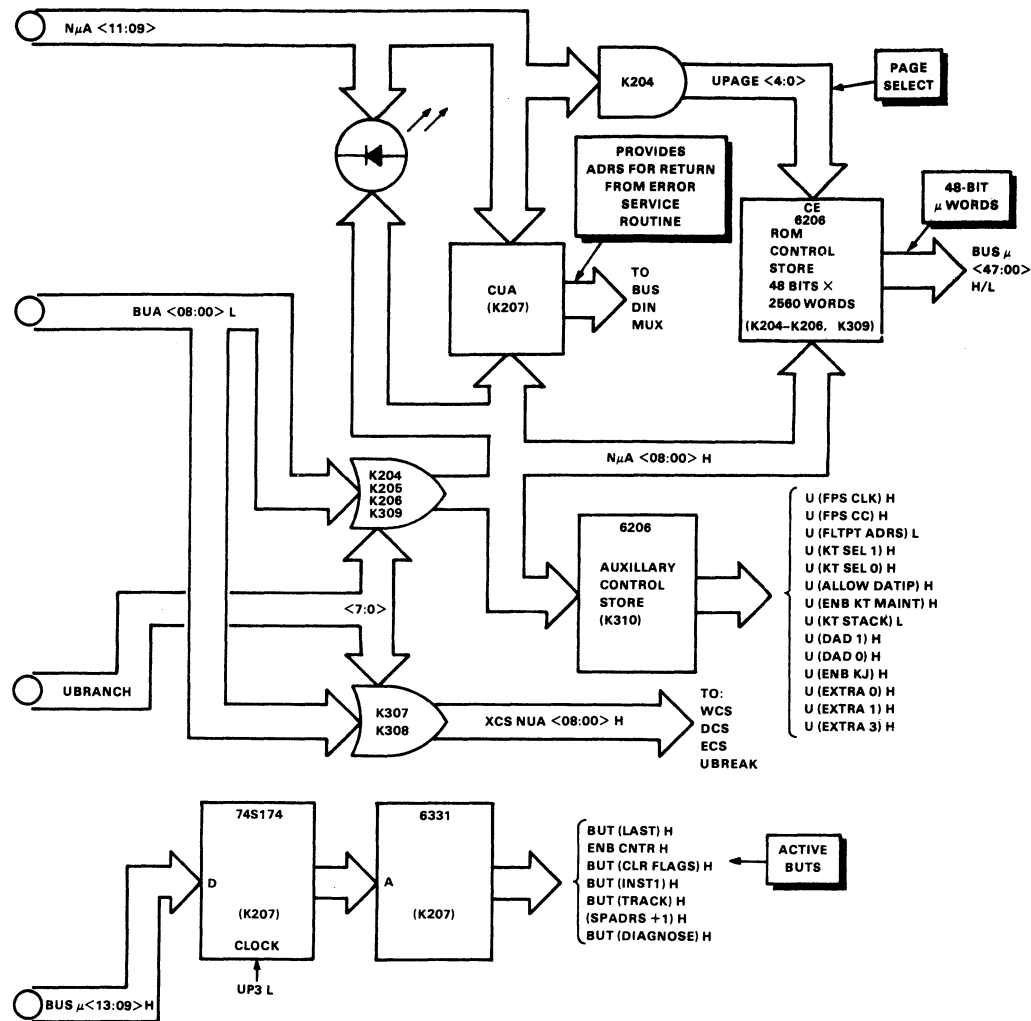
Table 4-2 Microword Bits

BUS UXXH/L	Source (Print Set)	Destination (Print Set)	Field(s)
0	K206	K203	UBF
1	K206	K203	
2	K206	K203	
3	K206	K203	
4	K206	K203	
5	K206	K203	
6	K206	K203	
7	K206	K203	
8	K205	K203	UPF
9	K206	K203, K207, K307	
10	K206	K203, K207, K307	
11	K205	K203, K207, K307	
12	K205	K203, K207, K307	
13	K205	K203, K207, K307	
14	K206	K405	
15	K206	K405	Scratchpad Rewrite
16	K205	K405	
17	K205	K405	
18	K205	K405	
19	K309	K311	BUS/BOX Control
20	K205	K203, K209, K407, K409	
21	K205	K209, K407, K409	
22	K205	K207, K407, K409	
23	K205	K207, K209, K404, K407	
24	K205	K207, K209, K405	
25	K309	K311	
26	K309	K311, K409	Clocks
27	K309	K404	
28	K309	K404	
29	K309	K409	
30	K204	K203, K209, K404	COUT, Page
31	K204	K203, K209, K404	
32	K204	K203, K209, K404	
33	K204	K203, K209, K406, K407	
34	K204	K203, K209, K406, K407	RIF
35	K204	K203, K209, K406, K407	ASEL
36	K204	K203, K209, K406, K407, K409	
37	K204	K203, K209, K311, K406, K407	AEN
38	K204	K203, K209, K405	
39	K204	K203, K209, K405	BSEL
40	K204	K203, K409, K407	
41	K204	K203, K209, K407	BEN
42	K309	K209, K405, K407	
43	K309	K209, K313, K405, K407	ALU, RETURN PAGE, EMIT
44	K309	K203, K209, K313	
45	K309	K203, K209, K313	
46	K309	K203, K209, K313	
47	K309	K209, K313	



11-5538

Figure 4-2 ROM Control Store Addressing (Sequencing) (Sheet 1 of 2)



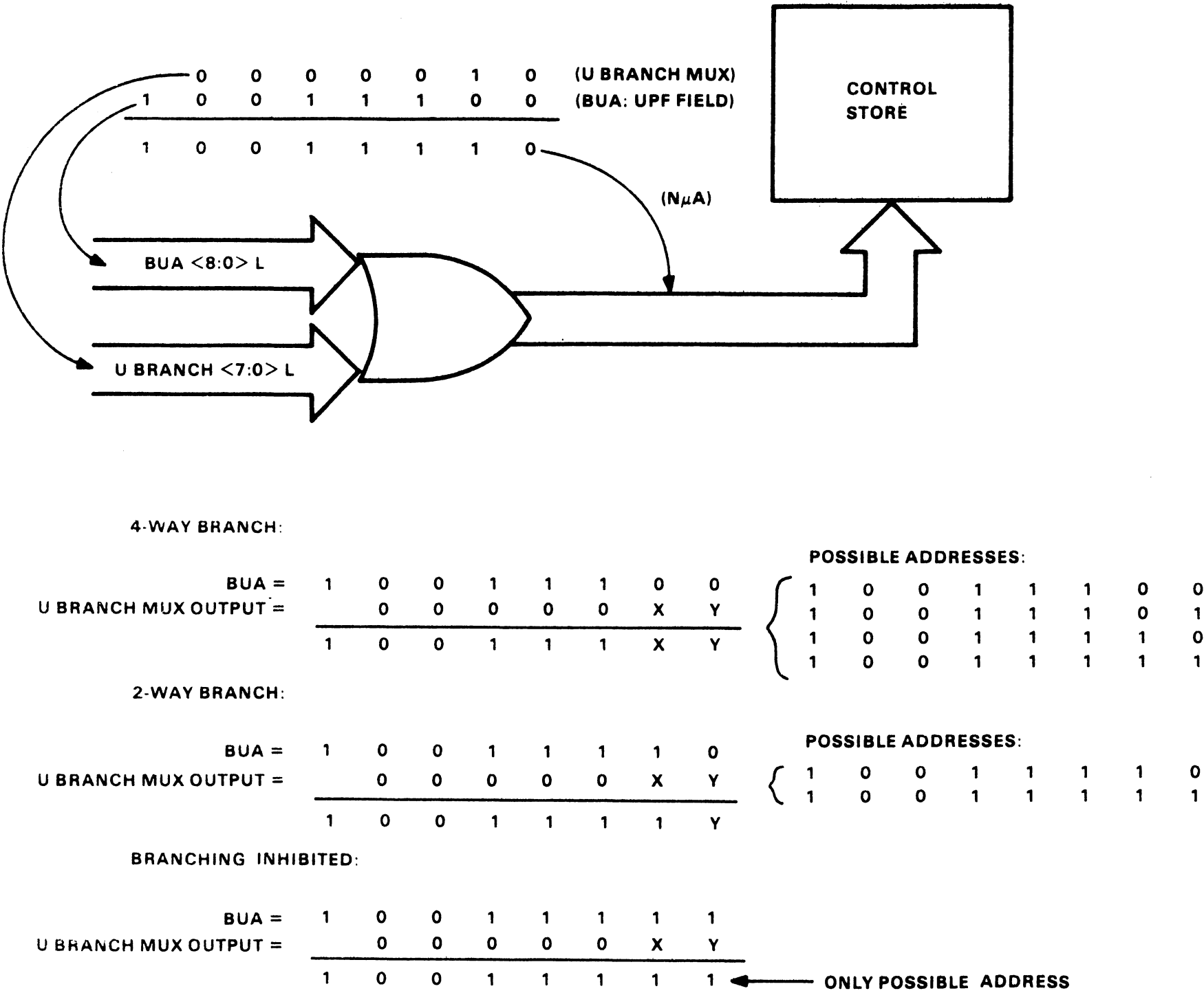
ROM CONTROL STORE ADDRESSES

PAGE NUMBER	$N_{\mu A}$	$BUA <8:0> / UBRANCH <7:0>$	2560 LOCATIONS
	11 10 9	8 7 6 5 4 3 2 1 0	
0	{ 0 0 0 0 0 0 0 0 1	{ 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0	} 512
1	{ 0 0 1 0 0 1 0 1 0	{ 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1	} 512
2	{ 0 1 0 0 1 0 0 1 1	{ 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0	} 512
3	{ 0 1 1 0 1 1 1 0 0	{ 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1	} 512
4	{ 1 0 0 1 0 0 1 0 0	{ 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	} 512

813

Figure 4-2 ROM Control Store Addressing (Sequencing)
(Sheet 2 of 2)

As previously noted, two microword fields, microbranch and micropointer, determine the NUA. The micropointer field (in the BUA register) contains the primary address of the next microinstruction. This address can be modified by any one of several conditional branches. The modification, which takes place in the OR gates described in the previous paragraph, is controlled indirectly by the UBF. Three of the UBF bits [BUS U(11:09)], clocked into the microbranch selection register, select one of the eight microbranch multiplexer ports. The multiplexer data input, e.g., a machine condition or decoded IR bits, is ORed with the UPF field bits [BUA (8:0)] to produce the low-order control store address bits. Figure 4-3 shows an example of this. The ORed result may or may not be the same as the primary address in the BUA field.



11-5546

Figure 4-3 Microaddress Modification

The BUA field configuration determines the number of possible branch addresses. If the UPF field bits to be ORed are all equal to 0, then the microbranch multiplexer output determines the address. However, the number of possible branch addresses is reduced if one or more of the BUA bits equals 1. When a BUA bit equals 1, the corresponding NUA bit is always equal to 1, regardless of the microbranch multiplexer output. Thus, a 4-way branch is converted to a 2-way branch when one of the two BUA bits is changed to a 1. Reducing the number of possible branch addresses may be done for several reasons, e.g., to prevent a machine condition from affecting the branch, or to decrease the range of the branch.

4.5 THE MICROBRANCH MULTIPLEXER

The microbranch multiplexer allows the NUA to be modified by certain machine and program conditions. Shown in abbreviated form on Figure 4-1, the microbranch multiplexer consists of ten, 8-to-1, type 74S251 and 74S151, and one, 2-to-1, type 74S158 multiplexer chips. Input selection for each chip is determined by UB SEL (2:0), which is developed from the UBF field bits BUS U(11:09) H in the microbranch selection logic. Note that both the asserted high and asserted low UB SEL (2:0) signals are used. The strobe input to each chip (in effect the CE input) is one of the UB HI SEL signals from another part of microbranch selection. These strobe signals are derived from BUS U(13:12) of the UBF field. Thus, the entire 5-bit UBF field is used to determine which one of the microbranch multiplexer inputs is ORed with the UPF field bits.

The microbranch multiplexer outputs, K307/K308 U BRANCH (07:00), are ORed with the UPF multiplexer outputs, K203 BUA (08:00), to produce the NUA for circuits in processor control and for the writable control store in the WCS module. Two groups of OR gates (actually 74S37 and 74S00 NAND gates) are used. One set (the 74S37s) is shown on K204, K205, K206; K309 provides the NUA for the control store and its 8-bit extension, the auxiliary control store (Paragraph 4.3), the current microaddress (CUA) register (Paragraph 4.14), and a row of LEDs on the microword module. The NUA output from the other set of OR gates (the 74S00s), shown on K307 and K308, goes to the writable control store.

Branch Microprogram Tests (BUTs)

As described in Paragraph 4.4, the UBF field, BUS U(13:09), is decoded to select conditional branches when they are required in microprogram sequencing. These UBF codes are defined as conditional branch BUTs. Another class of BUTs, active BUTs, results in some action other than control store address modification.

Table 4-3 lists all 30 conditional branch and active BUTs (BUT 30 is NULL and BUT 36 is not used at present). The BUT number corresponds to the UBF field configuration, while the microbranch multiplexer inputs are identified in the composition column. The BUT width indicates the number of bits that are ORed with the UPF field to establish the NUA. A width of 4, for example, would allow up to 16 possible branch addresses. The other table headings are self-explanatory.

Table 4-4 lists the active BUTs that are produced by the active BUT ROM shown on drawing K207.

Two examples of conditional branch BUTs are described in the following paragraphs: BUT (CASE) and BUT (COUNT). Additional information on BUTs can be found in the *PDP-11/60 Microprogrammer's Guide*.

4.5.1 BUT (CASE)

The CASE branch, BUT (CASE), is asserted when the UBF field, BUS U(13:09) = 0. With a 4-bit width, BUT (CASE) provides a choice of up to 16 possible control store addresses (NUA) (Figure 4-4). When a BUT (CASE) is decoded in the UBF field, the four low-order bits of the UPF field, BUA (3:0), are ORed with the four low-order bits from the shift register, K408 SR (3:0). (These SR bits constitute the microbranch multiplexer output at this time.) The result of this ORing is NUA (3:0). The high-order address bits, NUA (8:4), are equal to the state of BUA (8:4) L, the output of the BUA register. (Only the low-order address bits are traced in Figure 4-4.)

Table 4-3 Branch Microprogram Tests (BUTs)

Number	Composition	Width	Use or Action	Name and Synonyms
BUT00	SR(03:00)	4		CASE
BUT01	IR(15:12)	4		DOP
BUT02	INSTR 5 BR(4:0)	5		INSTR 5
BUT03	IR11 □ FLTPT BR(3:0)	5		FLPDECODE
BUT04	IR(09:06)	4		SOP
BUT05	MOV □ IR(05:03)	4		MOV □ DM, DM
BUT06	INSTR 1 BR(7:0)	8		INSTR 1
BUT07	SERVICE □ FLIPT RETURN(2:0)	4		FRET
BUT10	COUT07 □ DOUT 07 □ FPS05	3		FNORM
BUT11	DM0 □ SM0 □ BYTE	3		DM0 □ SM0 □ BYTE
BUT12	GUARD(3:2)	2		GUARD
BUT13	SR(01:00) □ [COUNT(7:0) ALL ONES]	3		MULSTEP
BUT14	BG INTERNAL □ MF INSTR □ MULTI BR	3		X SERVICE □ MFINC □ MULTIPLE
BUT15	[D(14:00) ALL ZEROS] □ DOUT 15	2		D(14:00) = 0 □ D15, D IS 0
BUT16	IR11 □ PS15	2		IR11 □ PS15
BUT17	[COUNT(7:0) ALL ONES] □ D(C)	2		COUNT IS -1 □ D(C), ASHBR
BUT20	NO INSTR OVLAP □ SERVICE	2		FOV □ SERVICE
BUT21	VECTOR □ 0	2		PSSYNINTERNAL
BUT22	SF ORed WITH ONE	Active	Low bit of IR source field ORed with one	(SPADRS+1)
BUT23	D(C) □ BA00	2		D(C) □ ODD ADDRESS
BUT24	JAMUPP-NOT INT ADRS) □ FLTPT FLAG	2		OTHER JAMUPP □ HOTWARM
BUT25	COUNT(7:0) ALL ONES	1		COUNT IS -1
BUT26	INTR REQUEST □ NO BR INSTR	2		INTR REQUEST □ SUCBRANCH
BUT27	FOVLAP SAVE □ FPS07	2		FOVPSAV □ FPS07, FD
BUT30	NULL	-		NULL
BUT31	UADRS TRACK	Active	Clock for latching CUA ENABLE, Disabled upon JAMUPP	TRACK
BUT32	CLR FLAGS	Active	Clear short term flags (MFPI, MTPI, T BIT MASK) Reset RES register	Clear flags
BUT33	DIAGNOSE	Active	Future use by ECS (Microdiagnostics)	
BUT34	SUBR B	Active	RETURN(11:00)←EMIT(14:03), PAGE(2:0)←EMIT(02:00)	SUBR B, GO TO
BUT35	SUBR A	Active	RETURN(11:00)←DOUT(14:03), PAGE(2:0)←EMIT(02:00)	SUBR A
BUT36	RESERVED	Active	Action undefined	
BUT37	RETURN	Active	PAGE(02:00)←RETURN(11:09), BUA(8:0)←RETURN(08:00) RETURN(11:00)←DOUT(14:03)	RETURN

- Notes:
1. Conditions are active high.
 2. All branch widths are justified to the low order U ADRS BITS.
 3. Active BUTs have no width, base microaddress unmodified.
 4. Connections desired
 1. MF INSTR goes to BUT21.
 2. MULTI BR used alone.

Table 4-4 Active BUT ROM (K207)

Signal Asserted	ROM Output Pin	Input					Output							
		BUS UXX H												
		13	12	11	10	09	M6	M5	M4	M3	M2	M1	M0	
K207 BUT (LAST) H	1	H	L	L	L	L	0	1	0	0	0	0	1	
K207 ENB CNTR H	2	L L H H H L L H H	H H L H H L H H H	L H H H H L H H H	L H L H L H H H H	H H H	0	1	0	0	0	1	0	
K207 BUT (CLR FLAGS) H	3	H	H	L	H	L	0	1	0	0	1	0	0	
K207 BUT (INST 1) H	4	L	L	H	H	L	0	1	0	1	0	0	0	
K207 BUT (TRACK) H	5	H	H	L	L	H	0	1	1	0	0	0	0	
K207 (SPADRS+1) L	6	H	L	L	H	L	0	0	0	0	0	0	0	
K207 BUT (DIAGNOSE) H	7	H	H	L	H	H	1	1	0	0	0	0	0	

Table 4-4 Active BUT ROM (K207) (Cont)

Signal Asserted	Destination and Function	
K207 BUT (LAST) H	{ K409: Clear counter register K606: Clear DATIP BUS BUSY flip-flop	
K207 ENB CNTR H	K409: Enable counter register	
K207 BUT (CLR FLAGS)	Asserted during Macro instruction retrieval (FET02)	{ K408: Clear RES CNTL register K210: Clear short term flags K209: K210 CLR UCON L clears UCON REG; BUS DIN ← EMIT (UCON default)
K207 BUT (INST 1) H	Asserted during Macro instruction retrieval (FET03)	{ K306: Inhibit A and B SPLO (Words 20-37 ₈) K706 LED (RUN) H: Indicate at console that machine is in RUN state
K207 BUT (TRACK) H	K207: Allow tracking of U code flow (NUA) by CUA register	
K207 (SPADES+1) L	K406: Address upper (ODD) half of a 32-bit register that consists of two 16-bit SP registers	
K207 BUT (DIAGNOSE)	K101: Transfers control from DCS to execute selected areas of the base machine U code. Then returns control to DCS.	

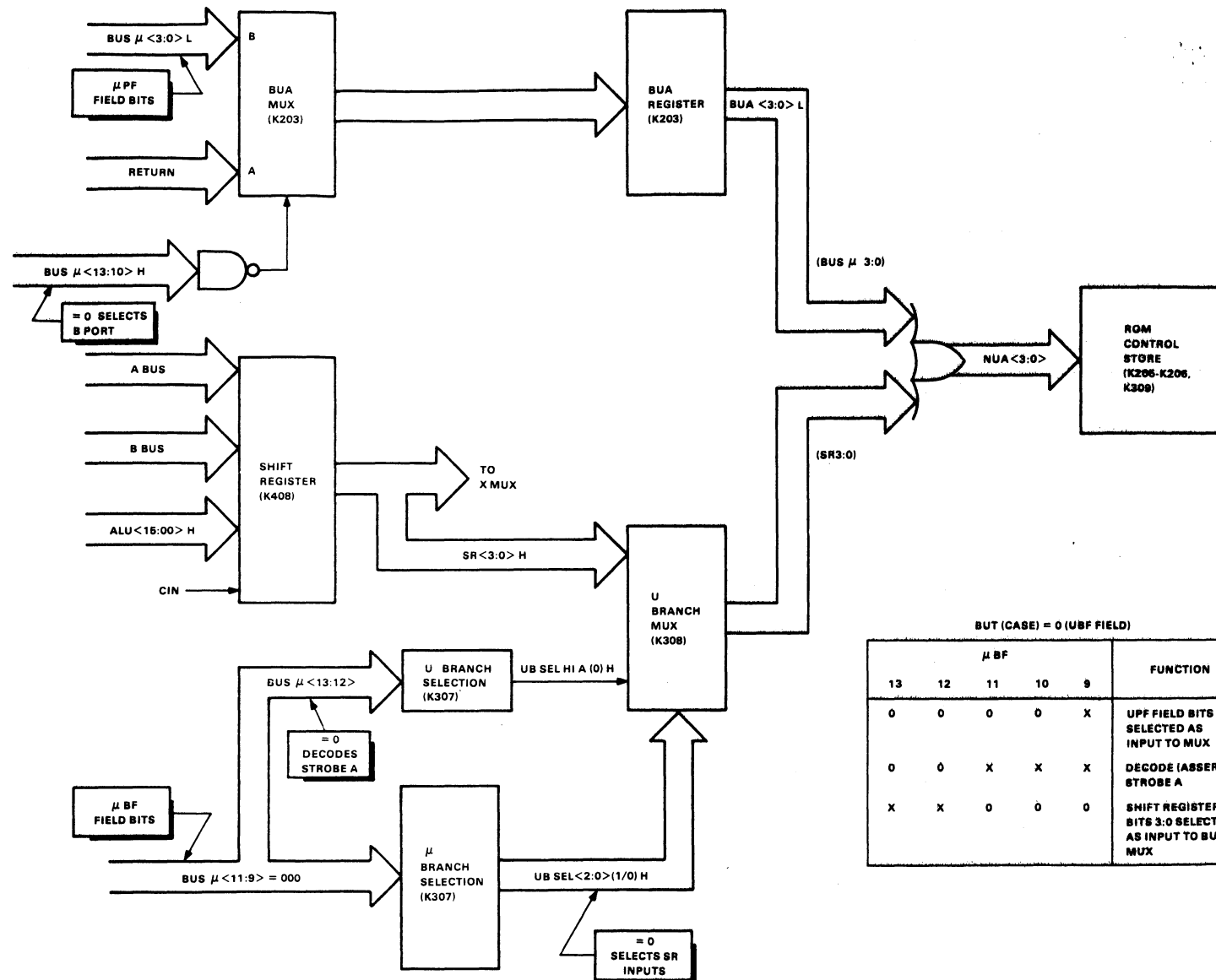


Figure 4-4 ROM Control Store Addressing
[Example: BUT (CASE)]

If the primary address from the BUA register has BUA (3:0) = 0, then there is a full 16-way branch capability. Consider, for example, a primary address of 420₈:

BUA	=	100010000	16 Possible
Microbranch MUX Output	=	<u>0000WYYZ</u>	Addresses:
		10001WXYZ	100010000
			through
			100011111

Consequently, with this configuration, the NUA will be one of the 16 addresses in the range 420₈ through 437₈. The number of possible branch addresses is reduced if BUA (3:0) is not equal to 0 (Figure 4-3).

The table in Figure 4-4 illustrates the different functions performed by the UBF field bits when a BUT (CASE) is decoded. Bits (13:10) = 0 select the B port data of the BUA multiplexers, BUS U(3:0). These are the four low-order bits of the UPF field. Bits (13:12) = 0 cause the assertion of K307 UB M1 SEL 0 (0) H, the strobe input to four low-order microbranch multiplexer chips. Finally, bits (11:9) = 0 select the shift register inputs to the microbranch multiplexer, K408 SR (3:0).

4.5.2 BUT (COUNT)

Some of the conditional branch BUTs perform multiple functions; they cause some action(s) in addition to the ORing of the BUA and microbranch bits to generate the NUA. One of these multiple action BUTs is BUT25, BUT (COUNT). Two operations occur when BUT (COUNT) is decoded in the UBF field:

1. The count in the counter register (Paragraph 2.5.1.1) is checked and then incremented.
2. If, when checked, the count equals 377₈, the counter is cleared and the signal CNT = ONES is asserted. The low-order bit from the microbranch multiplexer (the state of CNT = ONES at this time) is ORed with BUA 00 L to provide the LSB of the NUA. The logic involved is shown in Figure 4-5.

The purpose of BUT (COUNT) is to control when a program should branch after a determined number of loops through a routine. (Another use is to keep track of the number of loops necessary to accomplish some desired result.) If the count is less than the prescribed number (count = 377₈), the loop is repeated (NUA 00 H = 0). Once the desired count is reached (count = 377₈), K311 CNTR = ONES (1) H causes NUA 00 H to equal 1 and an exit from the loop is made.

The timing for this is shown in Figure 4-6. First, the 2's complement (374₈) of the desired count is placed on the B bus and the count is loaded (U14 and U16 asserted) at P3 time. Then, at UP3 time, the count is tested, and since it is not equal to 377₈ (it was previously cleared) CNT = ONES H is low and, therefore, NUA 00 H = 0. Since NUA 00 = 0, a branch address is not generated to allow an exit; the loop will be performed. At P2 time, the counter clock input goes low and the count, 374₈ in this case, is present at the output. (With the load input signal asserted, the counter does not have to be enabled for the count to be present at the output.)

Except for the load operation, the process is repeated each time BUT (COUNT) is encountered in a microinstruction. The enable signal, K203 ENB CNTR H, is asserted, the count is tested and then it is incremented. The counter is tested the fourth time (at UP3) and the resultant NUA 00 H = 0 allows the fourth loop to take place. The counter is incremented at P2 and at UP3 the carry, K311 CNTR = ONES (1) H is asserted. The modified NUA allows an exit to a new address outside of the loop.

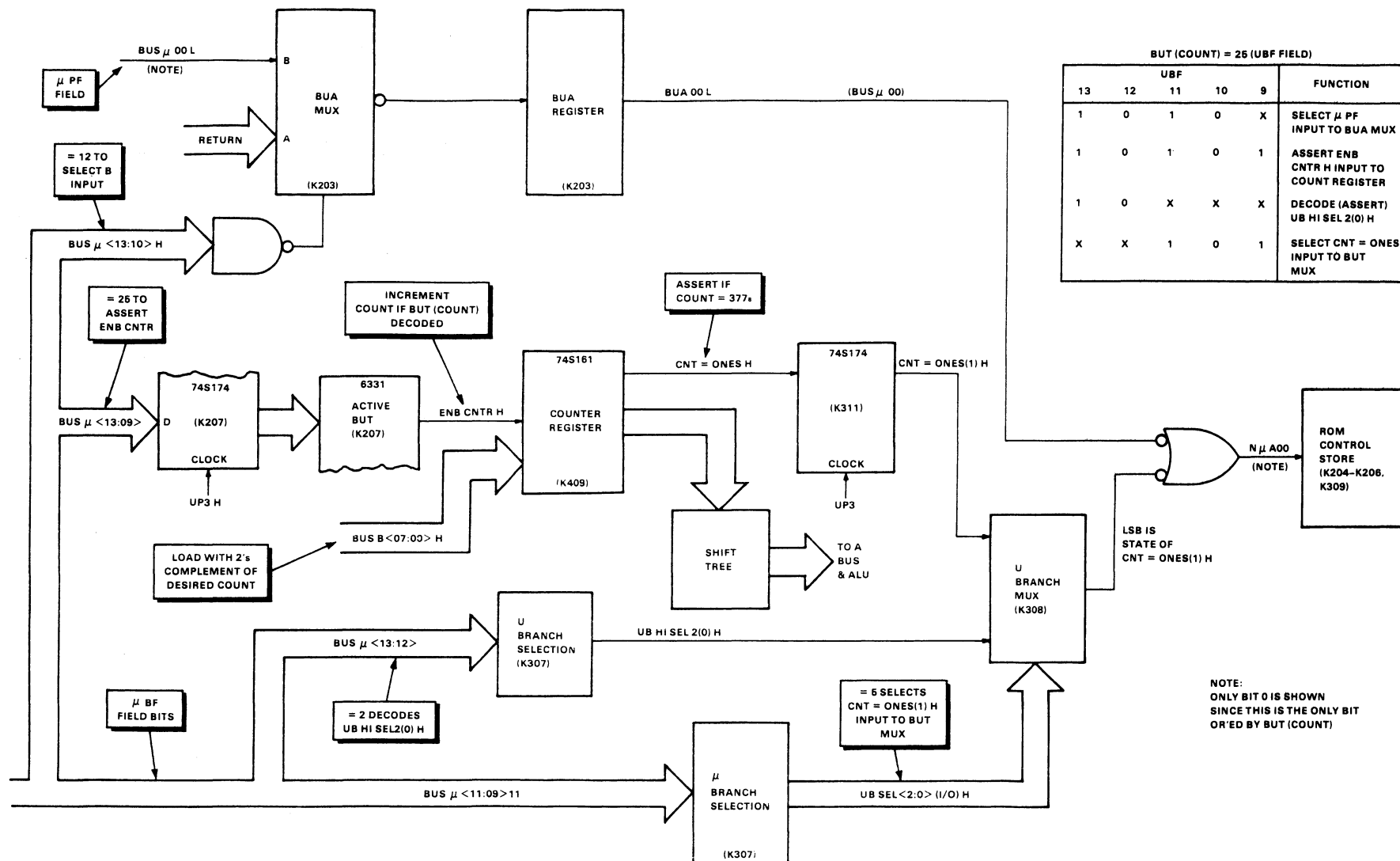
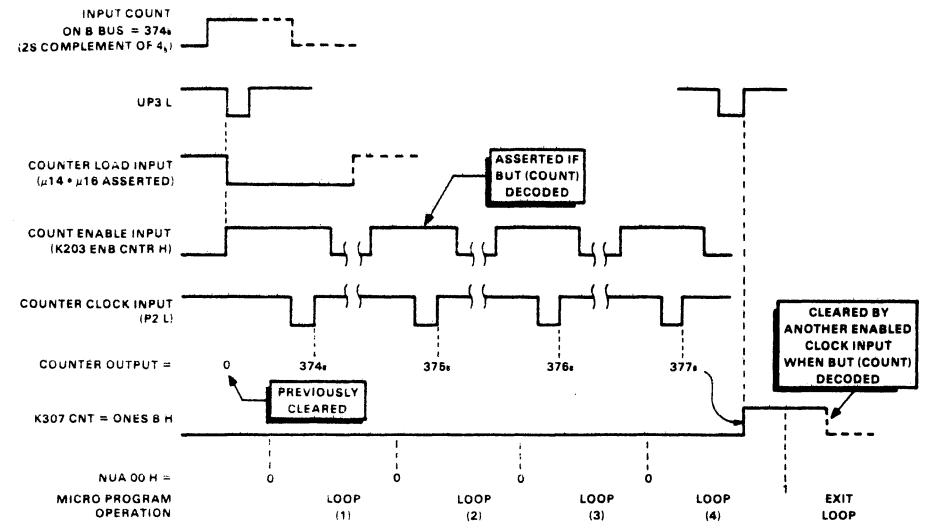


Figure 4-5 ROM Control Store Addressing
[Example: BUT (COUNT)]



11-5549

Figure 4-6 Counter Register Timing [Example: BUT (COUNT) Used to Count Four Passes Through a Loop]

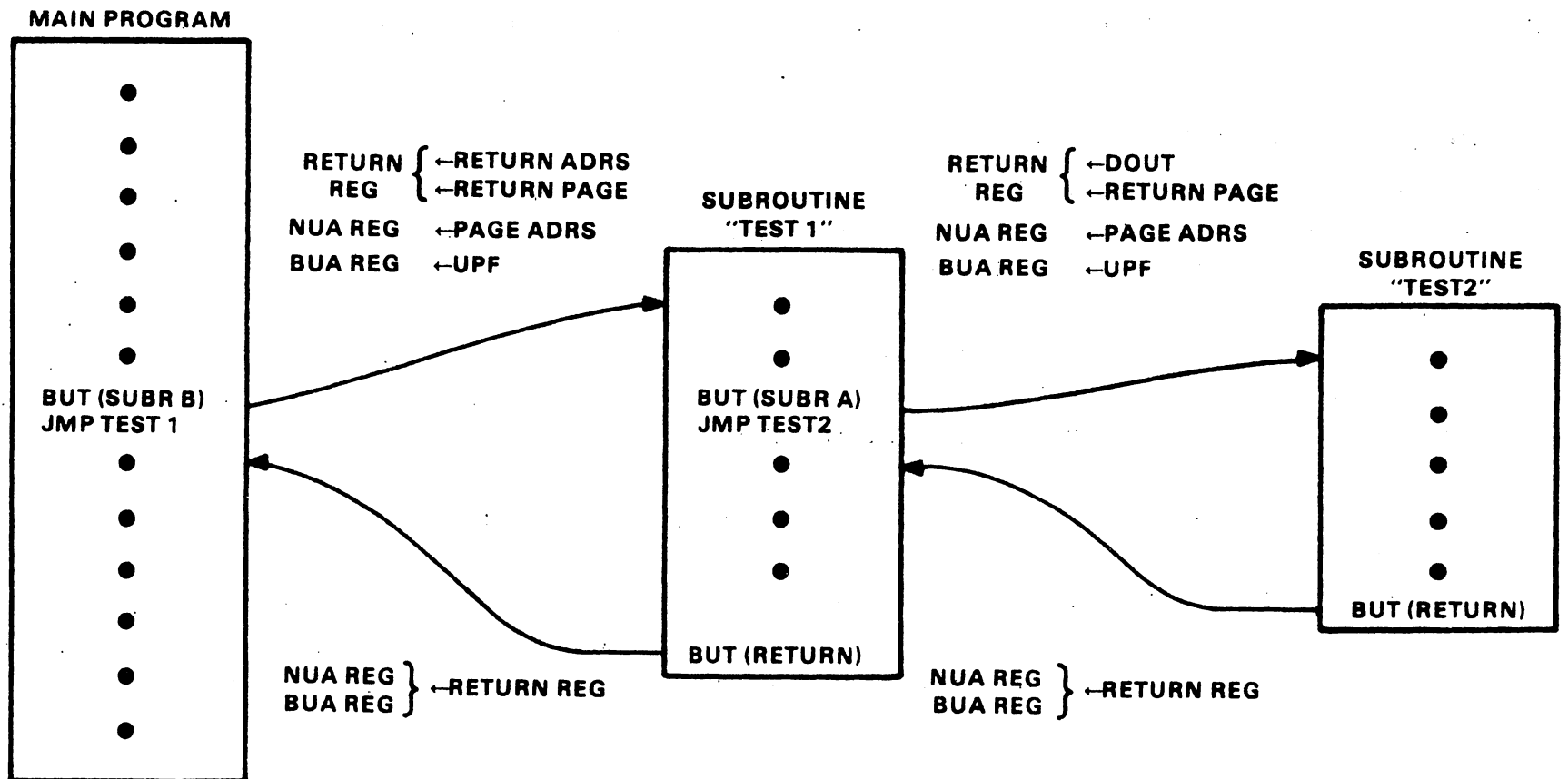
4.5.3 NUA and Return Registers

The NUA page register (Figure 2-2) provides the 3-bit page address [NUA (11:09)] to select one of the five, 512 word, pages in the control store (Paragraphs 4.3 and 4.4). The register consists of a single 74S194 shift register chip. Input data is obtained from the NUA multiplexer, a 2-to-1, 74S157 multiplexer chip. A high at the register mode control input is necessary to load data into the register from the multiplexer. This input is high when a BUT (SUBR A), BUT (SUBR B), or BUT (RETURN) is decoded in the UBF field [BUS U(13:11) all high]. At other times, the register mode control input is low and loading is inhibited; the register output remains unchanged. Consequently, the page address cannot be changed unless one of these three BUTs is decoded.

The return register, and the associated return multiplexer, are similarly constructed. Consisting of three, 74S194, shift register chips, the return register holds the address that the microprogram returns to after the execution of a subroutine. Inputs to the register are from three multiplexer chips (one 74S157 and two 74S158). The return register uses the same mode control input [BUS U(13:11) all high] to load data from the multiplexer. Therefore, the register contents can be changed only if BUT (SUBR A), BUT (SUBR B), or BUT (RETURN) is decoded.

4.5.4 NUA and Return Address Control

The three BUTs mentioned in the previous paragraph [BUT (SUBR A), BUT (SUBR B), and BUT (RETURN)] control address source selection when the microprogram jumps to, and later returns from, a subroutine. When a subroutine is to be executed, the starting address of the subroutine must be made available to the control store. At the same time, the program address where the operation will return to must be retained for use after the subroutine is completed. Figure 4-7 depicts how this is accomplished.



11-5550

Figure 4-7 Return Address Control (Simplified)

BUT (SUBR B) stores the return address in the return register via the EMIT field when a jump to a subroutine is called for in the main program. The return address consists of the contents of the return address and return page fields in the same microword in which BUT (SUBR B) is decoded in the UBF field. On completion of the subroutine, BUT (RETURN) causes the transfer of the address in the return register to the NUA register [to provide NUA (11:09)] and to the BUA register [to provide BUA (08:00)].

BUT (SUBR A) functions in a similar manner. However, instead of causing the transfer of the EMIT field to the return register, BUT (SUBR A) transfers DOUT (14:03) into the same register. Therefore, the return address can be obtained from data calculated or stored in the data path (D register). This allows BUT (SUBR A) to be used for nested subroutines (Figure 4-7) when return addresses must be saved in a stack. [BUT (SUBR A) can also be used in subroutine applications in place of BUT (SUBR B)].

BUT (RETURN) must be included at the end of a subroutine to cause the transfer of the return address to the NUA and BUA registers (where it is available to access the control store). However, BUT (RETURN) is not necessary if BUT (SUBR A) or BUT (SUBR B) is used in either the main program or a subroutine simply to change the page address and not in connection with a jump to a subroutine.

4.6 INSTRUCTION DECODING

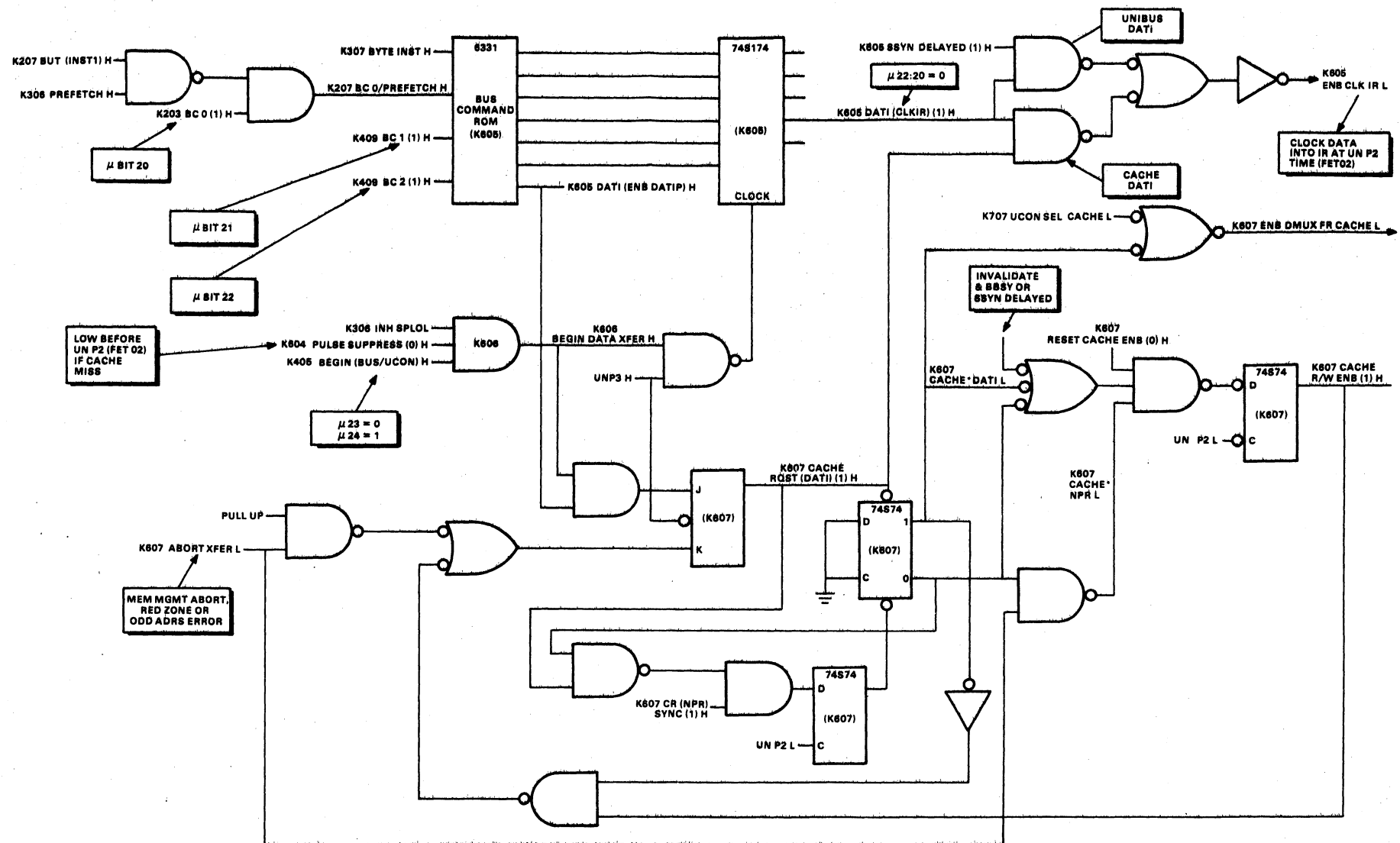
Macroinstructions from cache memory and the backing store are decoded in the processor control decoding logic on the M7873 module. The logic consists of a network of ROMs and gates that are supplied with all or part of the macroinstruction word held in the instruction register (IR). The decoded output from this network is delivered to different areas of the machine. For example, the microbranch multiplexer uses information from four ROMs in the decoding logic in the development of the NUA. Figure 4-8 shows the major components used to decode instructions and their relationship to the IR, microbranch multiplexer, and other circuits.

Instructions are input to the IR from the data multiplexer [K515 DMUX (15:00) H] on the cache module. The DMUX, which is shown on Figures 2-9 and 2-10, receives 16-bit words (both data words and instruction words) from the main memory (backing store) on the Unibus or from cache memory if the data at the accessed cache location is valid. The CSP on the data path module also uses the DMUX as the source for data: data words (and some instruction words) are sent to the MD register in the CSP (Figure 3-10).

4.6.1 Instruction Register

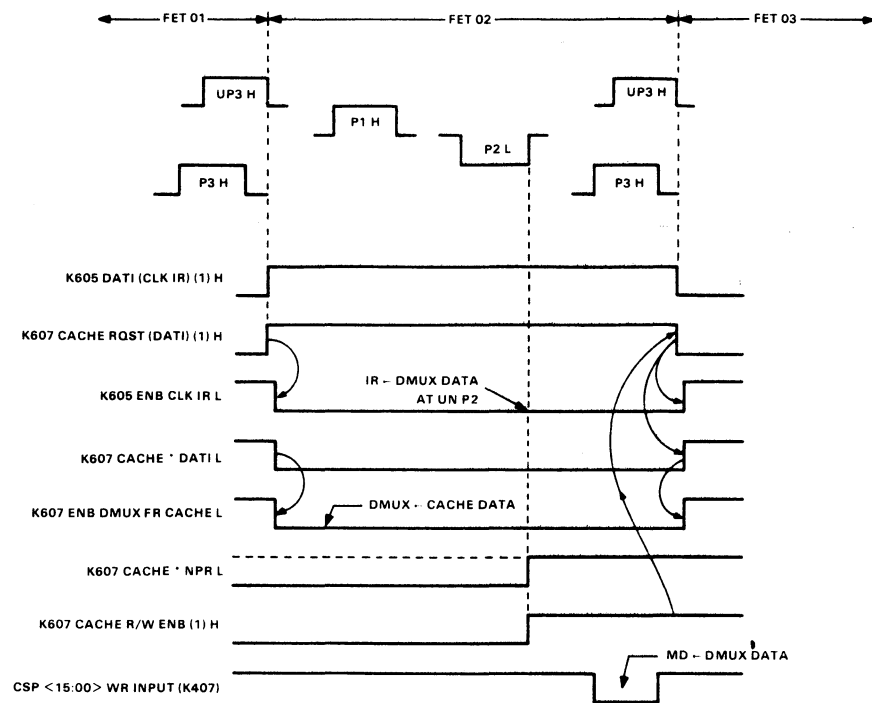
The IR consists of four 74S175, quad, D-type flip-flop chips shown on drawing K303. Macroinstructions are loaded (at P2 time) into the register by either one of two signals: K209 UCON ENB IR L, which allows the IR to be loaded under UCON control, or K605 ENB CLK IR L, which is developed from bus command ROM signals. K209 UCON ENB IR L is asserted if certain UCON signals are high: K209 UCON PROC (1) H (microword bit 36); K209 UCON 15 (1) H (microword bit 32); and microword bits 21, 23, and 24 [these three bits are asserted as K209 UCON XFER (1) H]. The IR is loaded under UCON control during the trap routine to allow yellow or red zone errors to occur (red/yellow zone errors can occur only if the destination register is R6). Other applications are during the implementation of the MFPI, MFTI, and MARK instructions.

The other gating signal for the IR clock input, K605 ENB CLK IR L, is asserted by K605 DATI (CLK IR) (1) H and K605 SSYN DELAYED (1) H for a Unibus (i.e., main memory) DATI if a cache miss occurs, or by K605 DATI (CLK IR) (1) H and K607 CACHE RQST (DATI) (1) H for a successful DATI to cache memory (Figure 4-9). K605 DATI (CLK IR) (1) H, necessary to assert the IR clock during both Unibus and cache DATI operations, results when microword bits (22:20) = 0 are the input to the bus command ROM. Timing for the assertion of the IR clock signal for a cache DATI hit is shown in Figure 4-10. Figure 4-11 shows the timing when a cache DATI miss results in a Unibus DATI cycle.



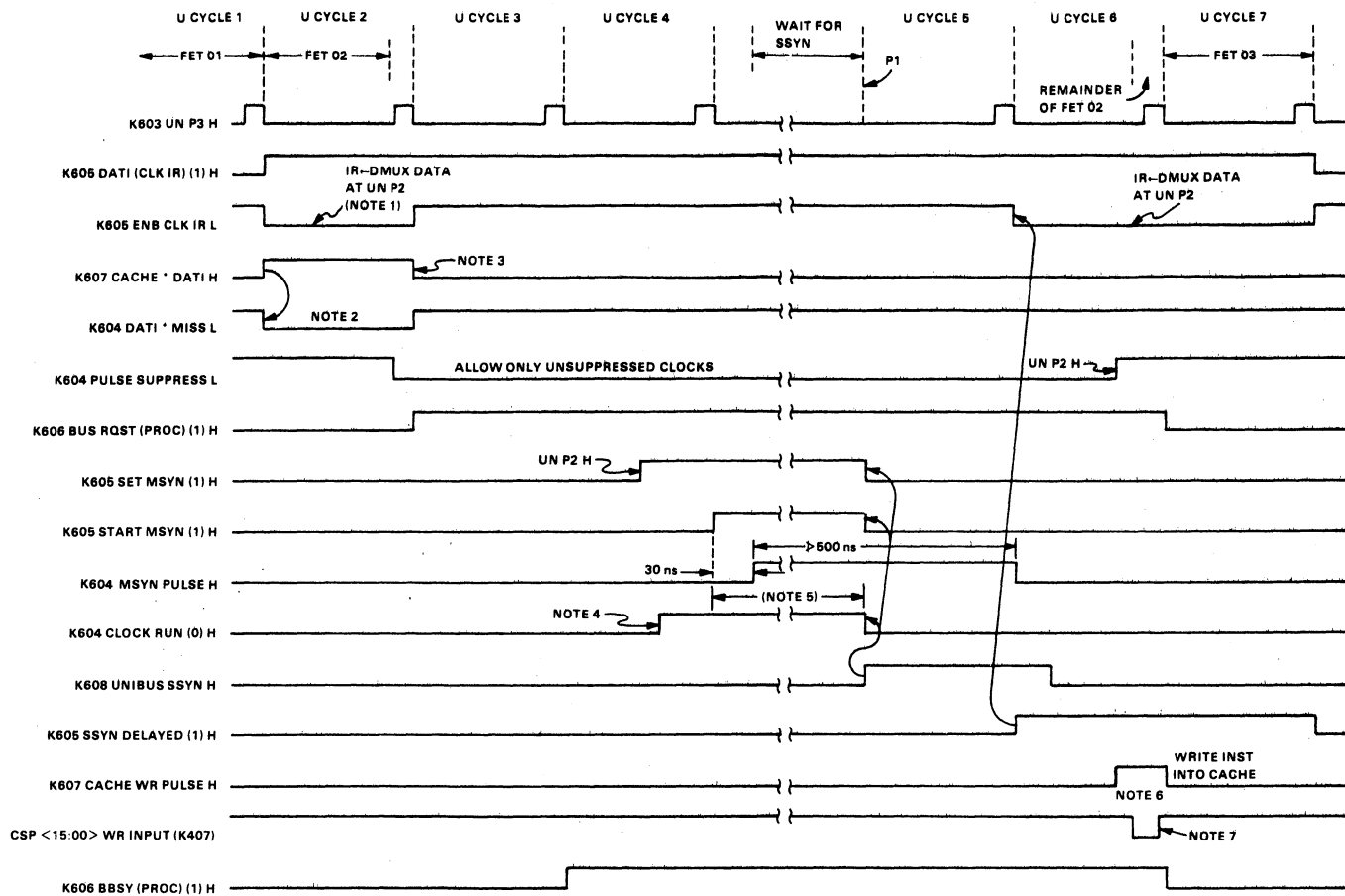
853

Figure 4-9 IR Clock Enable (Cache DATI)



11 5584

Figure 4-10 IR Clock Enable Timing (Cache DATI Hit)



NOTES:

- | | | |
|---|---|--|
| 1. IR CLOCKED TWICE IF CACHE MISS | 4. K605 SET MSYN (1) H • K603 CLOCK P2 L = K604 STOP (1) H WHICH CLOCKS THE CLOCK RUN F/F | 6. ASSERTED BY K607 CACHE R/W ENB (1) H AT UN P2 TIME; NEGATED BY K607 RESET CACHE ENB (0) H AT UN P3 TIME |
| 2. INHIBIT K603 P3 H DURING U CYCLE 2 IF CACHE MISS | 5. STOP CLOCKS AFTER P3 (U CYCLE 4). RESTART CLOCKS WITH P1 (U CYCLE 5) | 7. MD - DMUX DATA AT P3 TIME |
| 3. K607 CACHE * DATI L NEGATES K607 ENB DMUX FR CACHE L; THE DMUX BUS DIN (UNIBUS) PORT IS SELECTED | | |

11-5585

Figure 4-11 IR Clock Timing
(Cache DATI Miss - Unibus Cycle)

4.6.2 Preliminary IR Decoding (K304)

Initial decoding of the macroinstruction held in the IR takes place in the preliminary IR decode logic shown on drawing K304 and Figure 4-8. Consisting of two 74S138, 3-to-8 line decoder chips, plus associated AND and OR gates, this circuit decodes IR bits (15:03) to produce the signals used as inputs to the next level of decoding. Table 4-5 lists the instruction types and classes decoded from the input, K303 IR (15:03) (1) H.

Six bits from the IR, K303 IR (14:09) (1) H, are decoded by one of the 74S138 chips (E115) and two 74S133 NAND gates to assert several basic signals, e.g., K304 MOV H, K304 DOP H, and K304 SOP H, that identify the instruction type. Combinations of these basic signals produce additional signals that define the instruction class. There are seven classes of instructions, A through G. For example, an A class instruction (K304 A CLASS INSTR L) can be the XFC instruction, a floating point instruction (K304 FLTPT INSTR L), or any double operand, register to register instruction (both source and destination fields specifying mode 0). The other 74S138 chip (E104), which is enabled by IR (14:12) = 0, decodes the E (jump) and F (branch) class instructions from IR (11:09). K304 JSR/TRAP L, another signal decoded in this chip [IR (11:09) = 4], is one of the inputs to the KJ control logic [K312 KJ (1) H].

Figures 4-12 through 4-18 show the logic used to decode each of the seven classes of instructions. Definitions of the microbranch condition bit positions for each class are also included.

4.6.3 Microbranch Conditions (K305)

IR bits alone, and in combination with information from the flag register (K210) and the instruction class signals developed in the preliminary decode logic, assert three types of microbranch condition signals: K305 UBC (7:0) (INSTR1) H, K305 UBC (4:0) (INSTR5) H, and K3-5 UBC (3:0) (FLTPT) H. These signals are used in the implementation of the NUA (Paragraph 4.4).

4.6.3.1 Instruction 1 Branch Conditions – [K305 UBC (7:0) (INSTR1) H] is one of several groups of signals used in the determination of the NUA, the address of the next microword to be read from the control store ROM. To perform this function, K305 UBC (7:0) (INSTR1) H is selected as the output from the microbranch multiplexer (K307, K308) when microword bits (14:09) = 00110₂ (BUT 06). This occurs during the retrieval and decoding of the current macroinstruction when microinstruction FET 03 asserts K207 BUT (INSTR1) H, i.e., BUT 06.

Examples of how the instruction 1 branch condition signals are developed are shown in Figures 4-12 through 4-18. Refer to the B class instruction diagram, Figure 4-13. The output from the preliminary decode logic, K304 B CLASS INSTR H, is ORed with the C and E class signals to assert K305 B/C/E CLASS INSTR H. This signal, in turn, is ANDed with K303 IR 03 (0) H (the low-order bit of the destination field), which is high for the autoincrement, autodecrement and the index modes (modes 2, 4, and 6). Consequently, if a B class macroinstruction has a destination mode of 2, 4, or 6, K305 UBC 0 (INSTR1) H is always low and the state of K206 NUA 00 H is determined by the state of K203 BUA 0 (1) L. For the other addressing modes, K305 UBC 0 (INSTR1) H is high and, therefore, K206 NUA 00 H is high. (Refer to Paragraph 4.5 for a description of the microbranch multiplexer.

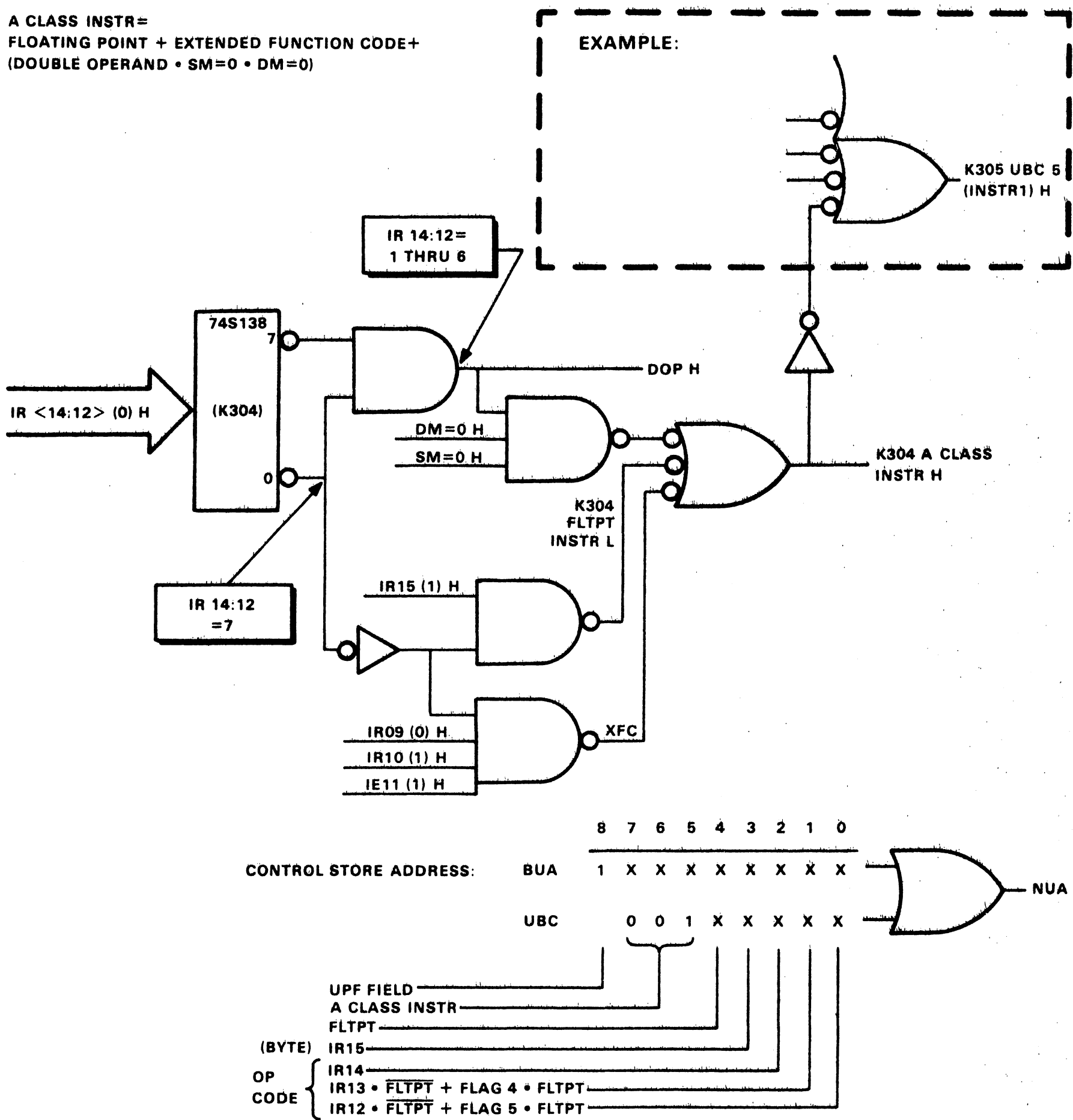
4.6.3.2 Instruction 5 Branch Conditions – Instruction 5 branch conditions [K305 UBC (4:0) (INSTR5) H] are provided by two, 6206 type ROMs that function as an extension of the preliminary decode logic (Figure 4-8). Only one of the ROMs is enabled at one time (the output is tristate); E78 is selected (and E88 is deselected) by K304 IR (15:08) = 0 L. E88 is selected, and E78 is deselected, when IR bits (15:08) ≠ 8. Tables 4-6 and 4-7 list the ROM inputs and outputs for each instruction that is decoded in this circuit. The ROM output, K305 UBC (4:0) (INSTR5) H, is selected as the output from the microbranch multiplexer (K307, K308) when microword bits (14:09) = 00010₂ (BUT 02) assert the proper multiplexer select signals.

Table 4-5 Preliminary IR Decode (K304)

Instruction Types and Classes	Input: K303 IR XX (1) H													Remarks
	15	14	13	12	11	10	09	08	07	06	05	04	03	
MOV	X	0	0	1	X	X	X	X	X	X	X	X	X	MUL, DIV, ASH, ASHC, XOR excluded SWAB and SXT excluded
DOP	X	(1 thru 6 ₈)			X	X	X	X	X	X	X	X	X	
SOP	X	0	0	0	1	0	1	X	X	X	X	X	X	
	X	0	0	0	1	1	0	0	X	X	X	X	X	
FLTPT INSTR	1	1	1	1	X	X	X	X	X	X	X	X	X	
SM=0	X	X	X	X	0	0	0	X	X	X	X	X	X	
DM=0	X	X	X	X	X	X	X	X	X	X	0	0	0	
XFC	X	1	1	1	1	1	0	X	X	X	X	X	X	
A CLASS INSTR	X	1	1	1	1	1	0	X	X	X	X	X	X	
	1	1	1	1	X	X	X	X	X	X	X	X	X	
B CLASS INSTR	X	(1 thru 6 ₈)			0	0	0	X	X	X	0	0	0	XFC FLTPT DOP • SM=0 • DM=0 SOP • DM=0 DOP • MOV • SM=0 • DM=0 MOV • SM=0 • DM=0 SOP • DM=0
	X	0	0	0	1	0	1	X	X	X	0	0	0	
	X	0	0	0	1	1	0	0	X	X	0	0	0	
	X	(2 thru 6 ₈)			0	0	0	X	X	X	0	0	0	
C CLASS INSTR	X	0	0	1	0	0	0	X	X	X	0	0	0	JMP BPL, BMI BR, BMI All branches except BPL, BMI, and BR DOP • SM=0
D CLASS INSTR	X	0	0	0	1	0	1	X	X	X	0	0	0	
	X	0	0	0	1	1	0	0	X	X	0	0	0	
E CLASS INSTR	0	0	0	0	0	0	0	0	0	1	X	X	X	
F CLASS INSTR	1	0	0	0	0	0	0	X	X	X	X	X	X	All branches except BPL, BMI, and BR DOP • SM=0
	X	0	0	0	0	0	0	1	X	X	X	X	X	
	X	0	0	0	(1, 2, or 3 ₈)			X	X	X	X	X	X	
G CLASS INSTR	X	(1 thru 6 ₈)			0	0	0	X	X	X	X	X	X	
JSR/TRAP	X	0	0	0	1	0	0	X	X	X	X	X	X	

Note: Other signals decoded on K304 are K304 IR (14:12)=0 L, K304 IR (15:08)=0 L, and K304 IR (14:09)=0 H.

A CLASS INSTR=
 FLOATING POINT + EXTENDED FUNCTION CODE+
 (DOUBLE OPERAND • SM=0 • DM=0)



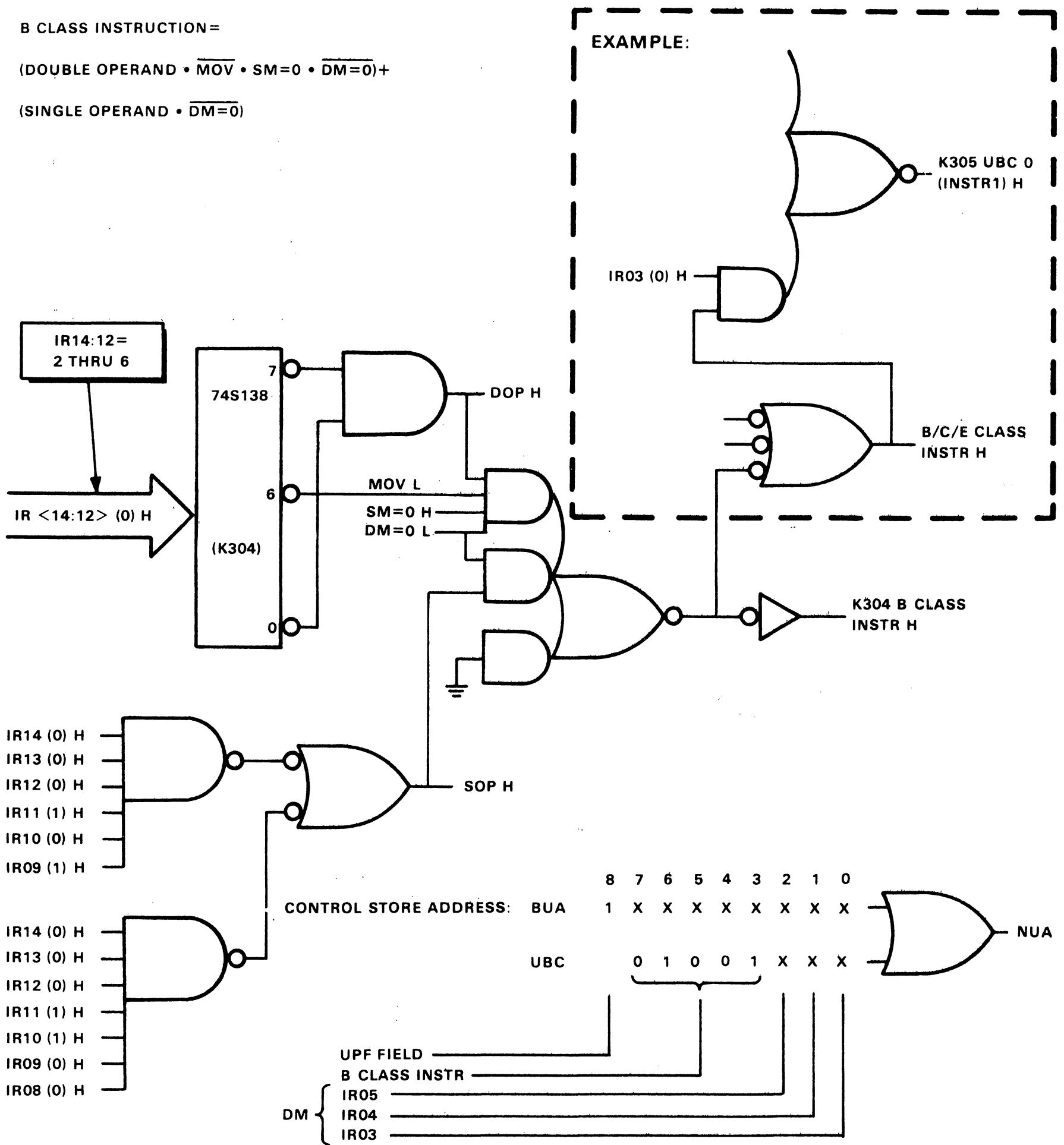
11-5586

Figure 4-12 A Class Instruction Decoding

B CLASS INSTRUCTION=

(DOUBLE OPERAND • $\overline{\text{MOV}}$ • $\overline{\text{SM}}=0$ • $\overline{\text{DM}}=0$) +

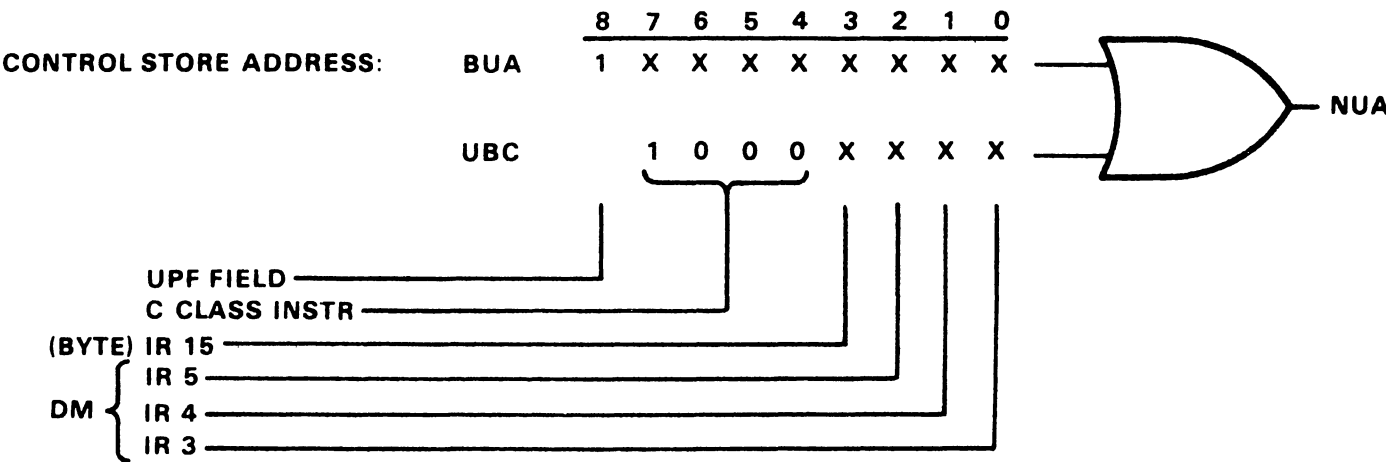
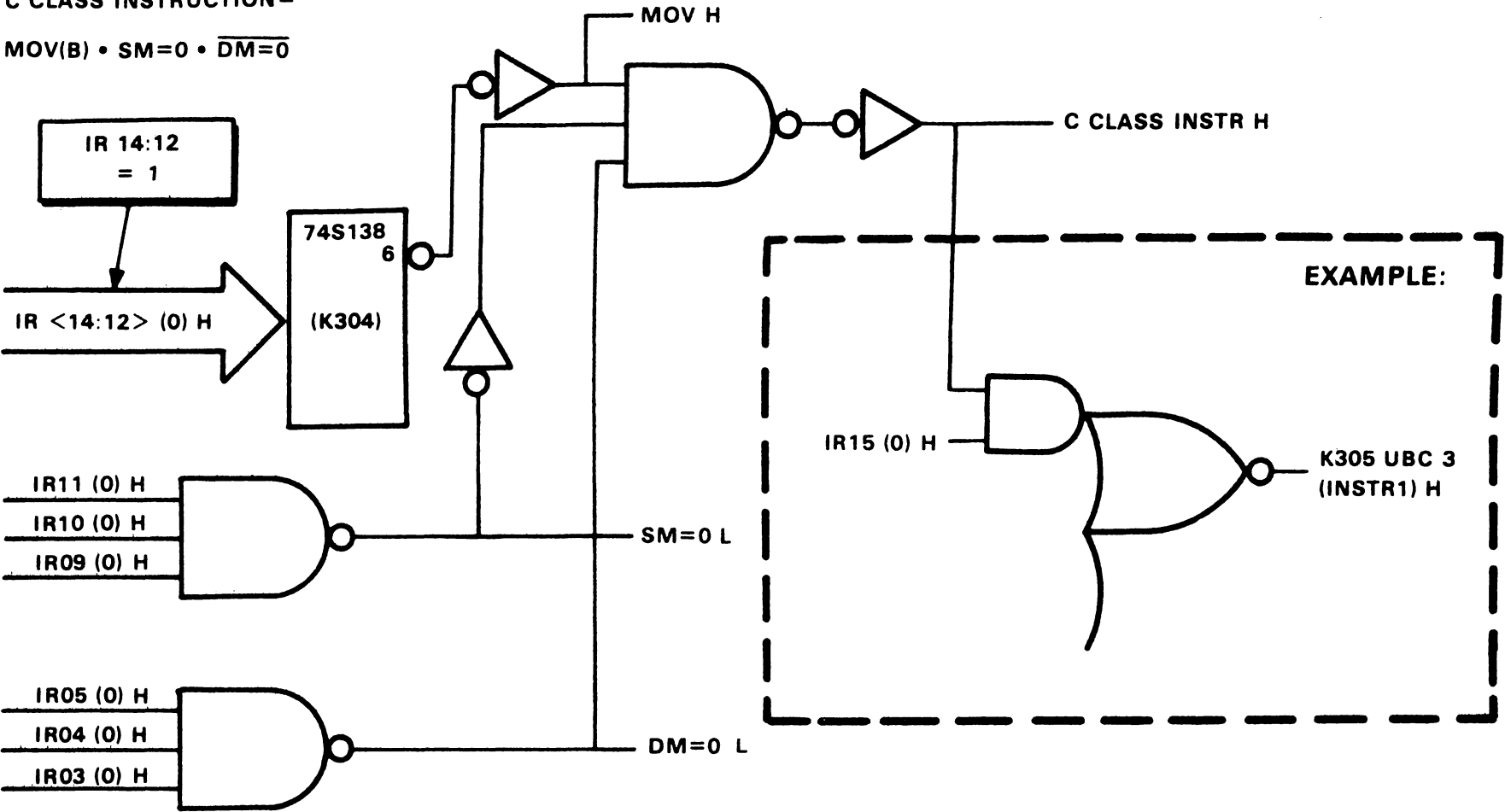
(SINGLE OPERAND • $\overline{\text{DM}}=0$)



11-5587

Figure 4-13 B Class Instruction Decoding

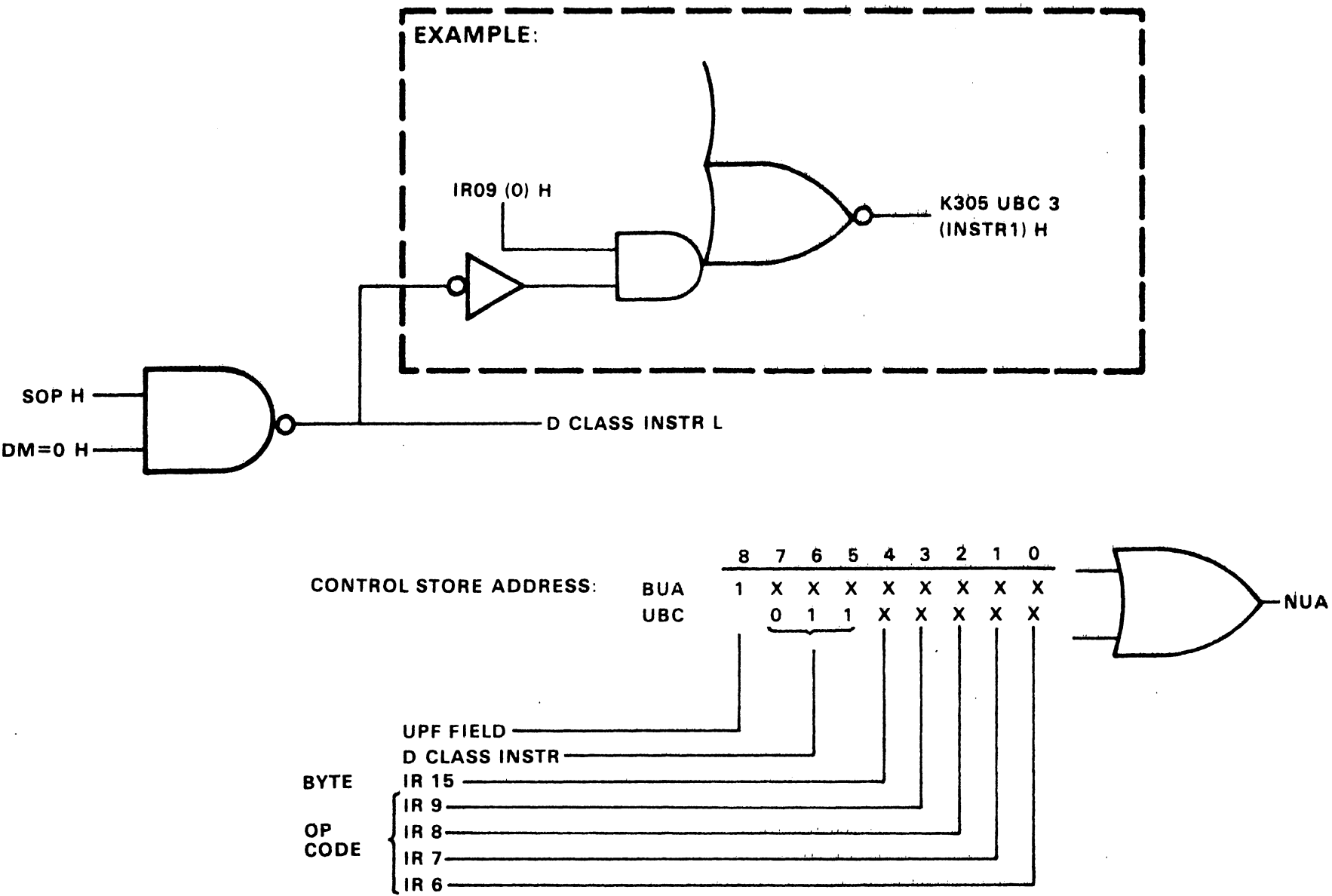
C CLASS INSTRUCTION=
 MOV(B) • SM=0 • $\overline{DM=0}$



11-5588

Figure 4-14 C Class Instruction Decoding

D CLASS INSTRUCTION=
 SINGLE OPERAND (BYTE) • DM=0



11-5589

Figure 4-15 D Class Instruction Decoding

```

JMP      (IR<15:06>=0001)

```

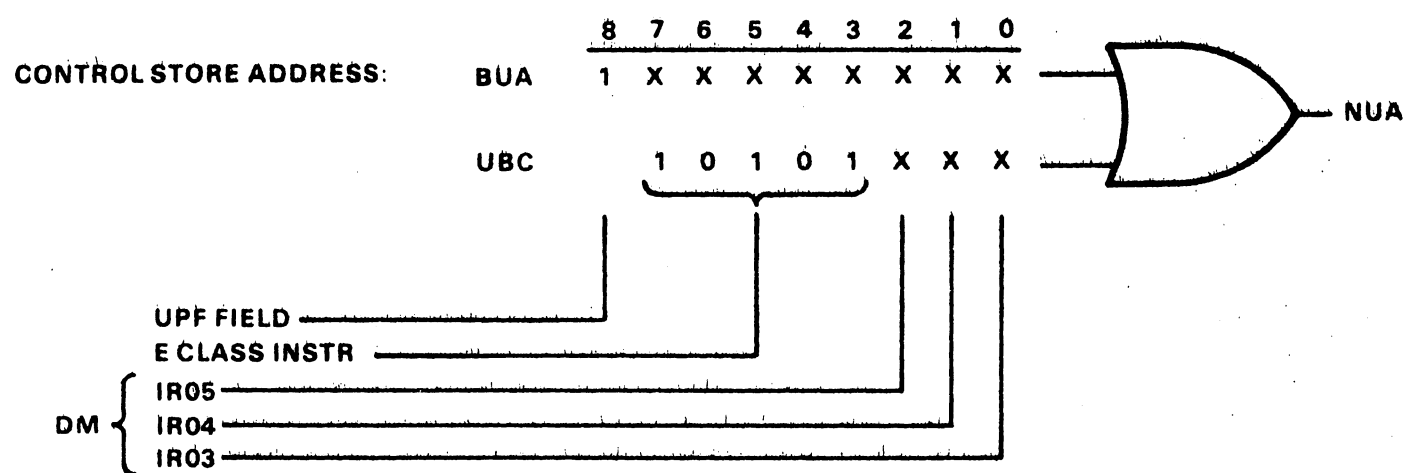
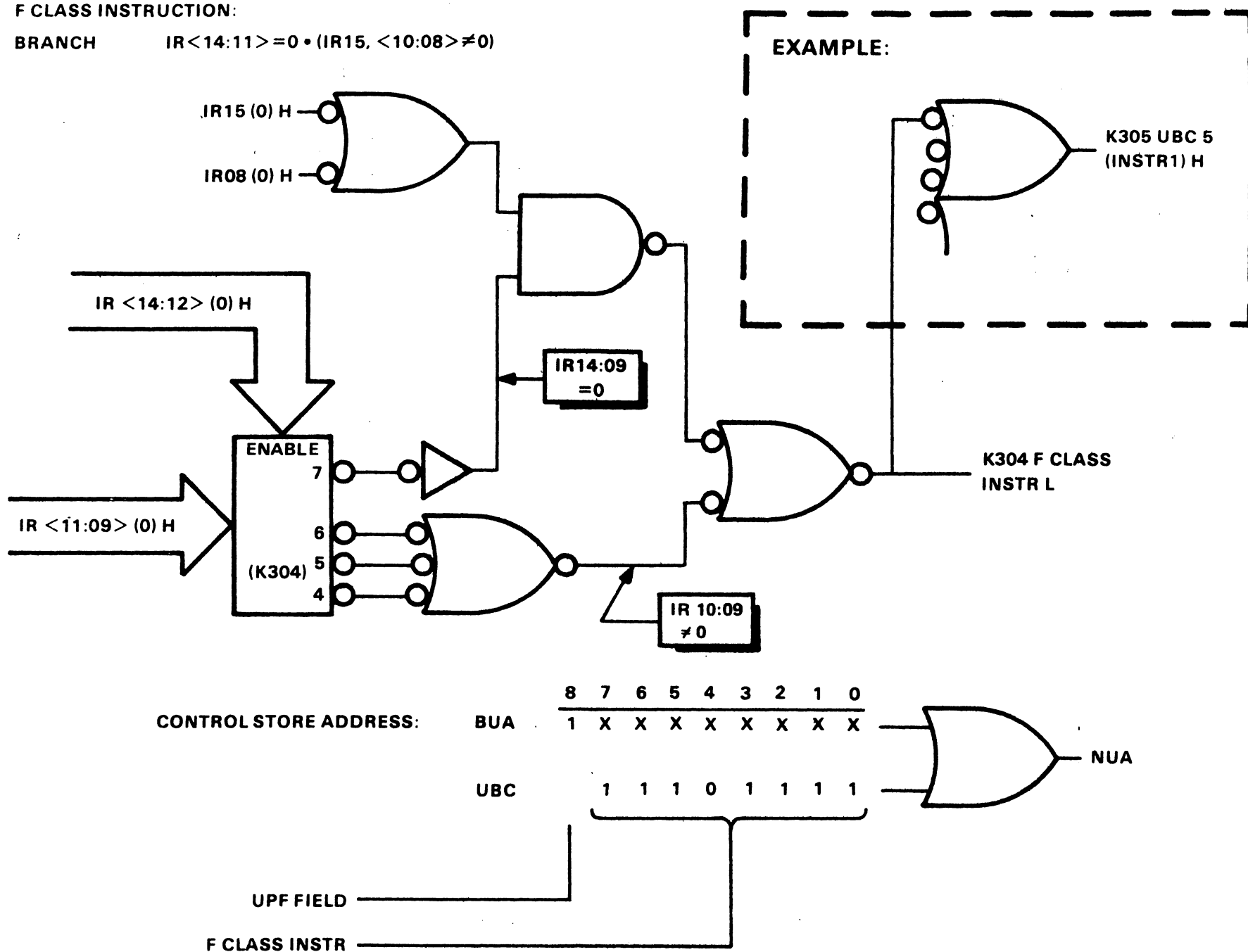


Figure 4-16 E Class Instruction Decoding

F CLASS INSTRUCTION:

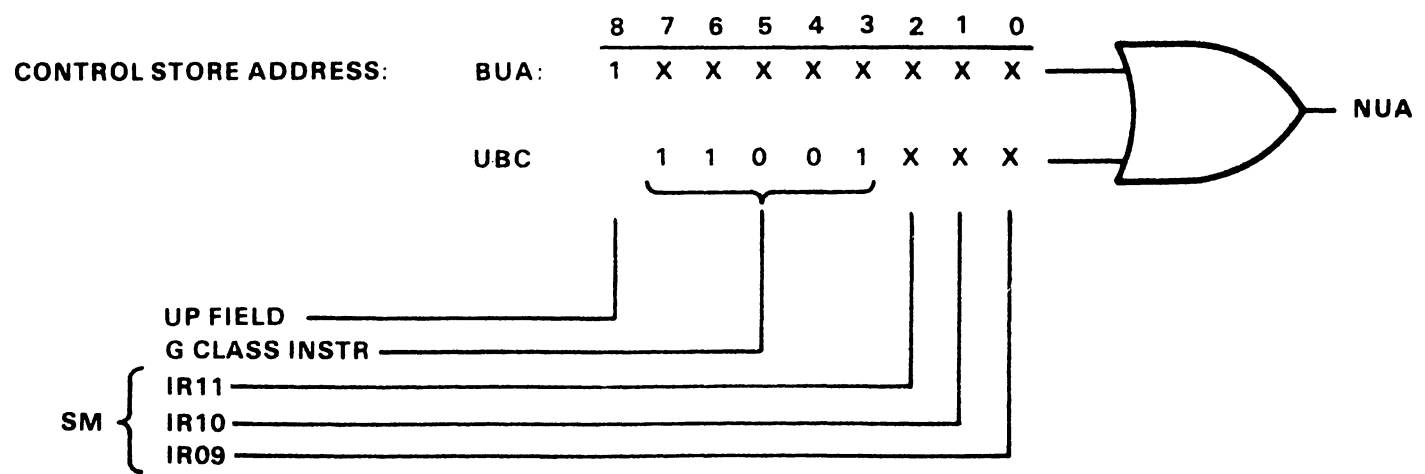
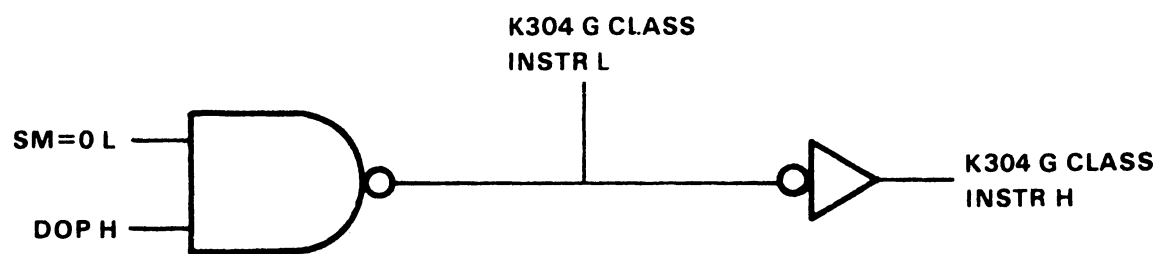
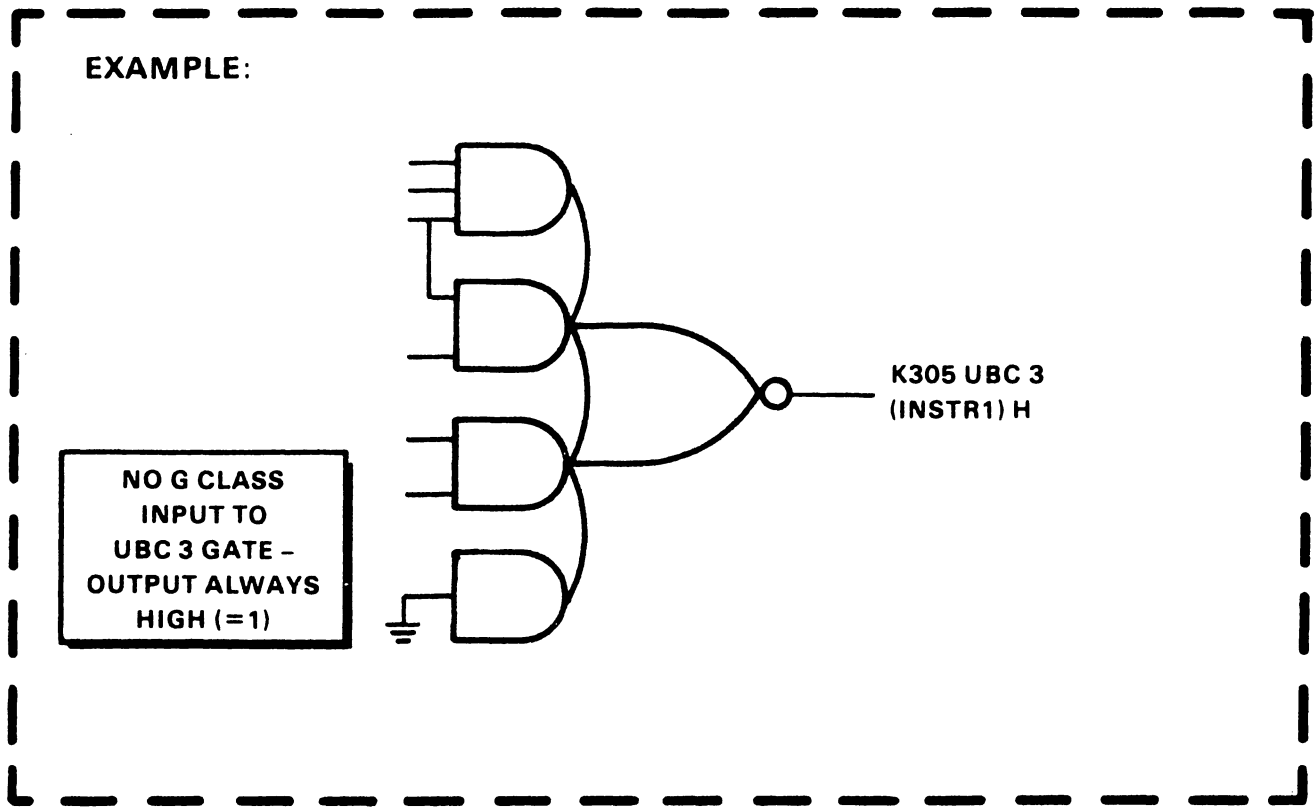
BRANCH $IR\langle 14:11 \rangle = 0 \cdot (IR15, \langle 10:08 \rangle \neq 0)$



11-5591

Figure 4-17 F Class Instruction Decoding

G CLASS INSTRUCTION =
DOUBLE OPERAND • $\overline{SM}=0$



11-5592

Figure 4-18 G Class Instruction Decoding

Table 4-6 Instruction 5 Branch ROM (E78)

Instruction	Input (Note 1) K303 IR XX (1) H									Output K303 UBCX (INSTR5) H				
	07	06	05	04	03	02	01	00	Octal	Octal	3	2	1	0
(Reserved)										0	0	0	0	0
WAIT	0	0	0	0	0	0	0	1	1	1	0	0	0	1
RTI	0	0	0	0	0	0	1	0	2	2	0	0	1	0
BPT	0	0	0	0	0	0	1	1	3	3	0	0	1	1
IOT	0	0	0	0	0	1	0	0	4	4	0	1	0	0
JMP (Note 2)	0	1	1	0	1	0	1	0	152	5	0	1	0	1
RTS	1	0	0	0	0	0	0	0	200	6	0	1	1	0
↓										↓				
RTS	1	0	0	0	0	1	1	1	207	6	0	1	1	0
RESET	0	0	0	0	0	1	0	1	5	7	0	1	1	1
SCC, CCC, NOP	1	0	1	0	0	0	0	0	240	10	1	0	0	0
↓										↓				
SCC, CCC, NOP	1	0	1	1	1	1	1	1	277	10	1	0	0	0
SWAB • DM=0	1	1	0	0	0	0	0	0	306	11	1	0	0	1
↓										↓				
SWAB • DM=0	1	1	0	0	0	1	1	1	307	11	1	0	0	1
JMP (Note 2)	0	1	0	1	0	1	0	1	125	12	1	0	1	0
SWAB • $\overline{DM}=0$	1	1	0	0	0	0	0	0	310	13	1	0	1	1
↓										↓				
SWAB • $\overline{DM}=0$	1	1	1	1	1	1	1	1	377	13	1	0	1	1
HALT	0	0	0	0	0	0	0	0	0	14	1	1	0	0
(not assigned)										15	1	1	0	1
RTT	0	0	0	0	0	1	1	0	6	16	1	1	1	0
(not assigned)										17	1	1	1	1

Notes: 1. 0=Low, 1=High.

2. Diagnostics.

Table 4-7 Instruction 5 Branch ROM (E88)

Instruction	Input (Note 1)										Output				
	K304 DM=0 H K304 IR (14:12)=0 L K303 IR XX (1) H 15 11 10 09 08 07 06 Octal										K303 UBC X (INSTR5) H Octal 3 2 1 0				
(Reserved)											0	0	0	0	0
FIS	0	1	0	1	0	1	0	0	0	250	1	0	0	0	1
					.					.					
					.					.					
					.					.					
	0	1	0	1	0	1	1	1	1	257					
	1	1	0	1	0	1	0	0	0	650					
					.					.					
					.					.					
					.					.					
FIS	1	1	0	1	0	1	1	1	1	657	1	0	0	0	1
EIS (MUL, DIV, etc.)	0	1	0	0	0	0	0	0	0	200	2	0	0	1	0
					.					.					
					.					.					
					.					.					
	0	1	0	0	1	1	1	1	1	237					
	1	1	0	0	0	0	0	0	0	600					
					.					.					
					.					.					
					.					.					
EIS (MUL, DIV, etc.)	1	1	0	0	1	1	1	1	1	637	2	0	0	1	0
FAST X 1	0	0	1	1	1	1	0	0	0	170	3	0	0	1	1
					.					.					
					.					.					
					.					.					
	0	0	1	1	1	1	1	1	1	177					
	1	0	1	1	1	1	0	0	0	570					
					.					.					
					.					.					
					.					.					
FAST X 1	1	0	1	1	1	1	1	1	1	577	3	0	0	1	1
S X T	0	0	0	1	1	0	1	1	1	67	4	0	1	0	0
S X T	1	0	0	1	1	0	1	1	1	467	4	0	1	0	0

Table 4-7 Instruction 5 Branch ROM (E88) (Cont)

Instruction	Input (Note 1)										Output				
	K304 DM=0 H K304 IR (14:12)=0 L K303 IR XX (1) H		15	11	10	09	08	07	06	Octal	K303 UBC X (INSTR 5) H				
	↓	↓									Octal	3	2	1	0
FLTPT (Note 2)	0	1	1	0	1	0	1	0	1	325	5	0	1	0	1
INC (Note 2)	1	0	0	1	0	1	0	1	0	452	5				
XOR • DM=0	1	1	0	1	0	0	0	0	0	640	6	0	1	1	0
↓					.					.	↓		↓		
XOR • DM=0	1	1	0	1	1	1	1	1	1	647	6	0	1	1	0
XOR • DM=0	0	1	0	1	0	0	0	0	0	240	7	0	1	1	1
↓					.					.	↓		↓		
XOR • DM=0	0	1	0	1	0	0	1	1	1	247	7	0	1	1	0
SOB	0	1	0	1	1	1	0	0	0	270	10	1	0	0	0
↓					.					.	↓		↓		
					.					.	↓		↓		
					.					.	↓		↓		
	0	1	0	1	1	1	1	1	1	277					
	1	1	0	1	1	1	0	0	0	670					
↓					.					.	↓		↓		
					.					.	↓		↓		
SOB	1	1	0	1	1	1	1	1	1	677	10	1	0	0	0
MARK	0	0	0	1	1	0	1	0	0	64	11	1	0	0	1
MARK	1	0	0	1	1	0	1	0	0	464	11	1	0	0	1
FLTPT (Note 2)	1	1	1	0	1	0	1	0	1	725	12	1	0	1	0
FLTPT (Note 2)	1	1	1	1	0	1	0	1	0	752	12	1	0	1	0
MFPI, MFDI	0	0	0	1	1	0	1	0	1	65	13	1	0	1	1
↓											↓		↓		
	0	0	1	1	1	0	1	0	1	165					
↓											↓		↓		
MDPI, MFDI	1	0	0	1	1	0	1	0	1	465	13	1	0	1	1
MTPI, MTDI	0	0	0	1	1	0	1	1	0	66	14	1	1	0	0
↓											↓		↓		
	0	0	1	1	1	0	1	1	0	166					
↓											↓		↓		
MTPI, MTDI	1	0	0	1	1	0	1	1	0	466	14				
	1	0	1	1	1	0	1	1	0	566	14	1	1	0	0

Notes: 1. 0=Low, 1=High.

2. Diagnostics.

Table 4-7 Instruction 5 Branch ROM (E88) (Cont)

Instruction	Input (Note 1)										Output				
	K304 DM=0 H K304 IR (14:12)=0 L K303 IR XX (1) H		15	11	10	09	08	07	06	Octal	K303 UBC X (INSTR 5) H				
	↓	↓									Octal	3	2	1	0
MTPI, MTDI EMT	1	0	1	1	1	0	1	1	0	566	14	1	1	0	0
	0	0	1	1	0	0	0	0	0	140	15	1	1	0	1
↓					.					.					
					.					.					
EMT TRP	0	0	1	1	0	0	0	1	1	143					
	1	0	1	1	0	0	0	0	0	540					
↓					.					.					
					.					.					
TRP JRS	1	0	1	1	0	0	0	1	1	543	15	1	1	0	1
	0	0	1	1	0	0	1	0	0	144	16	1	1	1	0
↓					.					.					
					.					.					
↓	0	0	1	1	0	0	1	1	1	147					
	1	0	1	1	0	0	1	0	0	544					
↓					.					.					
					.					.					
↓	1	0	1	1	0	0	1	1	1	547	16	1	1	1	0
	0	0	0	1	0	0	0	0	0	40	17	1	1	1	1
↓					.					.					
					.					.					
↓	0	0	0	1	0	0	1	1	1	47					
	1	0	0	1	0	0	0	0	0	440					
↓					.					.					
					.					.					
JSR	1	0	0	1	0	0	1	1	1	447	17				

Notes: 1. 0=Low, 1=High.
2. Diagnostics.

Floating point branch condition signals, K305 UBC (3:0) (FLTPT) H, are selected as inputs to the microbranch multiplexer by BUT 03 [IR11*FLTPT (3-0)]. These signals are the output of a 6206 type ROM (E86) shown on drawing K305. Table 4-8 lists the origins of all output configurations.

Table 4-8 Floating Point Branch ROM (E86)

Input IR(12:3)#FD	Output	Instruction IR11=0	Instruction IR11=1
00 020 XXX X	00	SETCLASS	STF1
00 0XX XXX X	01	STATUS	STF2
00 1XX 000 X	02	CTAN*DM0	DIV*DM0
20 1XX XXX X	03	CTAN*-DM0	DIV*-DM0
21 0XX 000 0	04	MULF*DM0	STEXPf*DM0
21 0XX 000 1	05	MULD*DM0	STEXPd*DM0
01 0XX XXX X	06	MULX*-DM0	STEXPX*-DM0
01 1XX XXX X	07	MODFD	STCT
10 0XX 000 X	10	ADD*DM0	STCP*DM0
10 0XX XXX X	11	ADD*-DM0	STCP*-DM0
10 1XX 000 X	12	LDFD*DM0	LDEXP*DM0
10 1XX 110 X	13	LDFD*DM6	LDEXP*DM6
10 1XX XXX X	14	LDFD*-(DM0+DM6)	LDEXP*-(DM0+DM6)
11 0XX 000 X	15	SUB*DM0	LDCT*DM0
11 0XX XXX X	16	SUB*-DM0	LDCT*-DM0
11 1XX XXX X	17	CMP	LDCP

4.6.3.3 Condition Code Control (V and C) – The overflow (V) and carry (C) condition code bits, K303 CC (V) H and K303 CC (C) H, are generated by the condition control logic shown on drawing K303 and on Figures 4-8 and 4-19. These two signals provide the source for the PSW V and C bits (Paragraph 3.2.2).

The circuit consists of two, type 6206 ROM chips and a 74S174, quad D-type, flip-flop chip. ROM E85 receives its address input from the nine high-order bits in the IR, K303 IR (15:06) (1) H, and the entire macroinstruction except for the source and destination fields. The ROM has four outputs: X, Y, Z (arbitrary designations), and K303 BYTE H. Table 4-9 lists those instructions that result in at least one (high) output. For example, the compare instruction (CMP) produces highs at the Y and X outputs while a compare byte (CMPB) instruction causes the assertion of K303 BYTE H in addition to the highs at X and Y. Other instructions, e.g., BITB and BICB, assert only the K303 BYTE H output (BIT and BIC produce no outputs at all). All four outputs from ROM E85 are clocked into the 74S174 at P2 time. K303 BYTE H is also one of the inputs to the byte decode logic (K312) that is used with the autodecrement and autoincrement instructions (Paragraph 4.6.3.4)

K303 BYTE H is redesignated K303 BYTE (1) H at the output of the 74S174. This signal, in turn, becomes K307 BYTE INST H. (Destinations are indicated on Figure 4-19.) The other five 74S174 outputs, together with the N bit, the carry out bit [K404 D(C) B H, Paragraph 3.2.2], the V bit, and a modify V signal, provide the address inputs to the other ROM (E62) in the circuit.

Table 4-10 lists the input bit configurations for those instructions that result in either the V or C output from the ROM. Note that the V bit is asserted by either one of two input configurations for each instruction.

Table 4-9 Overflow-Carry ROM (E85)

Instruction	Output (See Note 1)				Input [K303 IR XX(1) H]								
	BYTE	Z	Y	X	15	14	13	12	10	9	8	6	7
	(Note 2)				A8	A7	A6	A5	A4	A3	A2	A1	A0
CMP (B)	H	L	H	H	1/0	0	1	0	X	X	X	X	X
BIT (B)	H	L	L	L	1	0	1	1	X	X	X	X	X
BIC (B)	H	L	L	L	1	1	0	0	X	X	X	X	X
BIS (B)	H	L	L	L	1	1	0	1	X	X	X	X	X
MOV (B)	H	L	L	L	1	0	0	1	X	X	X	X	X
CLR (B)	H	L	L	L	1	0	0	0	0	1	0	0	0
COM (B)	H	L	L	L	1	0	0	0	0	1	0	0	1
INC (B)	H	L	H	L	1/0	0	0	0	0	1	0	1	0
DEC (B)	H	L	H	H	1/0	0	0	0	0	1	0	1	1
NEG (B)	H	L	H	H	1/0	0	0	0	0	1	1	0	0
TST (B)	H	L	L	L	1	0	0	0	0	1	1	1	1
ROR (B)	H	L	L	H	1/0	0	0	0	1	0	0	0	0
ROL (B)	H	L	L	H	1/0	0	0	0	1	0	0	0	1
ASR (B)	H	L	L	H	1/0	0	0	0	1	0	0	1	0
ASL (B)	H	L	L	H	1/0	0	0	0	1	0	0	1	1
SWAB	H	L	L	L	0	0	0	0	0	0	0	1	1
SBC (B)	H	L	H	H	1/0	0	0	0	0	1	1	1	0
ASH	L	H	H	H	0	1	1	1	1	0	0	0	0
ASHC	L	H	H	H	0	1	1	1	1	1	0	0	0
ADD	L	L	H	L	0	1	1	0	X	X	X	X	X
SUB	L	L	H	H	1	1	1	0	X	X	X	X	X
ADC (B)	H	L	H	L	1/0	0	0	0	0	1	1	0	1

Notes: 1. ZYX: Refer to Figure 4-19.

2. BYTE: K303 BYTE H; asserted only for byte instructions and SWAB.

102

Table 4-10 Overflow-Carry ROM (E62)

Instructions	Output (See Note)		Input (See Note)								Remarks	
	V	C	A8	A7	A6	A5	A4	A3	A2	A1		A0
ASR(B), ASL(B)	H	{	X	X	X	X	L	H	L	L	L	V set if number becomes negative and no carry or if number becomes positive and a carry.
			X	X	X	X	H	H	L	L	H	
			X	X	X	X	H	H	H	L	X	
ADD, INC(B), ADC(B)	H	{	X	X	H	H	X	L	H	L	H	V set if arithmetic overflow; ALU operation: A + B.
			X	X	L	L	X	L	H	L	L	
			X	X	X	X	H	H	H	L	X	
SBC(B), SUB, CMP(B), NEG(B)	H	{	X	X	H	L	X	H	H	L	L	V set if arithmetic overflow; ALU operation: A – B.
			X	X	L	H	X	H	H	L	H	
			X	X	X	X	L	H	H	L	X	
DEC(B)	H	{	X	X	H	L	X	H	L	H	L	V set if arithmetic overflow; ALU operation: A – B.
			X	X	L	H	X	H	L	H	H	
			X	X	X	X	H	H	H	L	X	
ASH, ASHC	H	{	H	L	H	X	X	H	H	H	L	
			H	L	L	X	X	H	H	H	H	
			X	X	X	X	H	H	H	H	X	
Any other bit Combination in the IR	L	{	X	X	X	X	X	L	L	L	X	
			X	X	X	X	H	L	L	L	X	

Note: Codes are:

V: K303 CC (V) H

C: K303 CC (C) H

A8: K306 PS (V) (1) H

A7: K311 MODIFY V (1) H

A6: K306 B BUS SIGN H

A5: K306 A BUS SIGN H

A4: K404 D(C) B H

A3: } X

A2: } Y Refer to Figure 4-19

A1: } Z

A0: K306 CC (N) L (Note: N bit is asserted low.)

4.6.3.4 Byte Decode – The byte decode logic (drawing K312 and Figure 4-8) determines if the source or destination field of the instruction in the IR specifies the autoincrement or autodecrement (addressing modes 2 or 4) of a byte address. The output signal, K312 BYTE CONST H, asserted if an instruction of this type is present, disables the CSP address multiplexer in the data path module (K407). This causes CSP location 17_8 to be accessed; location 17_8 contains the constant 1 (Paragraph 3.7).

Table 4-11 lists those input bit configurations to the byte decode ROM (E60) from the IR, preliminary decode logic, overlap decode logic, and auxiliary control store that result in the assertion of K312 BYTE CONST H.

4.6.3.5 Instruction Branch ROM – Another input to the microbranch multiplexer is a signal from the instruction branch ROM (E83), K307 INSTR BRANCH L. This ROM, shown on Figure 4-8, decodes the state of the condition code bits N, Z, V, and C when the IR contains the code for a branch instruction. Table 4-12 lists those conditions that cause the assertion of K307 INSTR BRANCH L. When selected by BUT (INTR*SUCBRANCH), UBF field = 26, K307 INSTR BRANCH L determines the state of the low-order bit position of the microbranch multiplexer (K308). 103

If asserted, K307 INSTR BRANCH L causes K308 UBRANCH 0 L = 0 and the branch will occur. [When K308 UBRANCH 0 L = 0, the state of K206 NUA 00 H is determined by the state of K203 BUA (1) L]. However, if the required conditions for a branch are not met (K307 INSTR BRANCH L is high, = 0), K308 UBRANCH 0 L = 1. This causes the program to fall through to the next control store address.

4.7 OVERLAP DECODE

The overlap decode logic is used to detect those macroinstructions that allow the prefetch of the next macroinstruction or data word. The purpose is to decrease the macroprogram run time. In a prefetch, part or all of the microcode used to fetch the next word is bypassed.

Two methods of information retrieval are employed: overlapped and nonoverlapped. Figure 4-20 shows the microinstructions required for both types of fetches. In a nonoverlapped fetch, the second macroinstruction is retrieved after the execution of the first macroinstruction is completed. One or more microinstructions (microwords X and Y) are required to execute the first macro. The operation is serial; an instruction is read out from cache (or backing store) after the previous macroinstruction has performed its operation. On the other hand, in an overlapped instruction fetch, part of the microcode for the second fetch is bypassed. The procedure is still performed serially but without the need for all of the microcode. However, only macroinstructions that are executed by a single microinstruction (microword X) can be overlapped.

An overlap will also occur if the source or destination operand of the first macro is in addressing mode 6 or 7 (index or index deferred). The data word that follows an instruction of this type is fetched without repeating the fetch microcode.

A macrofetch requires three microinstructions: FET01, FET02, and FET03. During the execution of these instructions the macroinstruction is addressed, read from storage, placed in the IR, and decoded. The macro is executed after the fetch is completed. For the execution of a nonoverlapped macro, one or more, additional microinstructions are required. However, macroinstructions that are overlapped (FET01 and FET02 are bypassed) require only a single microinstruction for execution. In this case, the fetch of the second macroinstruction begins during FET03 of the first word fetch and is completed by the single microinstruction that follows. Mode 6 or 7 prefetches are performed simply to speed up the retrieval power; no calculation on the source or destination is necessary when an index is fetched. After the mode 6 or mode 7 microflow is executed, the operation returns to FET01 for the next macroinstruction.

Table 4-11 Byte Decode ROM (E60)

Instructions	Input (See Note)										Output
	A8	A7	A6	A5	A4	A3	A2	A1	A0	(K312 BYTE CONST H)	
Double operand byte, source mode=auto inc/auto DEC	X	H	L	X	X	L	H	H	H	H	
double operand byte, source mode=register, destination mode=auto inc/auto DEC, destination register not stack or PC	H	X	L	X	H	X	H	H	H	H	
Single operand byte, destination mode=auto inc/auto DEC, destination register not stack or PC	H	X	L	H	X	X	H	X	H	H	
Double operand byte, destination mode=auto inc/auto DEC, destination register not stack or PC	H	X	H	X	X	X	H	H	L	H	

Note: A8: DM=2+4•DR=6+7 H

A7: SM=2+4 H

A6: DAD-1(1) H

A5: SOP H

A4: SM=0 H

A3: SR=6+7 L

A2: BYTE H

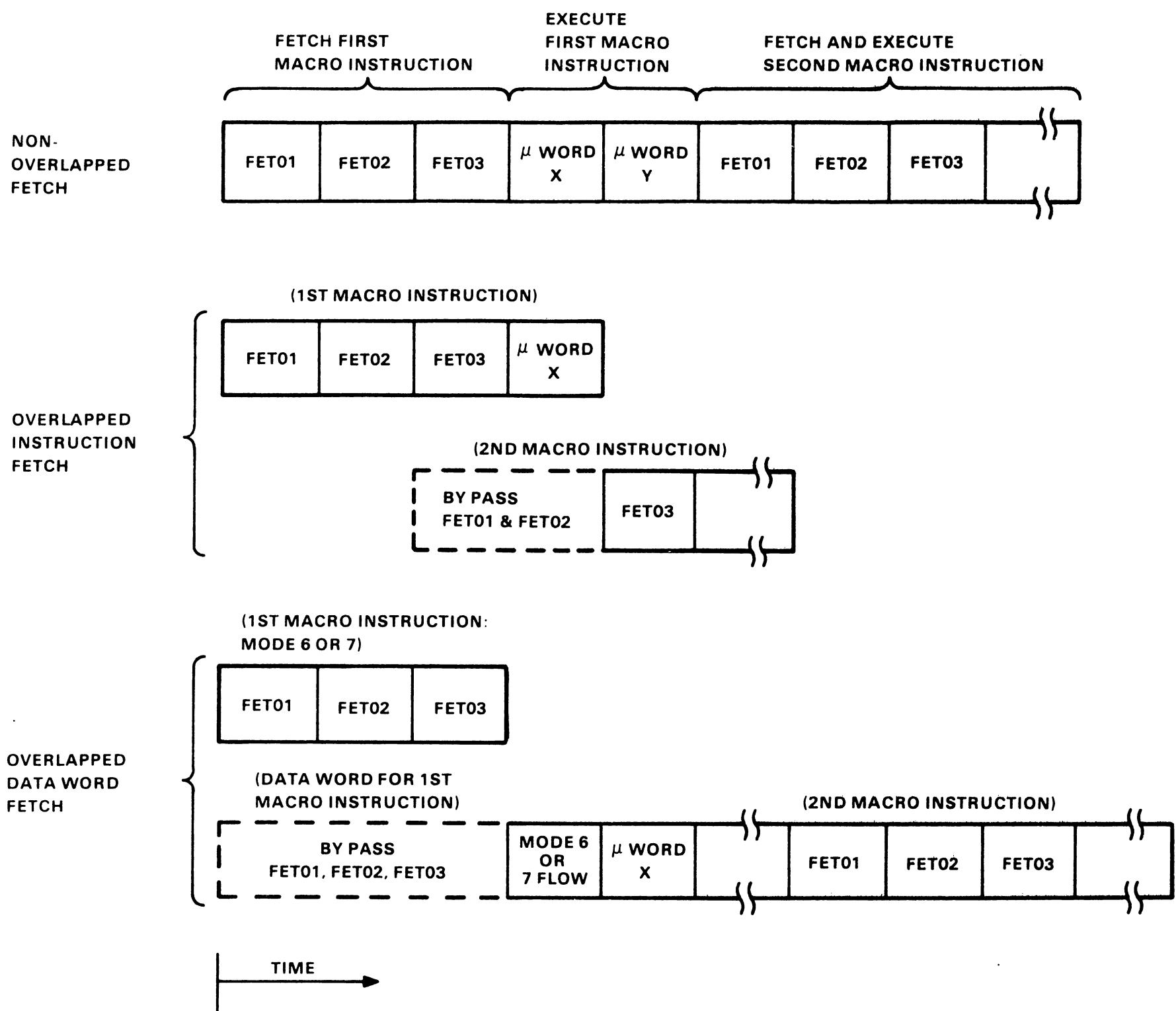
A1: DOP H

A0: DAD 0(1) H

Table 4-12 Instruction Branch ROM (E83)

Instruction	Input [K306 PS (N,Z,V,C) (1) H; K303 IR XX (1) H]								Output (See Note) K307 INSTR BRANCH L asserted if:
	N A7	Z A6	V A5	C A4	15 A3	10 A2	09 A1	08 A0	
BR	X	X	X	X	0	0	0	1	Unconditional
BNE	X	0	X	X	0	0	1	0	$\overline{Z}$
BEQ	X	1	X	X	0	0	1	1	Z
BPL	0	X	X	X	1	0	0	0	$\overline{N}$
BMI	1	X	X	X	1	0	0	1	N
BVC	X	X	0	X	1	1	0	0	$\overline{V}$
BVS	X	X	1	X	1	1	0	1	V
BCC (BHIS)	X	X	X	0	1	1	1	0	$\overline{C}$
BCS (BLO)	X	X	X	1	1	1	1	1	C
BGE	{ 0 1 }	X	0	X	0	1	0	0	$(N \bullet V) + (\overline{N} \bullet \overline{V})$
		X	1	X	0	1	0	0	
BLT	{ 1 0 }	X	0	X	0	1	0	1	$(N \bullet \overline{V}) + (\overline{N} \bullet V)$
		X	1	X	0	1	0	1	
BGT	{ 1 0 }	0	1	X	0	1	1	0	$(\overline{Z} \bullet N \bullet V) + (\overline{Z} \bullet \overline{N} \bullet \overline{V})$
		0	0	X	0	1	1	0	
BLE	{ X 1 }	1	X	X	0	1	1	1	$Z + (N \bullet \overline{V}) + (\overline{N} \bullet V)$
		X	0	X	0	1	1	1	
BHI	{ X X }	0	X	X	1	0	1	0	$\overline{C} + \overline{Z}$
		X	X	0	1	0	1	0	
BLOS	{ X X }	1	X	X	1	0	1	1	C + Z
		X	X	1	1	0	1	1	

Note: The branch will occur if the output signal (K307 INSTR BRANCH L) is asserted.



11-5594

Figure 4-20 Overlapped and Nonoverlapped Fetches

Table 4-13 lists those types of macroinstructions that result in the assertion of K306 OVERLAP L and K306 PREFETCH H by the overlap decode ROM (E80). Figure 4-21 shows the applications of these two signals.

Figure 4-22 is a flow diagram of the microcode used to fetch macroinstructions. The flow is entered at microinstruction FET01 during which the virtual memory address of the first macroword is clocked into the BA register from the PC, R7 in ASPLO. This action initiates a DATI to cache, or to the backing store if the data is not resident in cache. The PC is then updated to reflect the address of the second macroword and K605 DATI (CLKIR) (1) H, which enables the IR, is asserted (Figure 4-9).

In FET02 the result of the DATI (the data in the DMUX) is clocked into both the IR and the MD register and the decoding process begins. Then, early (P1) in the execution of FET03, the updated address in the PC is clocked into the BA register. This initiates a second DATI, which allows a head start on the retrieval of the second macroword from storage. The PC is updated again (to the third word address) and placed in the D register.

If, after decoding the first macroword the prefetch of a second word is disallowed, the DATI started at the beginning of the cycle (not P1 dependent) is inhibited. The PC is also inhibited from receiving the D register contents (address of third word). Instead, the processor executes the microinstructions necessary to complete the first macroinstruction. BUT (FOV/SERVICE), located in the last microword for this macro, causes the flow to return to FET01 and initiate the retrieval of the next (second) macro.

However, if the first macroword is one of the types listed in Table 4-13 and a prefetch is allowed (K306 OVERLAP L asserted), the overlap decode logic must also determine if this is to be a data word prefetch or an instruction word prefetch. Each type of prefetch uses a unique microflow path (Figure 4-22). In both paths the D register contents (address of third macro) are transferred to the PC (ASPLO register 7). This is possible because K306 OVERLAP L inhibits K306 INH SPLO L; with the latter signal high, the scratchpad WR inputs are enabled at P3 time (Figure 4-21).

In a data word prefetch this is followed by the addressing mode 6 or 7 microflow, and the data word (the result of the DATI initiated in FET03) is clocked into the MD register. Then, the microcode necessary to complete the first macroinstruction is executed and BUT (FOV/SERVICE) returns the operation to FET01. The signal K306 PREFETCH is asserted if the overlap decode logic determines that an instruction prefetch is allowed. This signal, ANDed with K207 BUT (INST 1) H (asserted during FET03) inhibits K207 BC0/PREFETCH H. Consequently, the DATI [K605 DATI (1) H], initiated at the beginning of FET03, is changed to DATI CLKIR [K605 DATI (CLK IR) (1) H], the IR enable signal (Figure 4-9).

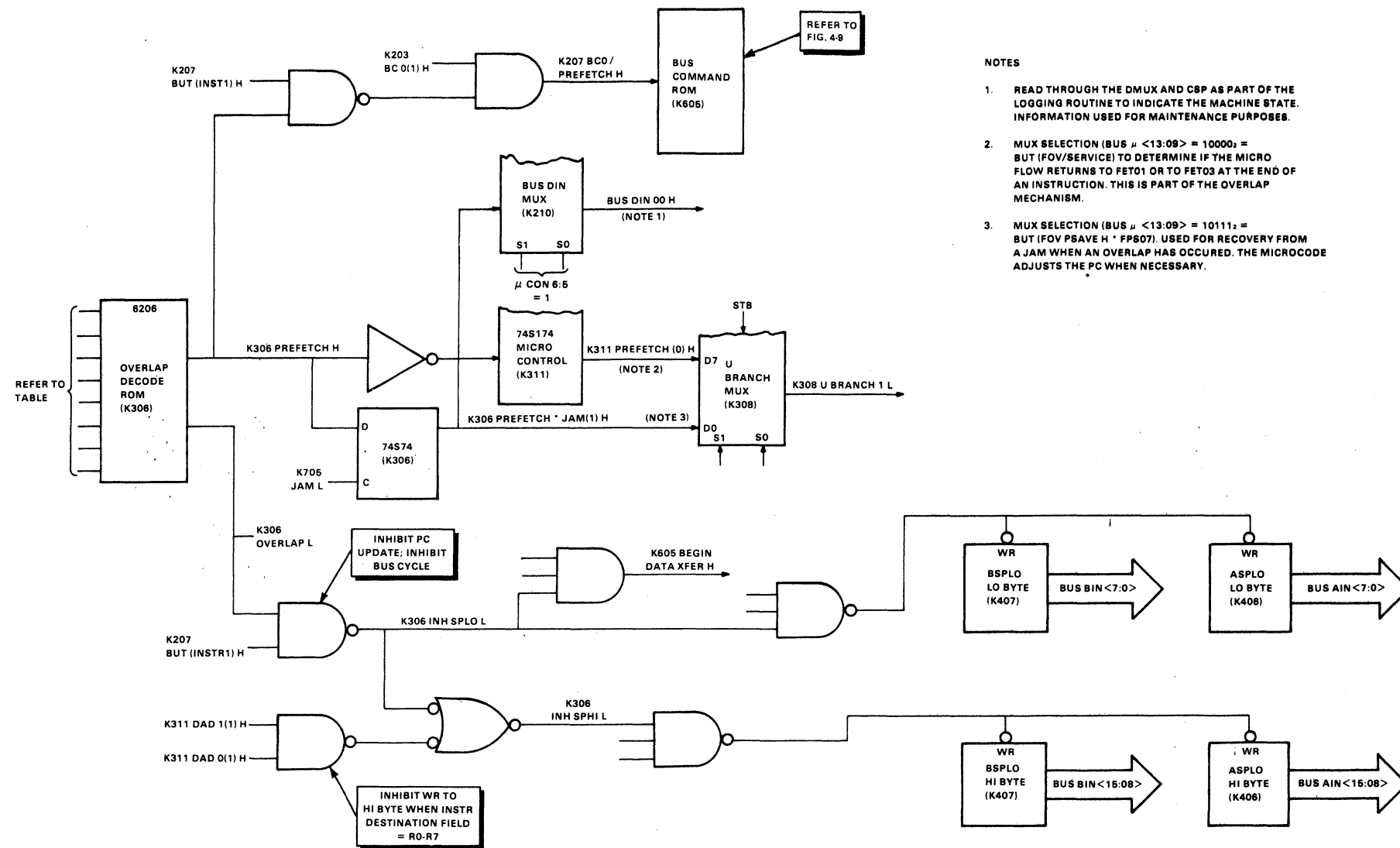
A single microword is now required to complete the instruction prefetch. This microword causes the transfer of the next macroinstruction (read from storage as the result of the FET03 DATI) into the IR and the MD register. BUT (FOV/SERVICE) then directs the flow back to FET03. Note that the operations that result from the microcode in the instruction prefetch path duplicate those operations carried out in FET01 and FET02. Therefore, FET01 and FET02 can be bypassed: an instruction prefetch is performed.

Table 4-13 Overlap Decode ROM (E80)

Input	Output		Notes
	K306 OVERLAP L	K306 PREFETCH H	
A8 A7 A6 A5 A4 A3 A2 A1 A0			
DOP•SM=0•DM=0• $\overline{\text{MOV}}\text{B}\cdot\overline{\text{SR}}7\cdot\overline{\text{DR}}7$	L	H	1
SOP•DM=0•(IR06+IR08⊕IR09)• $\overline{\text{DR}}7$	L	H	2
DOP•SM=6+7	L	L	3
DOP•SM=0•DM=6+7	L	L	3
SOP•DM=6+7	L	L	3

Notes:

1. The word following the current instruction word has to be another instruction: Word R to R and SR≠7, DR≠7.
2. All single operand instructions except NEG, ROR, ASR, and SBC. All others are one word executes. (The ROM inputs defined in Notes 1 and 2 constitute all the one word executes and cause the microflow to return to FET03 because of K306 PREFETCH H. During the execute state (following FET03), the data obtained during the DATA CLK IR in FET03 is placed in the IR.)
3. Double operand or single operand instruction Mode 6 or 7 that are followed by a data word used as an index. A word fetch is allowed in this case.



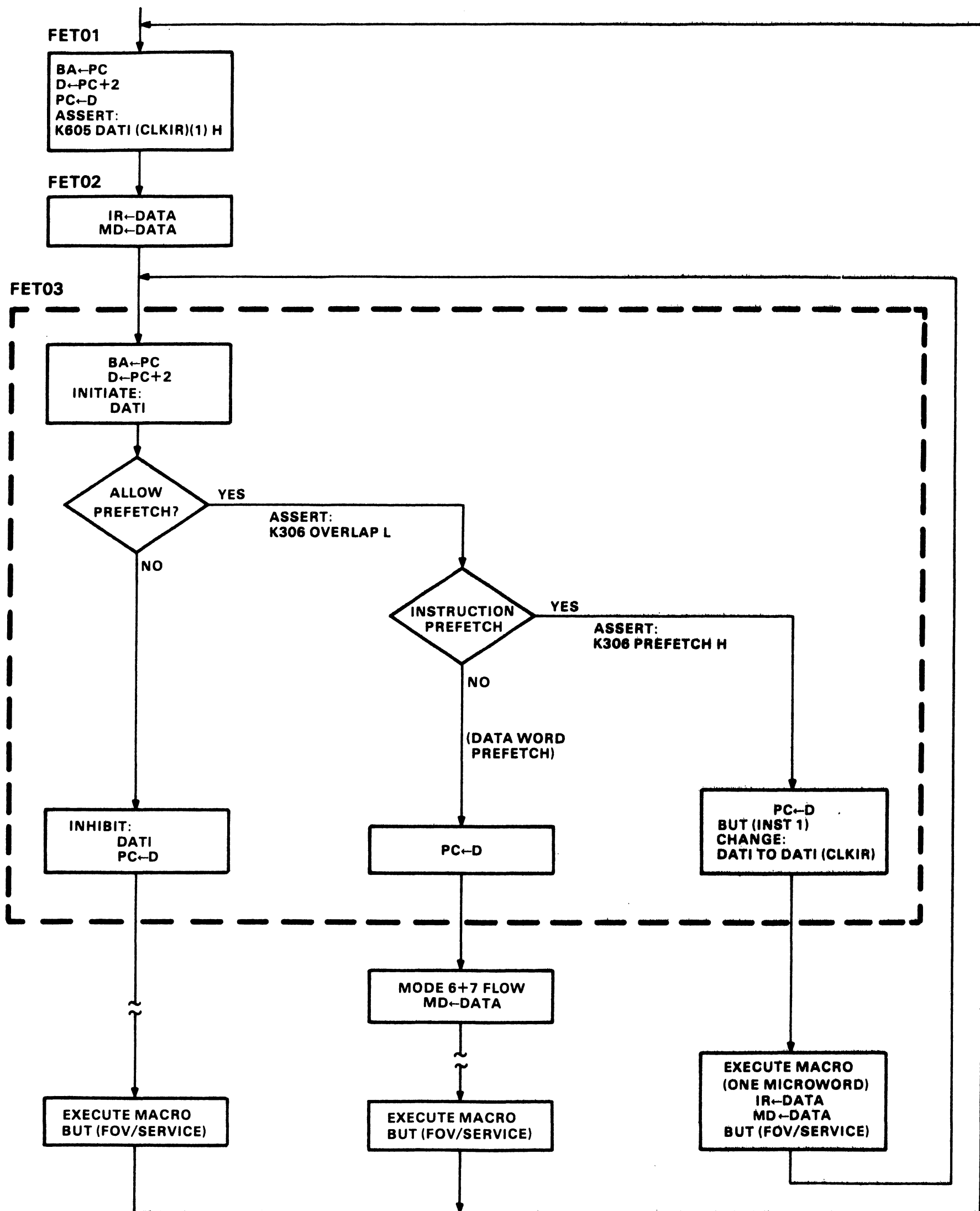
NOTES

1. READ THROUGH THE DMUX AND CSP AS PART OF THE LOGGING ROUTINE TO INDICATE THE MACHINE STATE. INFORMATION USED FOR MAINTENANCE PURPOSES.
2. MUX SELECTION ($\text{BUS } \mu <13:09> = 10000_2 = \text{BUT (FOV/SERVICE)}$) TO DETERMINE IF THE MICRO FLOW RETURNS TO FET01 OR TO FET03 AT THE END OF AN INSTRUCTION. THIS IS PART OF THE OVERLAP MECHANISM.
3. MUX SELECTION ($\text{BUS } \mu <13:09> = 10111_2 = \text{BUT (FOV PSAVE H * FPS07)}$). USED FOR RECOVERY FROM A JAM WHEN AN OVERLAP HAS OCCURRED. THE MICROCODE ADJUSTS THE PC WHEN NECESSARY.

10113

11-5595

Figure 4-21 Overlap and Prefetch Applications



11-5596

Figure 4-22 Fetch Overlap

4.8 PROCESSOR STATUS WORD (PS, PSW)

The PSW contains information regarding the processor mode (both current and previous), the processor priority, the trace bit, and the condition codes. Table 4-14 lists the bit assignments for the PSW; Figure 4-23 shows the PSW and related logic.

The PSW stores several types of data that are determined by the operation being performed. This data must be stored when the processor changes operation. This occurs, for example, every time there is an interrupt or trap. Because the PSW contents control much of the processor operation, modifications of the PSW contents are carefully controlled.

The PSW contains four fields of information: processor mode control (current and previous modes), processor priority control, trace (T) bit, and processor condition codes (CC).

Some of the PS bits control the operation of the processor, while other bits indicate the results of the last data manipulating operation.

In addition to receiving inputs from the D register via the DOUT lines, the PSW also receives inputs from the condition code control logic (Figure 4-19). In certain instances one bit of the PSW receives inputs from another PSW bit, (13:12 $\leftarrow$ 15:14), when the current mode bit replaces the previous mode bit.

Writing to the PSW is controlled either indirectly by previously set UCON control bits or, as with the case of the condition control bits, directly by microword bit 25. Input data gating is shown on Figure 4-23. Note that the three sections of the PSW can be loaded independently.

The destination of individual PSW sets are listed in Table 4-14.

4.9 FLOATING POINT STATUS (LOW BYTE) - (FPS)

The floating point status register (Figure 4-24) is used to retain and transmit to the processor via the BUS DIN lines two types of data related to floating point operations: the results of completed operations and the conditions under which forthcoming operations are to be performed.

The circuit consists of two chips: 74174, hex D-type flip-flop register; and a 74298 quad, 2-to-1 multiplexer with storage capability. In addition to data from the D register [K208 DOUT (07:00)A H], the inputs also include the condition code bits N, Z, and V from the floating point module. (There is no carry bit input; the C bit can be set only by K208 DOUT00AH.) FPS timing is shown on Figure 4-25.

The FPS high byte is stored at location 16 of ASPHI in the data path module.

Table 4-14 PSW Bit Assignments

Bit	Name	Utilization	Destination (See Note)
15-14	Current Mode	Specifies the current processor mode as follows: 1. When PS(15:14)=00, the processor is in kernel mode; all operations are legal. 2. When PS(15:14)=11, the processor is in user mode; HALT, RESET, and SPL instructions are illegal; USER address space is used if Memory Management is enabled.	Microbranch multiplexer (K308), KJ control (K312), KT decode ROM (K312)
13-12	Previous Mode	Specifies the processor mode prior to the last trap, interrupt, or loading of the PS.	MF same stack (K312), KT decode ROM (K312)
11-08 07-05	Unused Priority	Unused Sets the processor priority; this priority determines which levels of programmed and external device interrupt requests are honored.	Bus request (BR) compare (K703)
04	Trace	When PS04=1, the processor traps to the trace trap vector address after each instruction execution facility is used for debugging purposes.	BUT multiple multiplexer (K210)
Condition Codes			
03	N	This bit is set when the result of the last data manipulation is negative.	BUT multiple multiplexer (K210), Instruction branch (K307)
02	Z	This bit is set when the result of the last data manipulation is 0.	Instruction branch (K307)
01	V	This bit is set when the result of the last data manipulation is incorrect because of an arithmetic overflow.	Instruction branch (K307)
00	C	This bit is set when a carry occurs during data manipulation.	CINMUX (K313), Instruction branch (K307), D(c) MUX (K404)

Note: All PSW bits can be transmitted to the CSP (normally register 6) via the BUS DIN MUX (K210), BUS DIN, and the D MUX (K515).

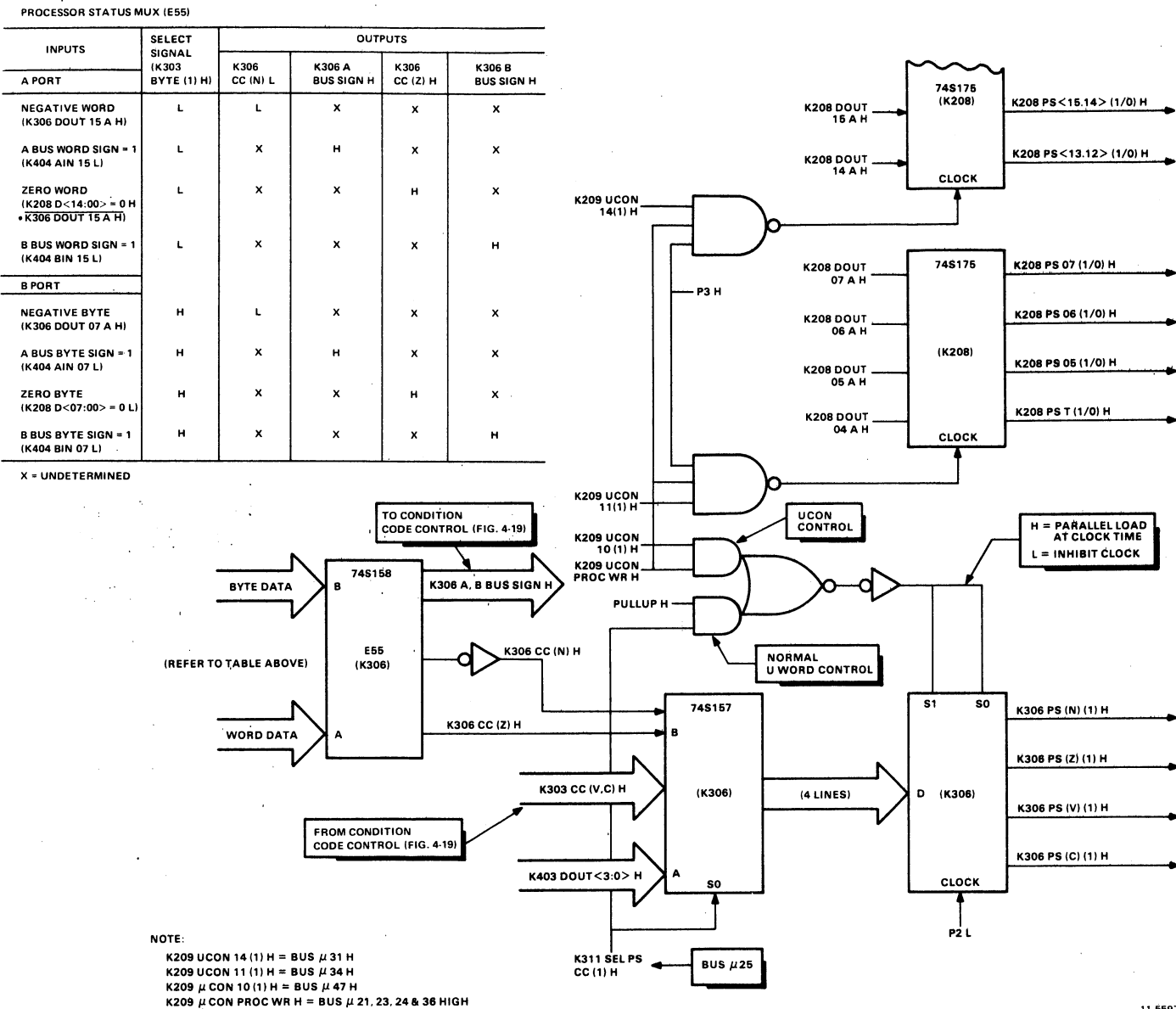
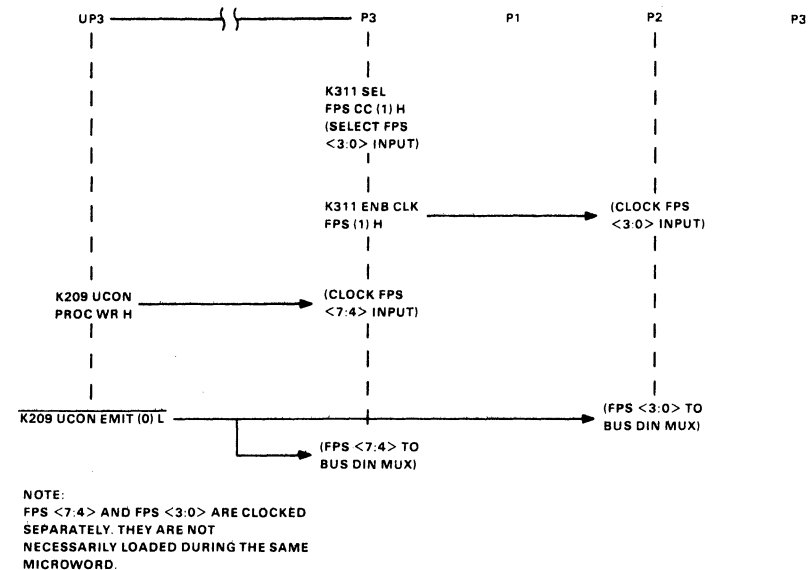


Figure 4-23 Processor Status Word



11-5599

Figure 4-25 FPS Timing (Low Byte)

4.10 KT DECODE AND CONTROL

The KT decode logic, consisting of a type 6202 ROM chip and part of a 74S153 4-to-1 multiplexer chip, is shown on drawing K312 and Figure 4-18. Four signals are produced by this circuit: K312 KT DST ADRS 3 L, K312 KT SRC ADRS 3 L, K312 USER MODE H, and K312 MF SAME STACK H. The first two signals are ORed with K310 U (FLTPT ADRS) L from the auxiliary control store to assert K312 DST ADRS 3 H and K312 SRC ADRS 3 H.

K312 DST ADRS 3 H, when high, causes A and/or B scratchpad locations 10 through 17₈ (K406) to be accessed if the destination field (IR 02:00) is selected (ASEL=2, BSEL=0) as the SP address. These scratchpad locations are dedicated primarily to floating point operations, i.e., floating point accumulator and floating point destination field. The user stack pointer is also found here (location 16).

K312 SRC ADRS 3 H, when asserted, allows the same SP locations (10-17₈) to be accessed if the source field (IR 08:06) is selected (ASEL=3, BSEL=1) as the SP address (K406). [However, if a floating point instruction is decoded (K304 FLTPT INSTR H), or IR08=0 (K406), only SP locations 10 through 13 can be addressed]. At the same time, K207 (SPADRS + 1) L, one of the active BUTs, can be used to address the odd SP location when two SP locations are used together as a 32-bit register. For example, if locations 10 and 11 function as a 32-bit register, K207 (SPADRS + 1) L will access location 11 only. Note that the "high" half of the register must have the odd address. Otherwise, the same SP location will be accessed twice.

Table 4-15 lists the input conditions that result in the assertion of the four outputs (including the signals described above) from the KT decode ROM.

The third signal produced by the KT decode logic, K312 USER MODE H, defines the processor mode (refer to Table 4-14). The signal is developed from one of two inputs from the KT decode ROM, the state of K208 PS (15, 14) (1) H, or negated by a ground input. Table 4-16 lists the four inputs to the user mode multiplexer ports, the selection signal configurations, and the resultant outputs.

K312 USER MODE H is clocked into a microcontrol register (shown on K311) at P3 time. The output of this register is then clocked into the adjacent KT control register by K311 CLK KTC (1) H (microword bit 26) at P1 time. Two signals are produced: K311 LED (USER) H, which lights a LED on the console; and K311 USER MODE (1) H, which is an input to MMR0 (K505). From there it is available (via the BUS DIN MUX, K508, and the DMUX, K515) to the processor for diagnostic purposes (Figure 2-18).

K312 MF SAME STACK H is used only with an MFPI instruction when accessing the stack and the previous and current modes are the same. The signal, input to a microbranch 1 multiplexer, causes a decrement of 2 to the address to be pushed onto the stack.

The origin of K312 KJ (1) H is the set output of the KJ control flip-flop. The set input is high when the processor is in the kernel mode [K208 (15, 14) (1) H], with K311 ENB KJ (1) H asserted [K310 U(ENB KJ) H high], and with one of three low inputs to an OR gate. The OR gate inputs are: K304 JSR/TRAP L, decoded in the preliminary decode logic (Table 4-5); destination register 6 with destination mode 4 or 6 specified (popping off the stack); or destination register 6 with destination mode 1 or 2 specified (pushing on the stack). The microcode gates off destination mode 0. K312 KJ (1) H, used to allow the assertion of a red or yellow zone error (Paragraph 2.2.5) in the cache module (K506) or a red zone error, if an internal address is decoded (K705), inhibits access to the lower 16 addresses (trap vectors, etc.) except in modes 0, 3, 5, or 7.

4.11 MICROADDRESS MATCH AND MICROBREAK (U BREAK)

The microbreak register (drawing K311 and Figure 4-26) can be preloaded with an 11-bit microprogram address that is then compared (for maintenance or diagnostic purposes) with the address of each microinstruction that is executed. When a match is detected the circuit asserts two signals: K311 NUA MATCH PULSE L, which is connected to backplane pin B03M2 (only) where it can be used as an oscilloscope sync pulse; and K311 U BREAK H, which is enabled by K210 FLAG 8 (1) H, the microbreak enable bit.

To load the microbreak register, load address 777770 and deposit the desired NUA. (This is a write only register.)

A separate register is used by the load floating point microbreak register, LDUB, if the hot floating point option is installed.

The flag 8 microbreak enable signal, K210 FLAG 8 (1) H, is derived from bit 15 of the long term flag register. The MED instruction (76600), with an op code of 344, is used to set bit 15; RO should have bit 15 set. This is done prior to the start of a microbreak operation.

Table 4-15 KT Decode ROM (E47)

Input (See Notes)										Output (See Notes)			
	A8	A7	A6	A5	A4	A3	A2	A1	A0	M3	M2	M1	M0
$\overline{\text{MFPI}} \bullet \text{PREVIOUS MODE} = \text{USER}$	X	X	H	H	X	X	X	X	X	H			
$\overline{\text{MFPI}} \bullet \text{CURRENT MODE} = \text{USER}$	X	X	L	X	H	X	X	X	X	H			
$\overline{\text{MTPI}} \bullet \text{PREVIOUS MODE} = \text{USER}$	X	H	X	H	X	X	X	X	X		H		
$\overline{\text{MTPI}} \bullet \text{CURRENT MODE} = \text{USER}$	X	L	X	X	H	X	X	X	X		H		
$\overline{\text{MTPI}} \bullet \text{DST REG 6} \bullet \text{CURRENT MODE} = \text{USER}$	X	L	X	X	H	L	X	X	X			L	
$\overline{\text{MTPI}} \bullet \text{DST MODE 0} \bullet \text{CURRENT MODE} = \text{USER} \bullet \text{DST REG 6}$	X	H	X	X	H	L	X	L	X			L	
$\text{FLOATING POINT INSTR} \bullet \text{DST REG 6} \bullet \text{CURRENT MODE} = \text{USER}$	L	X	X	X	H	L	X	X	X			L	
$\text{FLOATING POINT INSTR} \bullet \text{MTPI} \bullet \text{DST REG 6} \bullet \text{DST MODE 0} \bullet \text{PREVIOUS MODE} = \text{USER}$	H	H	X	H	X	L	X	H	X			L	
$\overline{\text{MFPI}} \bullet \text{SRC REG 6} \bullet \text{CURRENT MODE} = \text{USER}$	X	X	L	X	H	X	H	X	X				L
$\overline{\text{MFPI}} \bullet \text{SRC MODE 0} \bullet \text{SRC REG 6} \bullet \text{CURRENT MODE} = \text{USER}$	X	X	H	X	H	X	H	X	L				L
$\text{FLOATING POINT INSTR} \bullet \text{SRC REG 6} \bullet \text{CURRENT MODE} = \text{USER}$	L	X	X	X	H	X	H	X	X				L
$\overline{\text{MFPI}} \bullet \text{FLOATING POINT INSTR} \bullet \text{SRC MODE 0} \bullet \text{SRC REG 6} \bullet \text{PREVIOUS MODE} = \text{USER}$	H	X	H	H	X	X	H	X	H				L

Note: Inputs and Outputs

A8: K304 FLTPT INSTR L
 A7: K210 FLAG 1 (1) H (MTPI)
 A6: K210 FLAG 2 (1) H (MFPI)
 A5: K208 PS (13, 12) (1) H (Previous mode)
 A4: K208 PS (15, 14) (1) H (Current mode)
 A3: K312 DST REG = 6L

A2: K312 SRC REG = 6H
 A1: K304 DM = 0H
 A0: K304 SM = 0H
 M3: Outputs to user
 M2: MODE MUX (K312)
 M1: K312 KTDST ADRS 3L
 M0: K312 KTSRC ADRS 3L

Table 4-16 User Mode Multiplexer (E28)

Selection Input		Port Selected	Input	Input Description	Output
K310 U (KT SEL1) H	K310 U (KT SEL0) H				
0	0	A {	KT DECODE ROM OUTPUT M3	(MFPI • Previous mode=User) + (MFPI • Current mode=User)	K312 USER MODE H IF INPUT HIGH
0	1	B {	KT DECODE ROM OUTPUT M2	(MTPI • Previous mode=User) + (MTPI • Previous mode=User)	K312 USER MODE H IF INPUT HIGH
1	0	C	GROUND	Set to kernel mode	K312 USER MODE H
1	1	D {	K208 PS (15, 14) (1) H	Set to user mode if asserted or to kernel mode if not asserted	K312 USER MODE H IF INPUT HIGH



5

When an address comparison is made by the three 74S85 comparator chips in the microaddress match and microbreak circuit, the microprocessor jams at the selected microaddress. The microcode flow then proceeds to the logging routine during which bit 0 of the log jam register is set from BUS DIN 00 H. K210 FLAG 08 (0) H is reset and any additional assertions of K210 U BREAK H are inhibited. One of three paths is now entered (the original flow cannot be resumed once it is interrupted by the microbreak).

1. The microcode switches to console operation via the jam routine. The console can then be used to examine the conditions present in the internal registers at the time the address match occurred.
2. If the microbreak macrotrap enable bit (WHAMI bit 9) is set, the trap flow is entered. The processor traps through location 4 and sets bit 8 (U BREAK TRAP) in the CPU error register (address 777766). The correct way to set WHAMI bit 9 is to first read the register with a MED instruction that has an op code of 22. Then set bit 9 in the data that was read and store the results in R0. This data can then be written into WHAMI by using a MED instruction with an op code of 222.

NOTE

Both the flag and WHAMI registers can be loaded via the console but they will be cleared when the machine is started. However, if the PC is loaded with the desired starting address and the CONTinue switch pressed, the registers will not change.

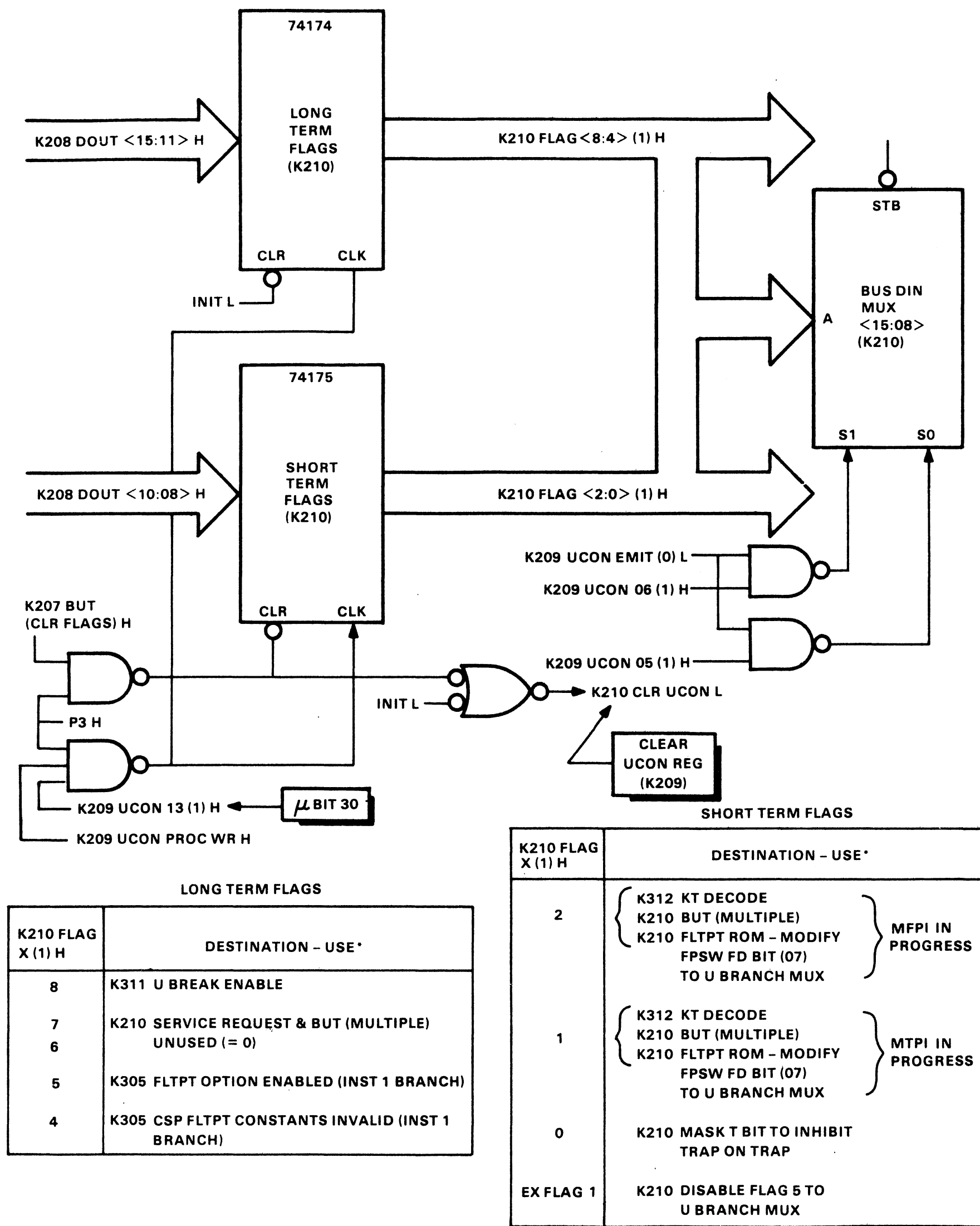
3. Control is transferred to WCS if WHAMI bit 9 is set and the WCS module is present (installed) when an address match occurs.

4.12 SHORT AND LONG TERM FLAGS

The short and long term flag registers are used to indicate certain base machine conditions and to define a number of controls for processor operation. Figure 4-27 shows the two registers and lists the destination and use for each of the nine flag bits. Under UCON ← UC control the flag bits are also output to high byte of the BUS DIN. [K209 UCON 06 (1) H (microword bit 39) and K209 UCON 05 (1) H (microword 38) both equal 1 and select the BUS DIN MUX A inputs].

The long and short term flag bits are loaded at the same time (P3) from DOUT under UCON ← UC control. The long term flags remain in the condition they are set until changed or INIT L clears the register. (The INIT flow sets the flag 4 and flag 5 bits). The short term flags are treated differently; they are cleared by K207 BUT (CLR FLAGS) H each time a macrofetch is performed (FET02).

CSP registers 0 through 13 are multifunction registers. They can be used to hold floating point constants, LOG information (viz., LOG WHAMI and LOG JAM), or console constants. K210 FLAG 4 (1) H is asserted when these registers contain anything other than floating point data. This indicates that the floating point constants are invalid.



* FLAG <8:4>, <2:0> ALSO INPUT TO THE BUS DIN MUX

11-5601

Figure 4-27 Long and Short Term Flags

4.13 EMIT REGISTER

The emit register (Figure 4-28) provides the means for entering 16-bit constants into the data path from the microword. The input to the register is the microword EMIT field, BUS U(47:44, 41:30), which is clocked into the register at UP3 time. The constant is then selected as the BUS DIN MUX input by K209 UCON 06 (1) H and K209 UCON 05 (1) H [BUS U(39:38) H] both equal to 0.

The EMIT field is also the default input to BUS DIN. The BUS DIN multiplexer input selection is controlled by the states of UCON ← UC bits 5 and 6; K209 UCON SELECT (1) H is normally high. However, during a macrofetch K207 BUT (CLR FLAGS) H causes the assertion of K210 CLR UCON L (Figure 4-35), which clears the UCON register. This causes K209 UCON SELECT (1) H to go low and disable the two BUS DIN MUX selection gates. The high outputs from these NAND gates select K209 EMIT (15:00) (1) H as the input to the BUS DIN MUX.

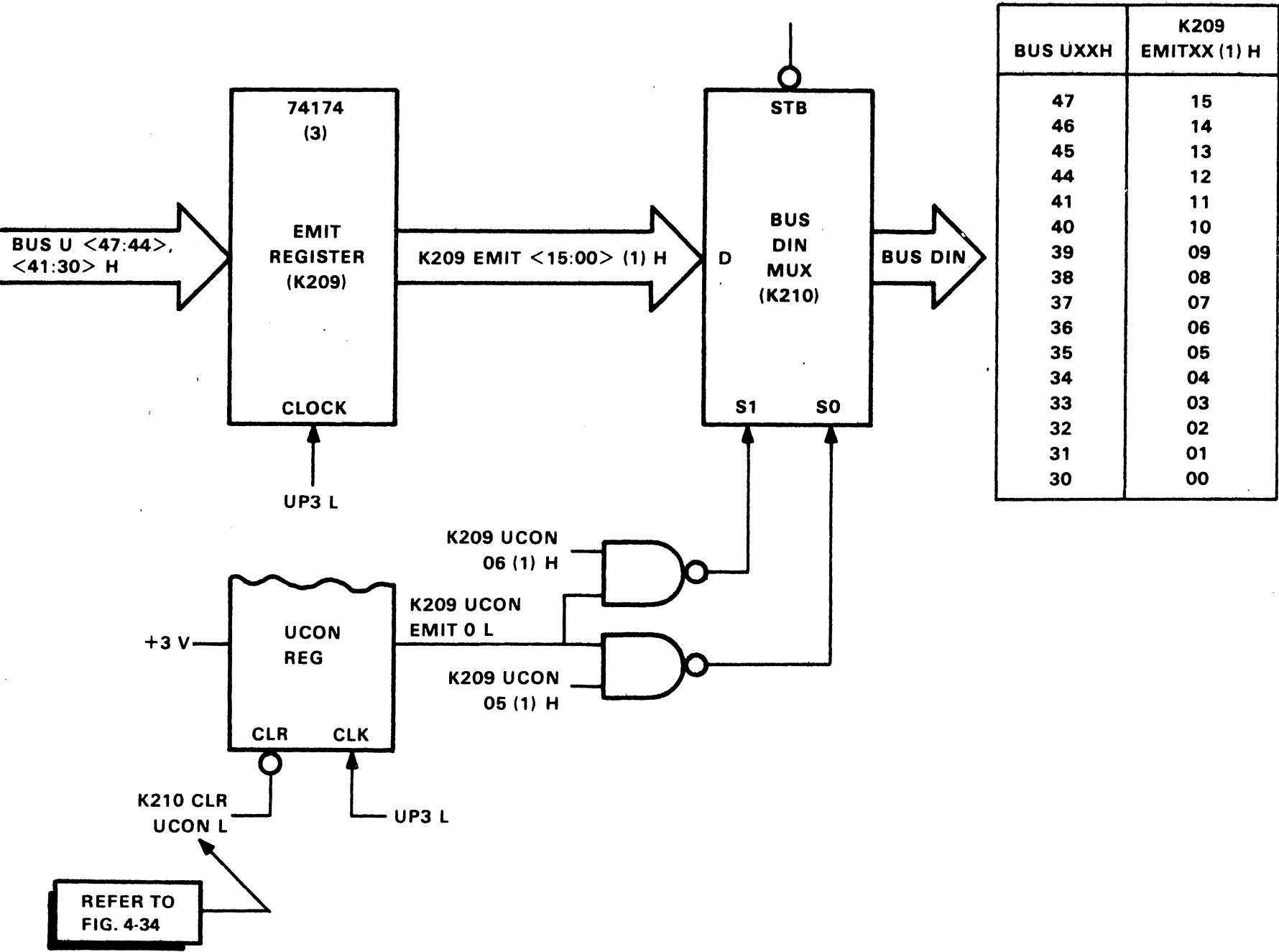


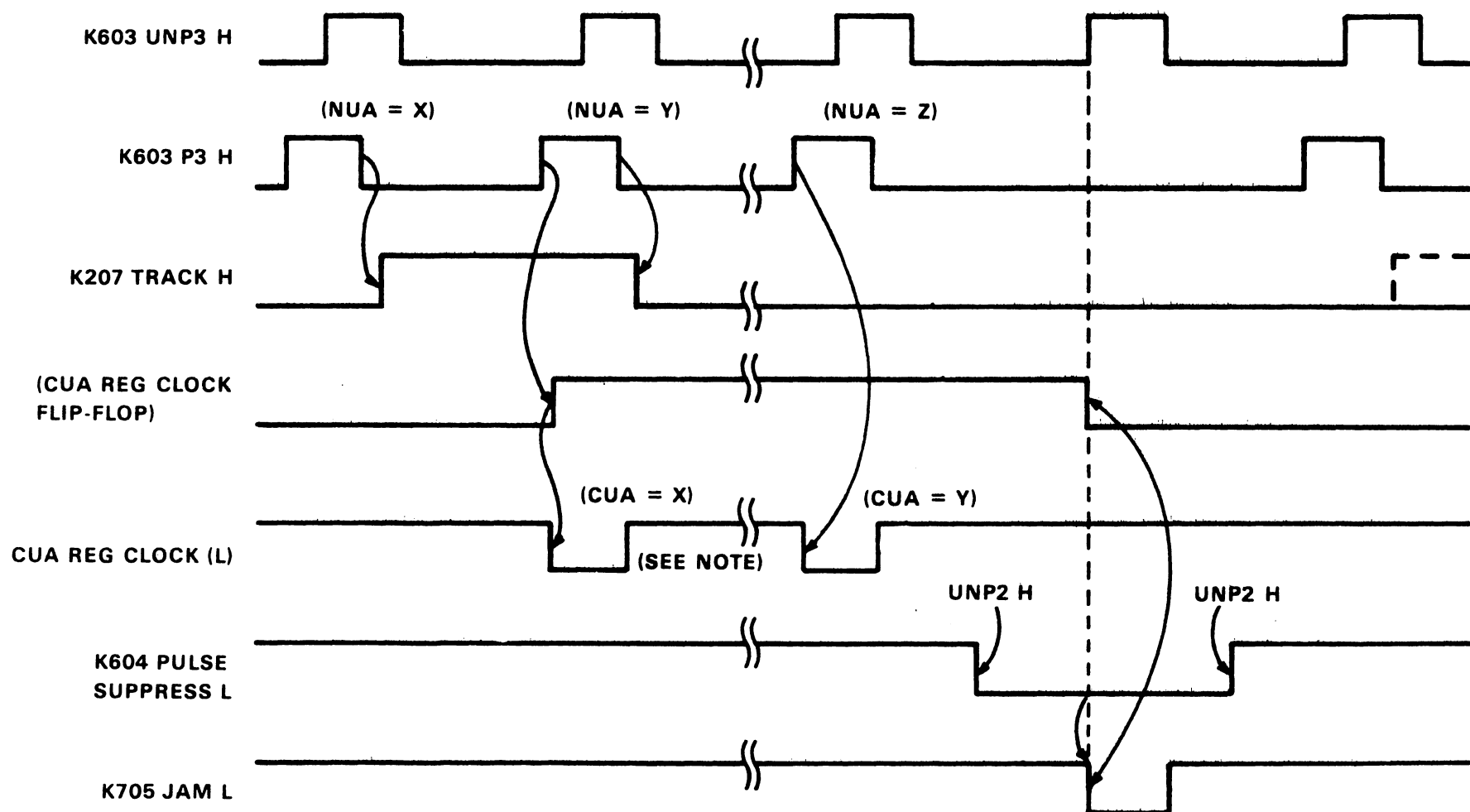
Figure 4-28 Emit Register

11-5602

4.14 CURRENT MICROADDRESS REGISTER (CUA)

The CUA register is used to store the address of the current microaddress; the register is said to "track" the microcode flow. A new address is clocked into the CUA register each time the NUA is developed for another microinstruction. In the event of a jam, the tracking halts; at this time, the register contains the address of the microinstruction that caused the jam. This address is then available, via BUS DIN, to the processor for diagnostic purposes or for normal returns such as internal address references.

Refer to drawing K207 and the timing diagram shown in Figure 4-29. Tracking is not initiated until the active BUT K207 BUT (TRACK) H is asserted [BUS U(13:09) H = 11001₂]. At the following P3 time, a flip-flop, consisting of two cross-coupled NAND gates at the CUA clock input, is set. K207 BUT TRACK H is normally negated at this time. With the flip-flop set, a 12-bit NUA configuration is clocked into the CUA register at the trailing edge of each P3 pulse. (Note that the address in the register is the current microaddress and that this address lags the NUA by one microcycle).

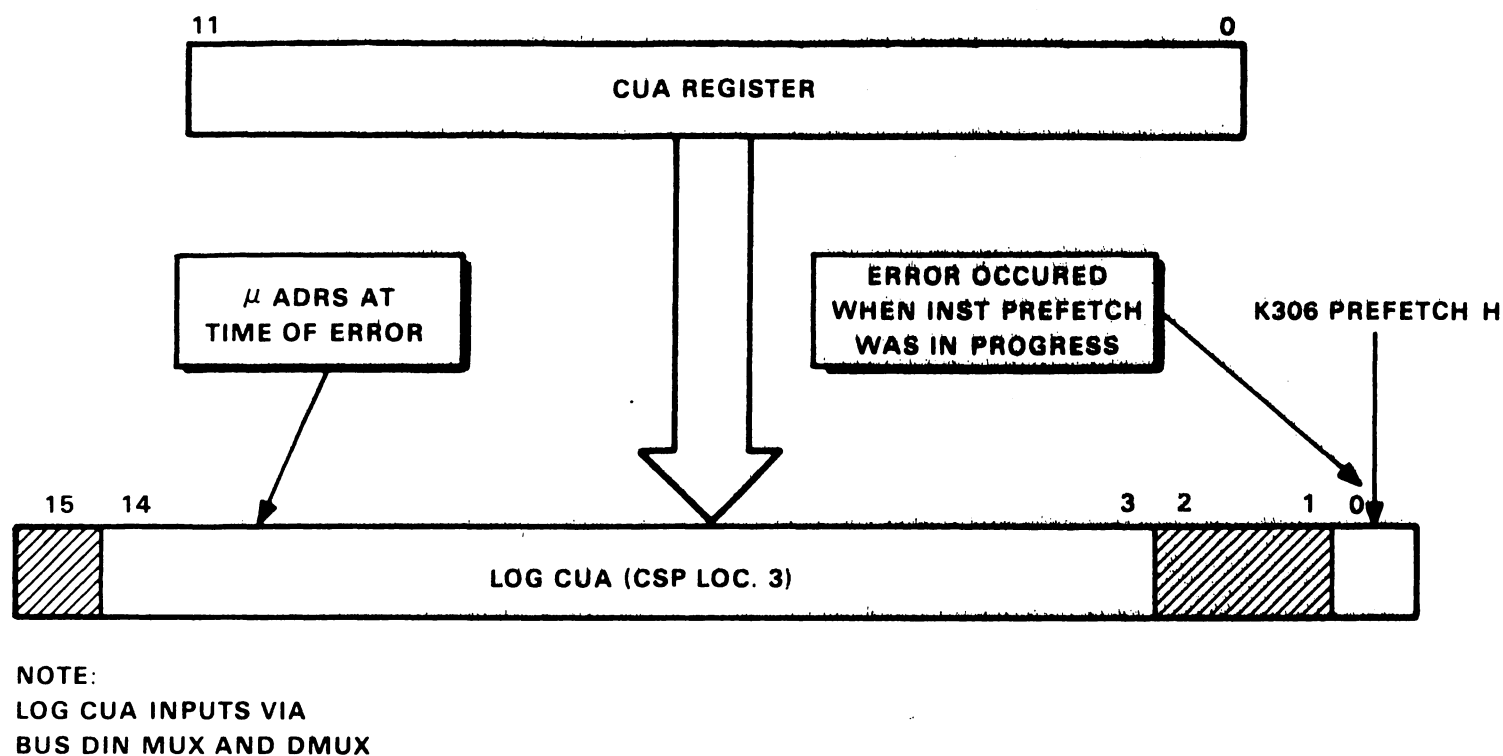


NOTE:
THE FIRST REGISTER CLOCK AFTER K207 TRACK H IS ASSERTED
MAY OR MAY NOT OCCUR. THE NEXT CLOCK WILL OCCUR.

11-5603

Figure 4-29 CUA Register Timing

The tracking continues until a jam condition occurs, e.g., internal address, odd address error, red zone, or memory parity error (Paragraph 1.7.3). At that time, K604 PULSE SUPPRESS L and K705 JAM L are asserted. The latter signal resets the flip-flop that gates the CUA register clock pulse [additional clock pulses are inhibited until K207 BUT (TRACK) H is asserted again]. The address of the questionable microinstruction, now locked in the CUA register, is written into the log CUA register (CSP location 3) during the subsequent error logging routine (Figure 4-30). In the Log CUA, the address is available for inspection at the console or as the input to the return register if the microcode is to be re-entered at the questionable point.



11-5604

Figure 4-30 Log CUA Register

4.15 BUS DIN MULTIPLEXER

The BUS DIN multiplexer is used to select the data from the processor control logic that is placed on the BUS DIN for transfer to various CSP registers. The multiplexer consists of 8, 4-to-1, type 74LS253 tristate multiplexer chips shown on drawing K210. Figure 4-31 depicts an abbreviated version of the multiplexer logic. Inputs to the four multiplexer ports are from the floating point status, flag, processor status, emit, and CUA registers. K306 PREFETCH H from the overlap decode logic is another input.

Multiplexer controls, both selection and enable (STB), are in the form of signals from the UCON register. K209 UCON (06:05) (1) H selects which input is placed on BUS DIN (refer to the table in Figure 4-31). A frequently used input is the data from the emit register (Paragraph 4.13), the source of constants that are inserted into the data path. This input at port D is the default input; the data from the emit register is automatically selected when K209 UCON SELECT (1) H goes low and both S1 and S0 select inputs go high. This occurs during each macrofetch (FET02), when K207 BUT(CLR FLAGS) H asserts K210 CLR UCON L, which clears the UCON register (K209).

If data from any of the other processor control sources is required, the UCON signals are set up during a preceding microinstruction.

BUS DIN is also used to convey Unibus data when a Unibus operation is performed (Figure 4-31). At this time, the circuits that are not used in the transfer of Unibus data must be disconnected from the BUS DIN. These circuits, tristate multiplexers in processor control, KT cache, and bus control are disabled by K607 INH UCON (BUS XFER) L. The assertion of this signal disables the enable (ENB or STB) inputs to these multiplexers. In the absence of an enable input, the multiplexer outputs present a high impedance to (and are thus disconnected from) the BUS DIN lines.

4.16 MICROCONTROL (UCON) INTERFACE

The UCON interface is an interface contained within the processor that is used for intraprocessor transfers of control, flag, and secondary information (Figure 4-32). These transfers are directed by control signals that are produced from a portion of the microword called the UCON word. The UCON word, which is further broken down into four functional bit groups (Figure 4-33), is stored in the UCON register (Figure 4-34) at a time specified by the microcode. Hardware associated with the UCON register produces additional control signals. Once the control signals are stored in the UCON logic, this part of the microword can be dedicated to an alternative control function(s) during the execution of subsequent microinstructions. The UCON control signals continue to perform their particular roles during this period – until they are specifically changed or the UCON register is cleared. The register is cleared by K207 BUT (CLR FLAGS) H, which is asserted (in addition to other times) during the execution of FET02 when a macroinstruction is fetched.

The UCON hardware provides for the transfer of macrodata, control information, or flag information. Two types of transfers can occur;

- Data is loaded into the UCON interface
- Data is gated onto BUS DIN from the interface

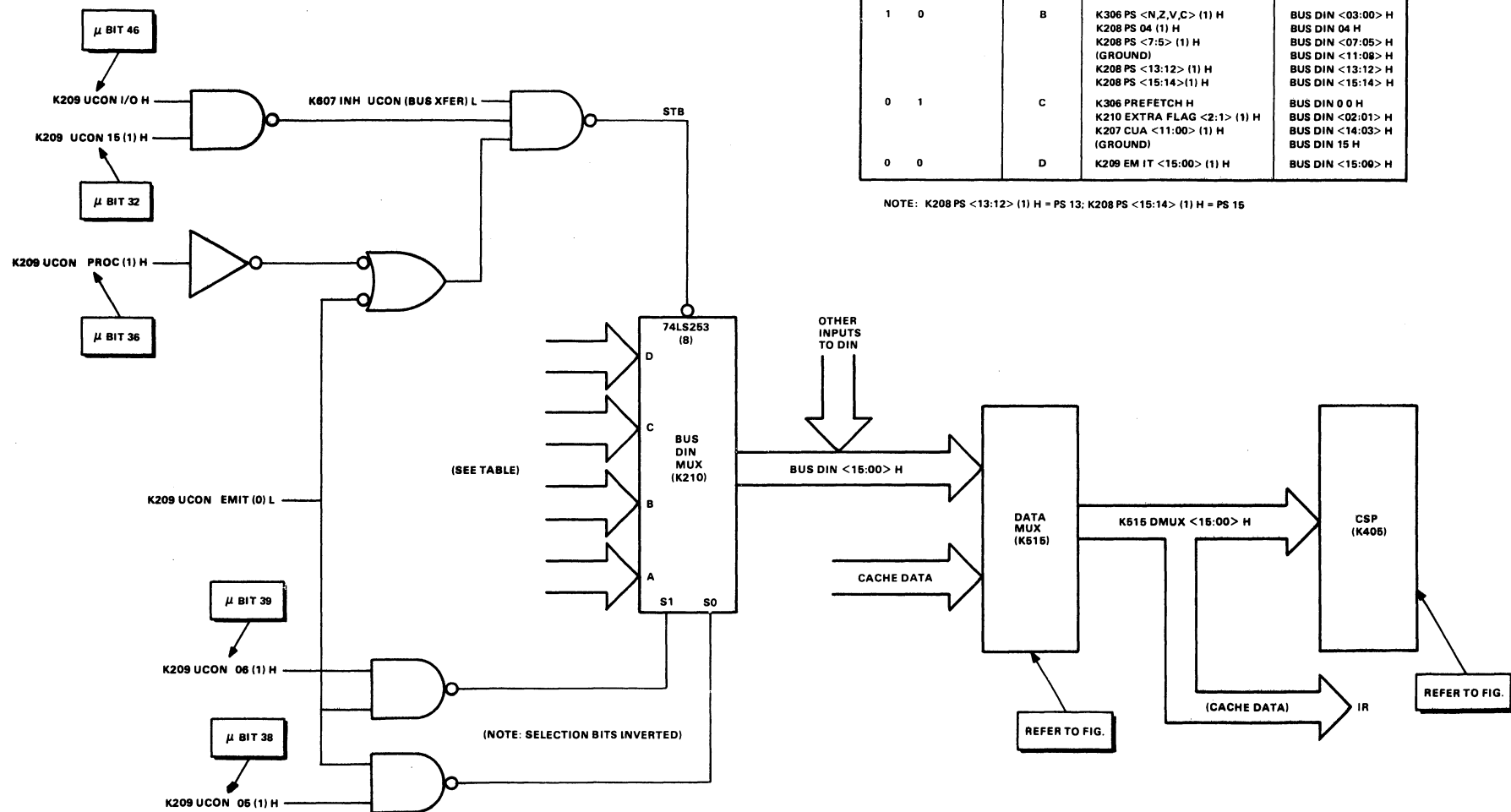
Three different types of signals are necessary for each of the two types of transfers. The signals used for loading data into the UCON interface are:

- Clocking pulse (P3, P2)
- Interface selection [e.g., K209 UCON KT (1) H]
- Transfer to interface [K209 UCON XFER (1) H]

The signals used to gate data onto BUS DIN from the interface are:

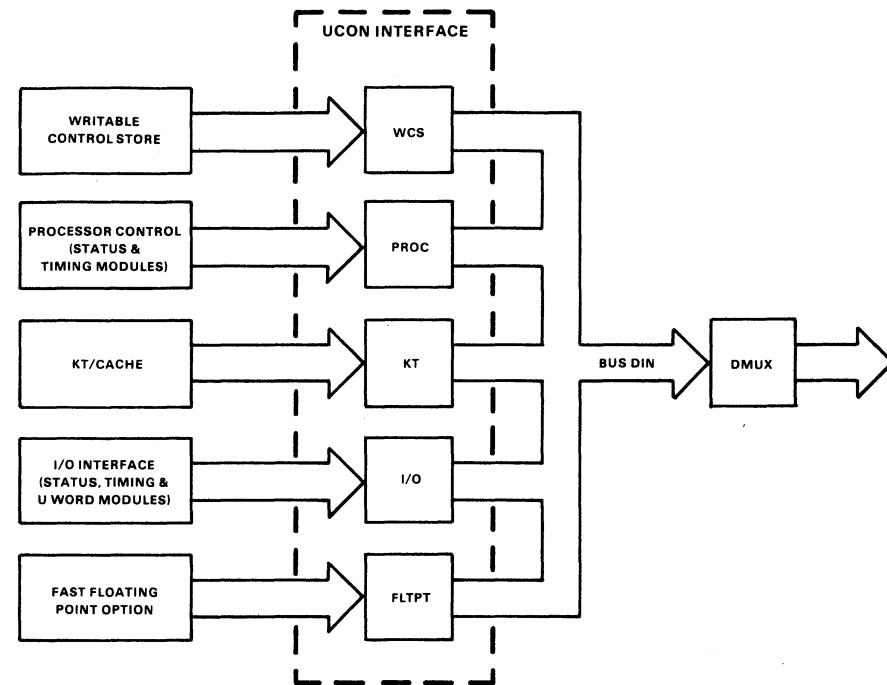
- Bus control enable [K209 UCON SELECT (1) H]
- Interface selection [e.g., K209 UCON Proc (1) H]
- Possible disable by another UCON signal [e.g., K209 UCON I/O) (1) H • K209 UCON 15 (1) H disables BUS DIN ← BUS DIN MUX on K210]

Frequently used data inputs to BUS DIN are the microword constants from the emit register. The register contents are changed each time a new microword is read from the control store (Paragraph 4.13). This is the default input to BUS DIN; it is not necessary to set-up the UCON register in order to effect the data transfer. When the UCON register is cleared by K207 BUT (CLR FLAGS) H, K209 UCON SELECT (1) H (now low) inhibits the port select gates to the BUS DIN multiplexer. The resultant high select inputs choose the multiplexer port connected to the emit register.



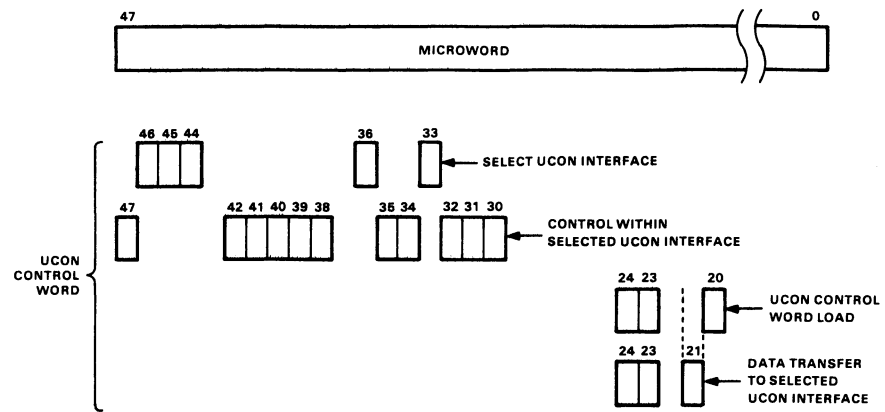
11 - 5605

Figure 4-31 BUS DIN Multiplexer



11-5606

Figure 4-32 UCON Interface



11-5607

Figure 4-33 UCON Control Word

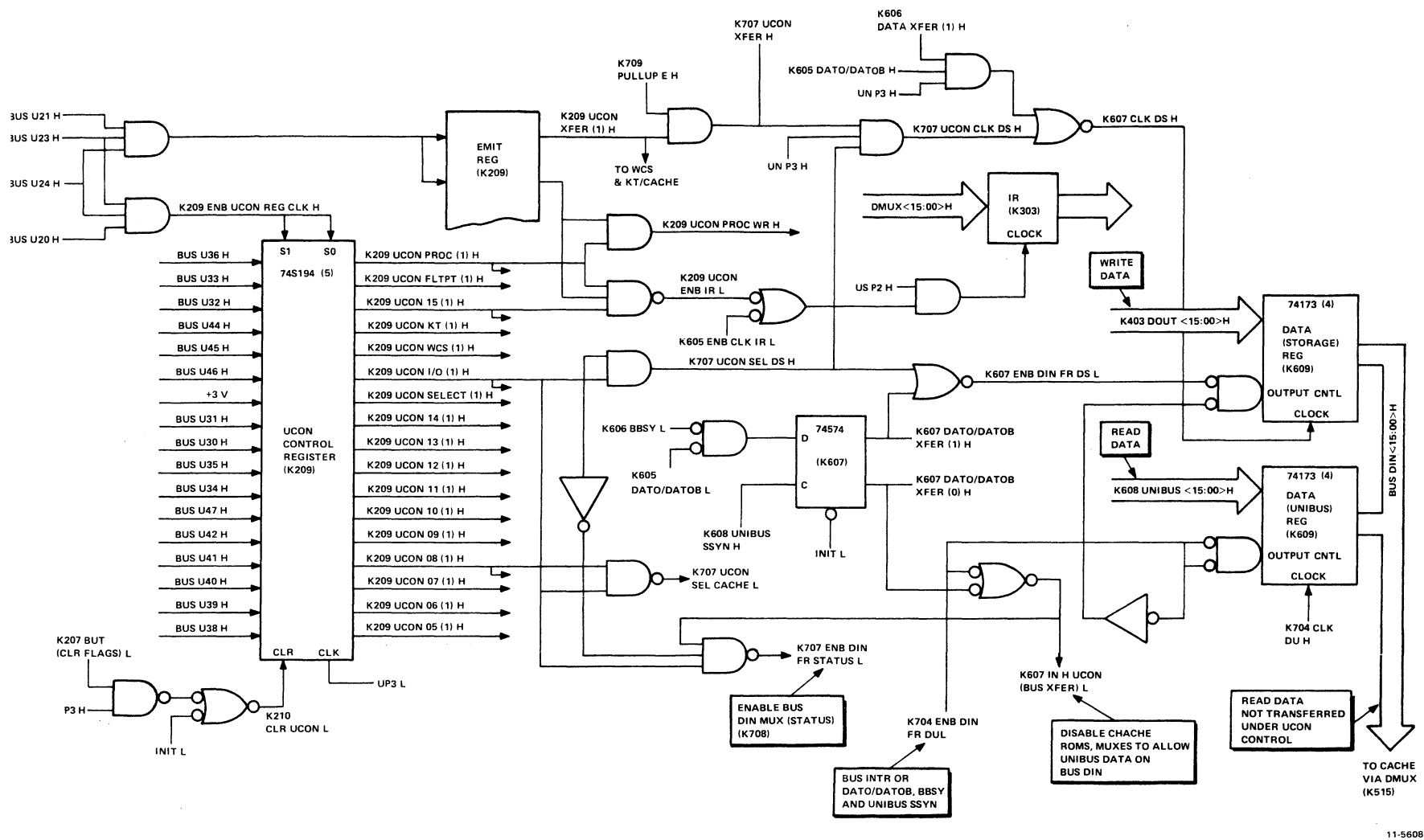


Figure 4-34 UCON Signal Generation

The UCON register, shown on drawing K209, consists of five, type 74S194, shift register chips that are loaded directly from the microword control store output bus. Inputs consist of 16 microword bits that are converted to UCON control signals (refer to Figure 4-34). The input is loaded at UP3 time if the load enable signal, K209 ENB UCON REG CLK H, is asserted (BUS U20, U23, and U24 all = 1). The output UCON signals, listed in Table 4-17, determine the source of the data that is placed on the BUS DIN for transfer to the CSP registers or cache.

Load control (BUS U20, U23, and U24) can be asserted in one microword or across several microwords. Each UCON word consists of an interface selection (Figure 4-33), which is normally singular, and specific control bits that exercise control within the selected interface. The UCON register provides two classes of signals to perform these functions. For example, K209 UCON KT (1) H and K209 UCON I/O (1) H select the cache and I/O interfaces, while K209 UCON 09 (1) H and K209 UCON 15 (1) H further define information transfers within the cache and I/O interfaces. (Other control signals, e.g., K209 UCON ENB IR L, are combinations of these two signal types).

Figures 4-35 and 4-36 show the distribution and application of UCON and UCON-related signals developed by the hardware (Figure 4-34).

Table 4-17 UCON Signals

Input BUS UXX H	Output K209 UCON XX (1) H	Destination(s) and Function (Note: Some functions require more than one UCON signal)
32	15	K209 – Clock IR, K210 – BUS DIN MUX Disable K707 – Console signals, data store register control
31	14	K208 – PS (15:13) Clock, K101 – WCS K505 – Inhibit relocate, K707 – Console signals
30	13	K210 – Clock flags, K507 – Determine INT ADRS type K707 – Console signals
35	12	K208 – FPS (7:4) Clock, K507 – Enable INT ADRS ROM K707 – Console signals
34	11	K208 – PS (7:4) Clock, K707 – Console signals K507 – Read Only register MUX enable
47	10	K306 – PS (N,Z,V,C) Load enable, K708 – BUS DIN MUX (status) select, K508 – BUS DATA MUX enable
42	09	K311 – U Break clock, K708 – BUS DIN MUX (status) select K507 – Read Only register MUX select
41	08	K707 – Select cache as DMUX output K514 – WR register decode input
40	07	K707 – Clear error signals, allow BG K514 – WR register decode input
39	06	K210 – BUS DIN MUX select, K508 – Bus data MUX select; K514 – WR register decode select
38	05	K707 – Clear error signals, allow BG, K506 – AMUX select
33	FLTPT	
36	PROC	K210 – BUS DIN MUX enable, K209 – Clock IR and assert PROC WRITE signal
46	I/O	K210 – BUS DIN MUX disable, K707 – Remote console CNTL register, Enable console signals, Select cache as DMUX output, Data store register control
45	WCS	WCS, ECS, DCS Modules
44	KT	K505 – Inhibit relocate, Select MMR0 input, K507, K508 – Enable cache ROMs, MUX to BUS DIN, K506 AMUX select
+3 V	SELECT(1)H	K210 – Causes BUS DIN ← EMIT (default input to BUS DIN when UCON register cleared by BUT (clear flags)

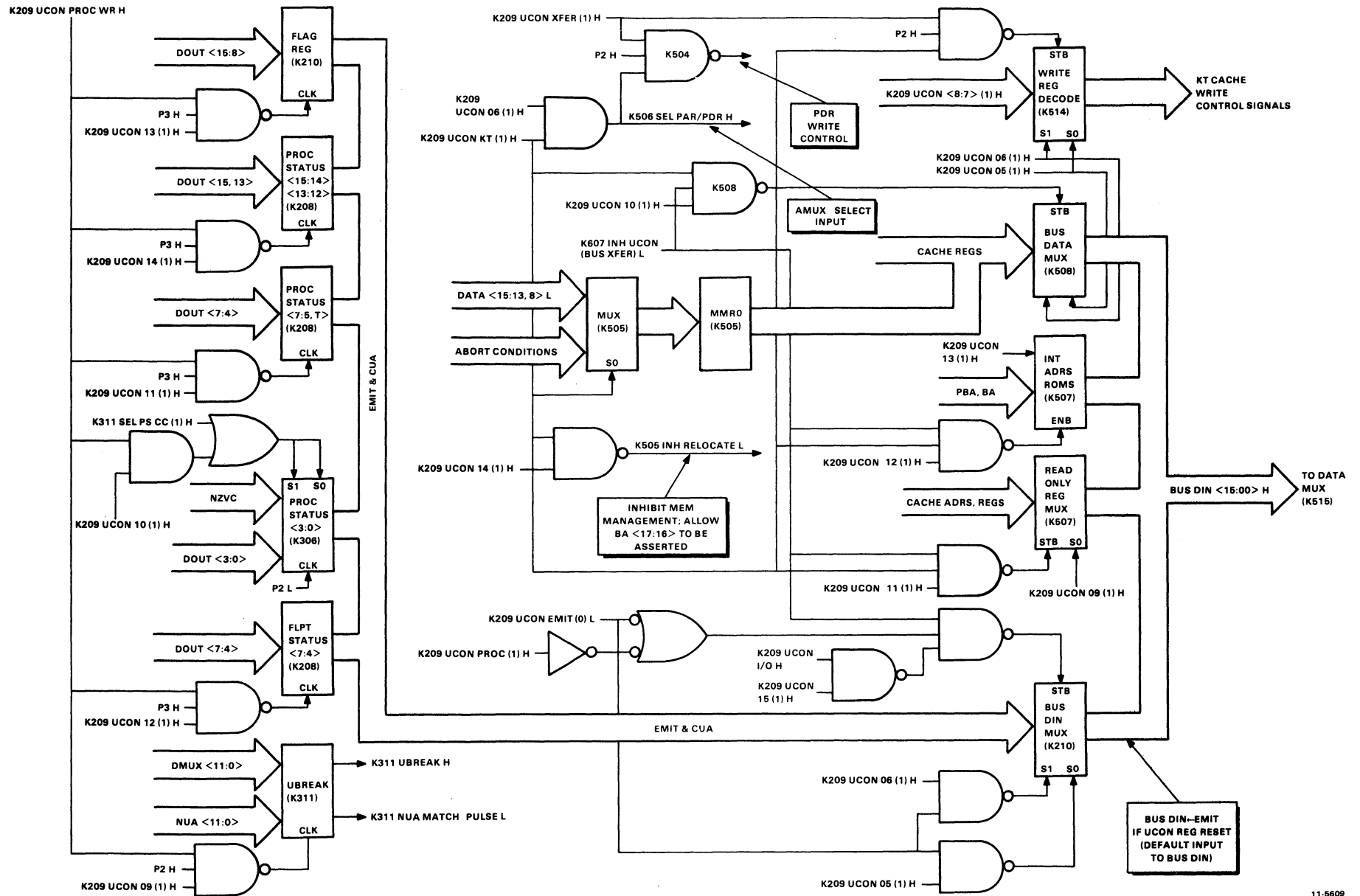


Figure 4-35 UCON Interface
(Processor Control and KT Cache)

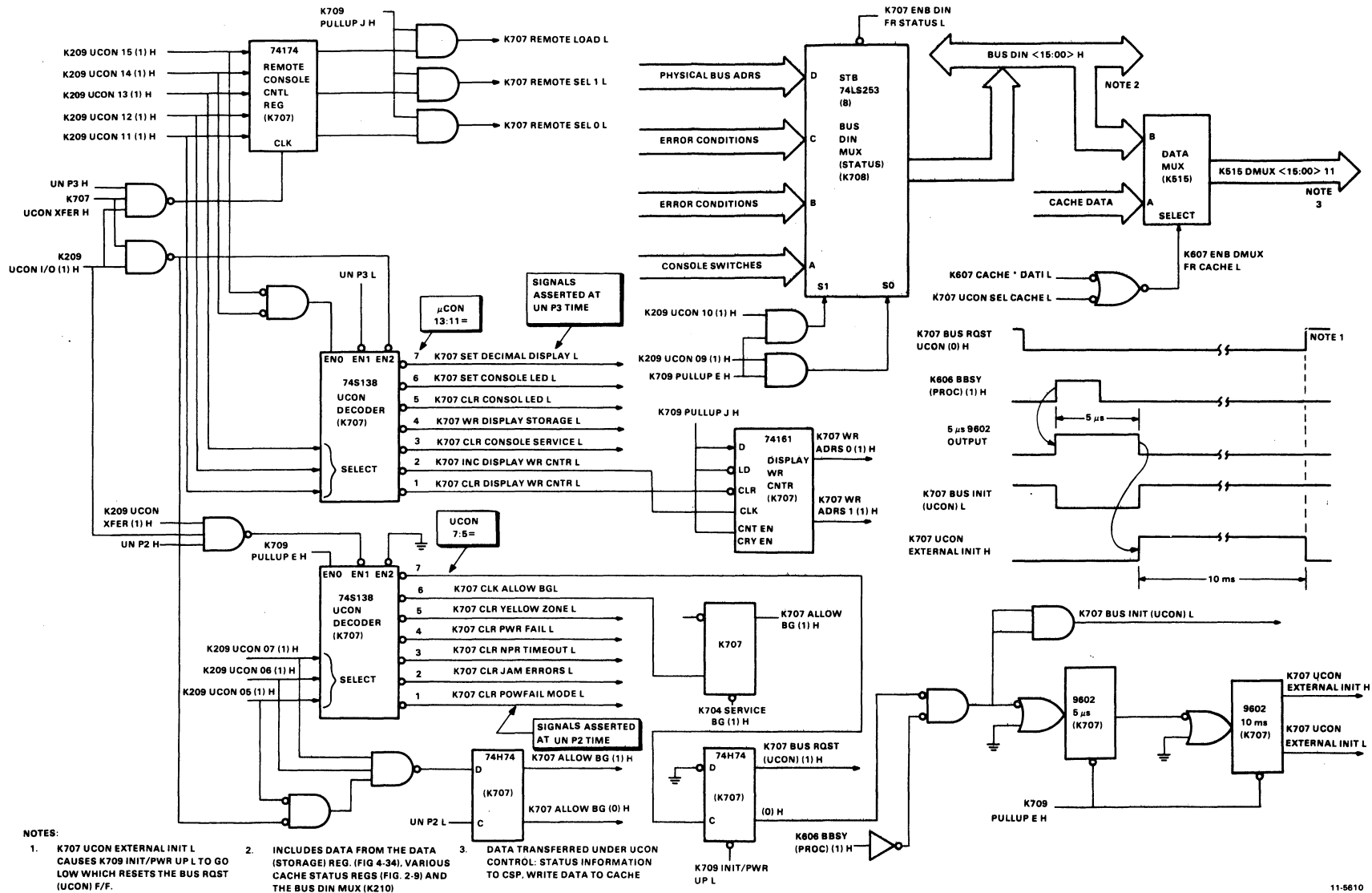


Figure 4-36 UCON Interface (Bus Control)

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

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

What features are most useful? _____

What faults or errors have you found in the manual? _____

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

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

☐ Please send me the current copy of the *Technical Documentation Catalog*, which contains information on the remainder of DIGITAL's technical documentation.

Name _____ Street _____

Title _____ City _____

Company _____ State/Country _____

Department _____ Zip _____

Additional copies of this document are available from:

Digital Equipment Corporation
444 Whitney Street
Northboro, Ma 01532
Attention: Communications Services (NR2/M15)
Customer Services Section

Order No. _____

3
4

----- **Fold Here** -----

1

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

**FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.**

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

Postage will be paid by:

**Digital Equipment Corporation
Technical Documentation Department
Maynard, Massachusetts 01754**

