MOHAWK DATA SCIENCES

SERVICE DIVISION

Q64 CPU

Theory Of Operation

M-4936
(TD-4204)

TECHNICAL MANUAL

Q64 CPU

THEORY OF OPERATION

FIRST EDITION
October, 1984

ABSTRACT

This manual describes MDS Qantel's Q64 CPU. Specific theory is integrated with a detailed description of the microword specifications and a gate-by-gate discussion of the logic.

MANUAL REVISION RECORD

TITLE: Q64 CPU THEORY OF OPERATION

FORM NO: M-4730-0100
EDITION DATE: 1084

DCN NUMBER	DATE	PAGES AFFECTED	REASON(S) FOR REVISION

FIGURES

TABLES

1.0 INTRODUCTION

1.1 GENERAL DESCRIPTION

The Q64 is a four-board CPU, utilizing bit-slice microprocessor elements and pipelined architecture. The standard Qantel instruction set is interpreted by ROM microcode residing in the CPU assembly.

The Q64 does not utilize field programmable logic arrays(FPLAs) such as found in the Q29 & Q30 CPUs. Processing has been streamlined by the Q64 through the use of a 64-bit microword and a "Fetcher" which performs fetches from memory and some decoding of macrocode.

This document is an in-depth examination of the Q64 CPU architecture. The logics cited and discussed herein are all for the revised(R) versions of the four CPU boards and the Diagnostic Panel. With the exception of the ALU-R, logic for previous releases of the boards is essentially the same, and the R board schematics may be used to troubleshoot previous versions of the boards. WCS was added to the ALU-R and the logics are substantially different. Document D32091-001 should be used in troubleshooting earlier versions of the ALU Board.

1.2 RELATED DOCUMENTS

The following is a selected list of Q64 documents available from the MDS Qantel Drafting Department:

D32158 --- Logic, ALU Board - R
D32184 --- Logic, Auxilliary Board - R
D32181 --- Logic, Memory Address Board - R
D32183 --- Logic, Memory Data Board - R
D32159 --- Logic, Diagnostic Panel - R
D44203 --- PCB Fab/Artwk ALU - R
D44447 --- PCB Fab/Artwk Aux - R
D44427 --- PCB Fab/Artwk Mem Add - R
D44437 --- PCB Fab/Artwk Mem Data - R
D44476 --- PCB Fab/Artwk Diag Panel - R
L51067 --- Q64 CPU Microcode Listing
L51065 --- IOU ROM(Auxilliary Board)
L51066 --- BCD Adder/Subtracter(Auxilliary Board)
L51068 --- Increment/Decrement Control(Memory Address Board)
L51077 --- Fetch Mode Decode(Memory Data Board)
L51078 --- Fetch Sequence Control(Memory Data Board)
L51079 --- Fetch Vector Logic(Memory Data Board)
L51080 --- Fetch Memory Control(Memory Data Board)
L51081 --- Fetch Logic(Memory Data Board)
L51082 --- Barrel Shifter(Memory Data Board)
M44204 --- ALU-R PWA/ROMS
M44501 --- Aux-R PWA/ROMS
M44499 --- Mem Add-R Bd PWA/ROMS
M44500 --- Mem Data-R PWA/ROMS
M44475 --- Diag Panel-R PWA

2.0 Q64 BASICS

This section deals with the basic architectural concepts of the Q64. The microprocessor, the Fetcher, the Memory Interface and related circuits are discussed. Block diagrams, tables and timing diagrams are used to show the relationship of the major circuit networks in the Q64 architecture.

2.1 THE 2901B MICROPROCESSING ELEMENT

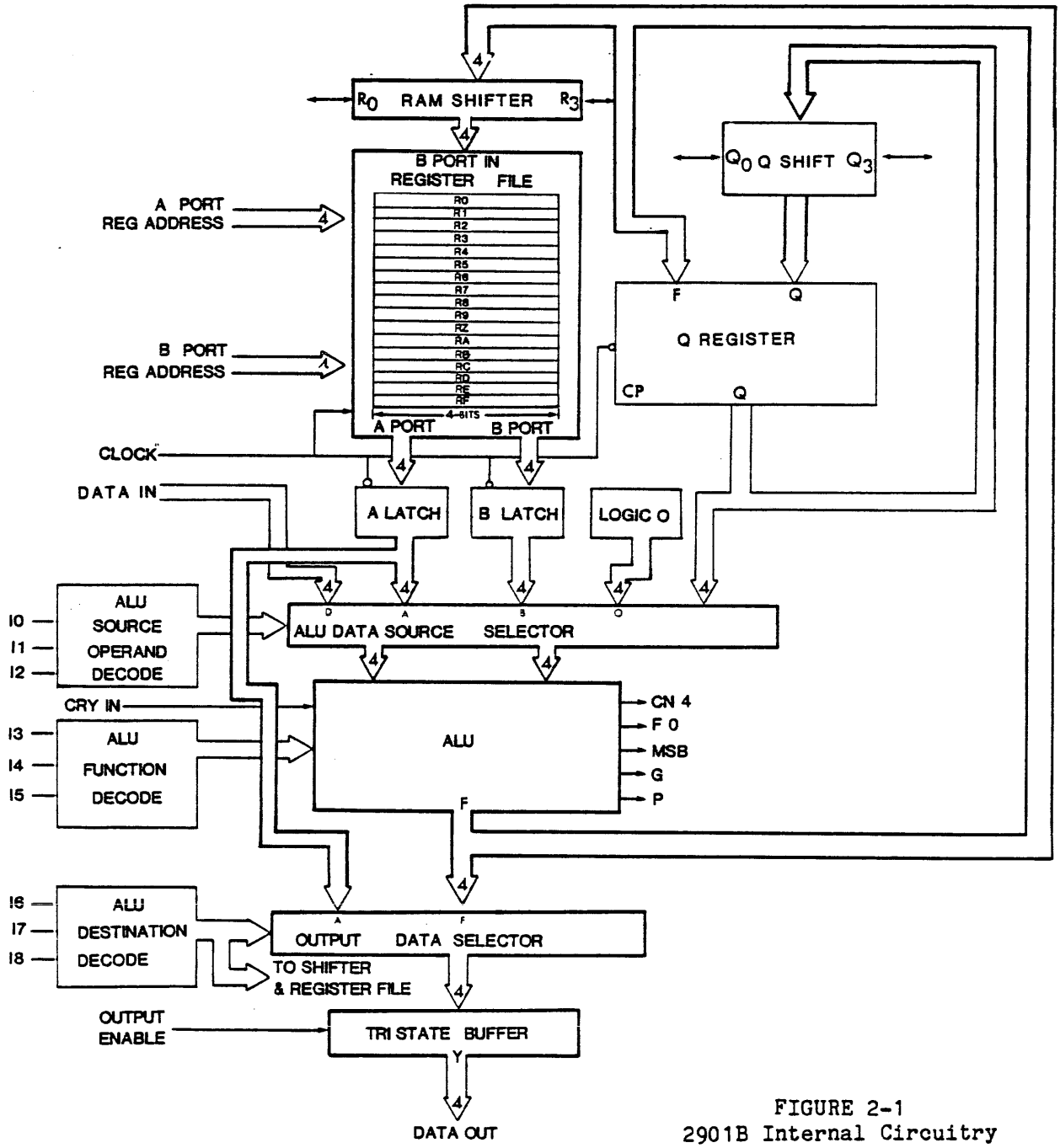


FIGURE 2-1
2901B Internal Circuitry

The Q64 microprocessor is made up of four cascaded 2901B bit slice microprocessor devices. Each 2901B is a 4-bit wide processing element containing a 16 register file, a high speed ALU, and all the necessary decoders and multiplexers for ALU function decoding, data routing, and shifting operations.

Figure 2-1 is a block diagram of the 2901B internal circuitry. The 16 location register file has 2 output ports, permitting 2 file registers to be accessed simultaneously. The A and B port register addresses specify the location of the registers in the register file. Figure 2-2 is a more detailed block diagram of the 2901B, showing the latches and multiplexers discussed below.

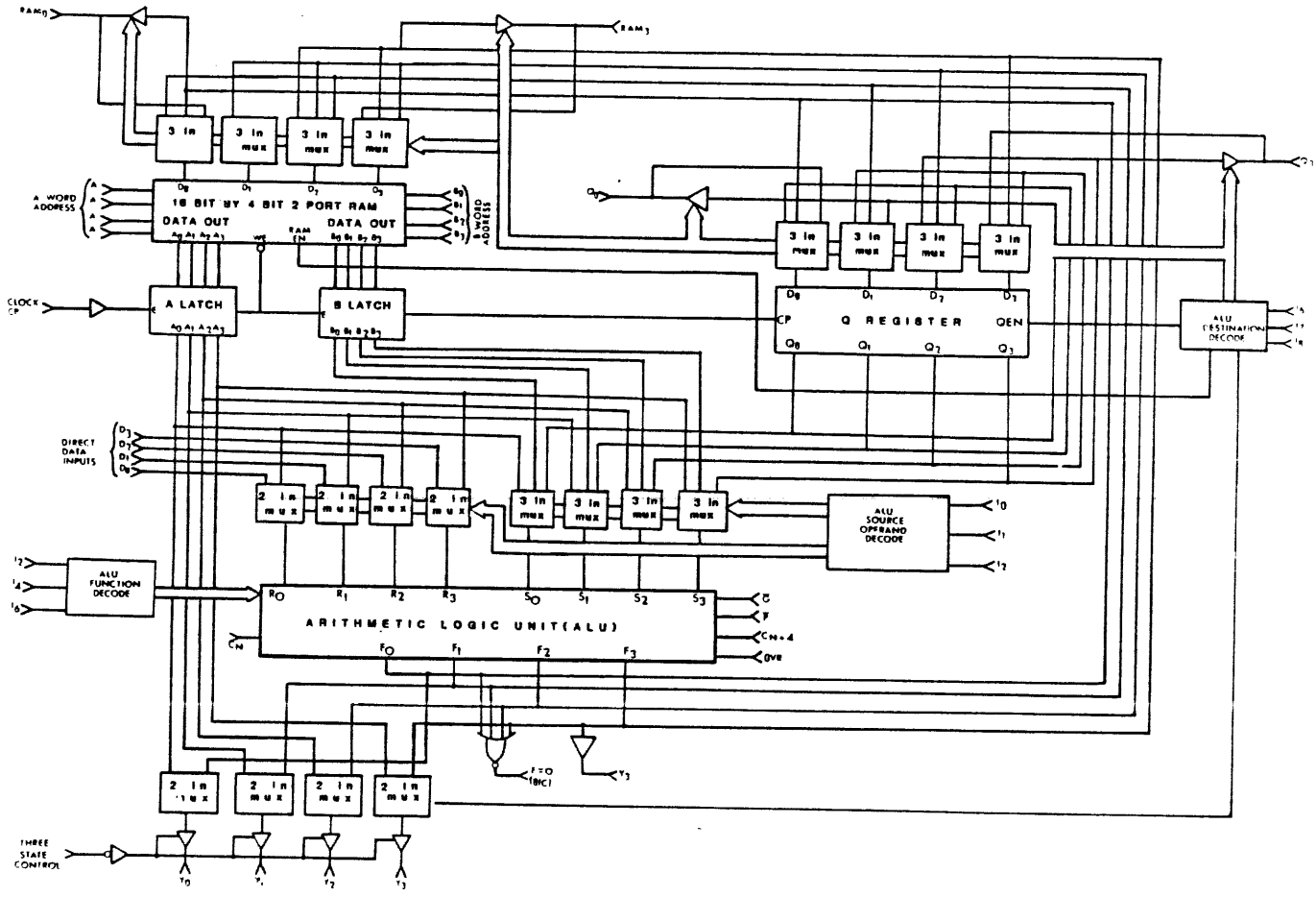


FIGURE 2-2
2901B Block Diagram

The clock input controls the 2901B RAM, the Q register and the A and B data latches. Data is clocked into the Q register on the low-to-high transition of the clock. When the clock input is high, the A and B latches are open and will pass whatever data is present at the 2901B RAM outputs. When the clock input is low, the latches are closed and will retain the last data entered. If the 2901B RAM is enabled, new data will be written into the file defined by the B address field when the clock is low. If the register file addresses and the Instruction Bits are stable, the register files may be accessed and the ALU will function without the clock dropping low. Figure 2-3 shows the 2901B internal timing.

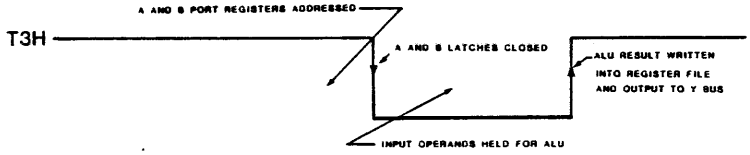


FIGURE 2-3

The 2901Bs are controlled by nine Instruction Bits(I0-I8) which are encoded in three 3-bit fields. I0-I2 selects the data source for the R & S input; I3-I5 selects the ALU function; and I6-I8 selects the destination for the resultant of the operation. Table 2-1 lists the coding of the Instruction Bits. ALU instruction coding is discussed in detail in Section 3.1 of this document.

ALU Function Control

MICRO CODE				ALU
I_5	I_4	I_3	Octal Code	Function
L	L	L	0	R Plus S
L	L	H	1	S Minus R
L	H	L	2	R Minus S
L	H	H	3	R OR S
H	L	L	4	R AND S
H	L	H	5	$\overline{R}$ AND S
H	H	L	6	R EX-OR S
H	H	H	7	R EX NOR S

ALU Source Operand Control.

MICRO CODE				ALU SOURCE OPERANDS	
I_2	I_1	I_0	Octal Code	R	S
L	L	L	0	A	Q
L	L	H	1	A	B
L	H	L	2	O	Q
L	H	H	3	O	B
H	L	L	4	O	A
H	L	H	5	D	A
H	H	L	6	D	Q
H	H	H	7	D	O

ALU Destination Control.

MICRO CODE				RAM FUNCTION		Q-REG. FUNCTION		Y OUTPUT	RAM SHIFTER		Q SHIFTER	
I_8	I_7	I_6	Octal Code	Shift	Load	Shift	Load		RAM_0	RAM_3	Q_0	Q_3
L	L	L	0	X	NONE	NONE	F→Q	F	X	X	X	X
L	L	H	1	X	NONE	X	NONE	F	X	X	X	X
L	H	L	2	NONE	F→B	X	NONE	A	X	X	X	X
L	H	H	3	NONE	F→B	X	NONE	F	X	X	X	X
H	L	L	4	DOWN	F/2→B	DOWN	Q/2→Q	F	F_0	IN_3	Q_0	IN_3
H	L	H	5	DOWN	F/2→B	X	NONE	F	F_0	IN_3	Q_0	X
H	H	L	6	UP	2F→B	UP	2Q→Q	F	IN_0	F_3	IN_0	Q_3
H	H	H	7	UP	2F→B	X	NONE	F	IN_0	F_3	X	Q_3

TABLE 2-1
2901B Instruction
Bit Coding

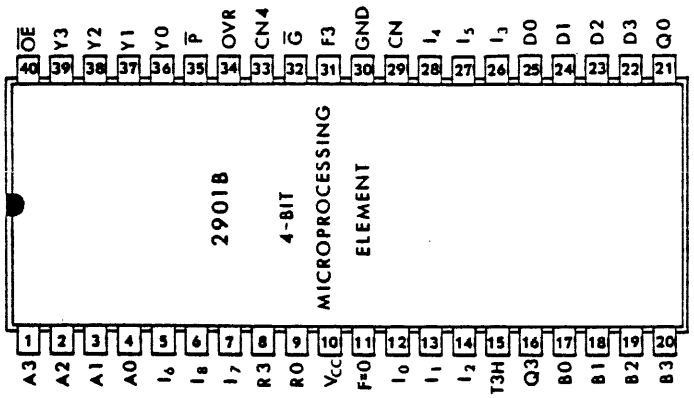


TABLE 2-2

A0-A3 A PORT REGISTER ADDRESS INPUTS used to specify one of the 16 register file locations, causing its contents to be routed to the A port output of the register file.

I_6-I_8 INSTRUCTION CONTROL INPUTS used to specify the destination for the resultant of an ALU operation, specifies the source of the data outputted from the 2901B (pins Y0-Y3) and defines any shift operation.

R3 MSB RAM SHIFTER The most significant bit of the shifter situated on the data inputs of the register file. Serves as an input on shift right operations and an output on shift left operations. When the I_6-I_8 inputs do not specify a shift operation, R3 enters tri-state mode.

Vcc POSITIVE SUPPLY TERMINAL +5V power supply connection.

F = 0 ZERO OUTPUT an ALU status signal indicating the resultant of an ALU operation equals zero.

I_0-I_2 INSTRUCTION CONTROL INPUTS used to specify the sources of the input operands to the ALU.

T3H CLOCK INPUT used to control the A latch and B latch, and specifies the time when data is written into the register file.

Q3 MSB Q SHIFTER most significant bit of the Q shifter.

B0-B3 B PORT REGISTER ADDRESS INPUTS used to select one of the 16 register file locations, causing its contents to be presented to the B port output of the register file and acting as a destination pointer when data is written into the register file.

Q0 LSB Q SHIFTER least significant bit of the Q shifter.

D0-D3 DIRECT DATA INPUTS a 4-bit slice of data entering the 2901B which may be selected as one of the input operands of the ALU by the I_0-I_2 Instruction Control Inputs.

I_3-I_5 INSTRUCTION CONTROL INPUTS used to specify what function the ALU will perform (see Table A).

CN CARRY INPUT the carry in to the ALU.

GND GROUND TERMINAL 0V power supply connection.

F3 MOST SIGNIFICANT BIT an ALU status signal displaying the state of the most significant bit of the resultant of an ALU operation.

$\overline{G}$ CARRY GENERATE an ALU status signal routed to the carry lookahead generator, indicates (even before the ALU operation begins) that the most significant bits of the ALU input operands are high and a carry is inevitable.

CN4 CARRY OUT the carry out from the ALU (Not Used).

OVR OVERFLOW an ALU status signal, not used in the Q30.

$\overline{P}$ CARRY PROPAGATE an ALU status signal routed to the carry lookahead generator indicates (by examination of the ALU input operands) there will be a carry propagated out of this 2901B if a carry enters from a less significant ALU slice.

Y0-Y3 DATA OUTPUTS four bits of data are outputted when these tri-state lines are enabled. Data may be outputted from the ALU or the contents of the A latch may be outputted. The source of the output data is specified by I_6-I_8.

$\overline{OE}$ OUTPUT ENABLE when logically high the Y0-Y3 outputs are disabled and enter tri-state mode.

R0 The least significant bit of the shifter situated on the data inputs of the register file. Serves as an input on shift left operations and as an output on shift right operations. When the I_6-I_8 inputs do not specify a shift operation, R0 enters tri-state mode.

The ALU in the 2901B can receive its input operands from: the contents of any combination of register pairs; external data; or forced 0 multiplexed onto the R or S inputs of the ALU. Eight ALU functions are employed: three arithmetic and five logical. The data output by the 2901B may be the resultant data emerging from the Y outputs of the ALU, or the contents of the A latch.

The ALU data output can be routed to several destinations(see Table 2-1). It can be the device output, it can be stored in the RAM or Q register, or both output and stored. The data output may be input to the 2901B RAM non-shifted, shifted up one position(multiplied by 2) or shifted down one position(divided by 2). (Note: up is toward the MSB and down is toward the LSB). The 2901B RAM inputs are driven by a 3-input multiplexer. The shifter has two ports: RAM0 and RAM3. In the shift up mode, the RAM3 buffer is enabled and the RAM0 multiplexer input is enabled. In the shift down mode, the RAM0 buffer and RAM3 multiplexer are enabled. The Q register is also driven from a 3-input multiplexer and functions parallel to the RAM Shifter.

The ALU in the 2901B generates 4 status signals that are used in the Q64: ALU resultant equals zero(F=0), most significant or sign bit(F3), carry generate(G) and carry propogate(P). The carry generate and carry propogate are used with a Lookahead Carry Generator. A fifth status signal from each 2901B(carry out) is generated from the propogate and generate signals to the Lookahead Carry Generator(see below).

Table 2-2 is a list of pin definitions for the 2901B microprocessing element.

2.2 Q64 MICROPROCESSOR

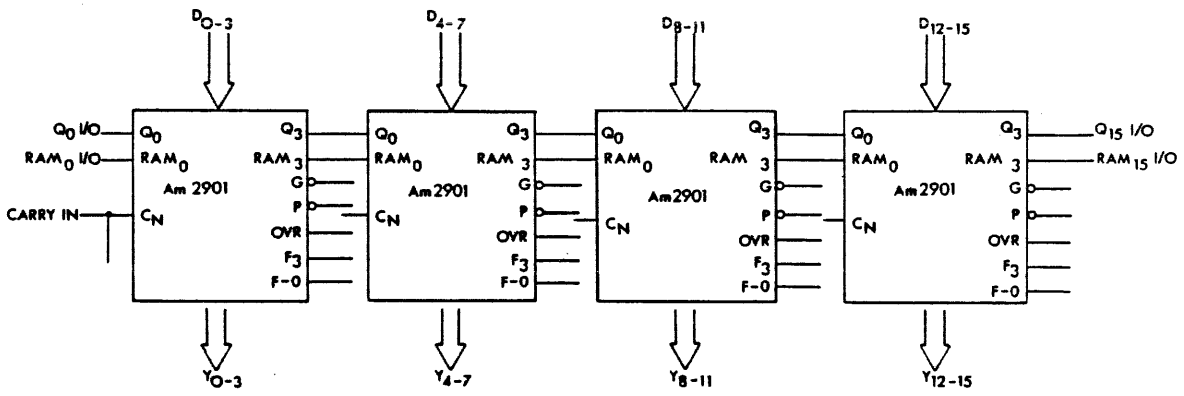


FIGURE 2-4
16-Bit ALU

Thus far the discussion has centered around the function of a single 2901B device, in isolation from other devices. As noted previously, the Q64 microprocessor is made up of four cascaded 2901Bs. Figure 2-4 shows the basic interconnections for a 16-Bit CPU. Figure 2-5 is a block diagram of the Q64 CPU, showing the relationship of the 2901Bs to the Pipeline Register, the Lookahead Carry Generator and the Shift MUX.

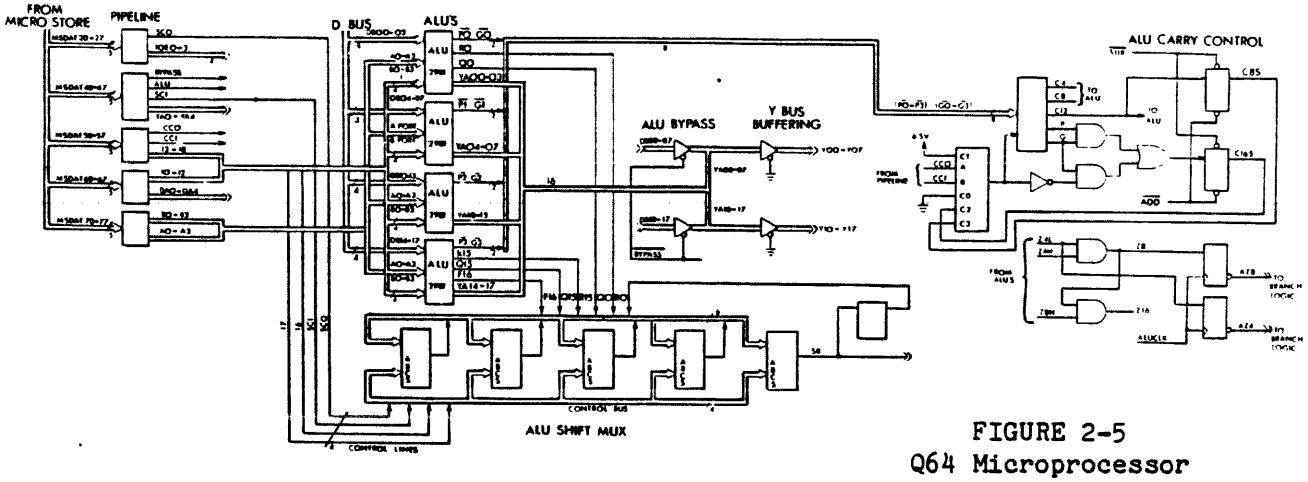


FIGURE 2-5
Q64 Microprocessor

The Q64 uses a Lookahead Carry Generator for carry between devices as well as carry-out from the array because of the greater speed. The anticipated carry is faster than ripple carry through cascaded devices. Carry controls(CC0 & CC1) in microcode determine the carry input(C0) to the least significant 2901B. Specific coding of the carry controls is given in Section 3.2.4.

The Q register and the RAM shift data transfers occur between devices over bi-directional lines. At the ends of the array, three-state multiplexers are used

to select the new inputs to these registers during shifting. Shift controls(SC0 & SC1) are generated by the microcode. Specific coding is listed in Section 3.2.5.

2.3 COMPUTER CONTROL UNIT

The Computer Control Unit is the portion of the Q64 hardware that performs the microcode fetches. The speed of a computer is is largely dependent upon the access time to the microprogram memory, called the Microstore in the Q64. Figure 2-6 shows the relationship of the Next Address Control Logic, the Microstore, the Pipeline Register and the 2901Bs in the Q64.

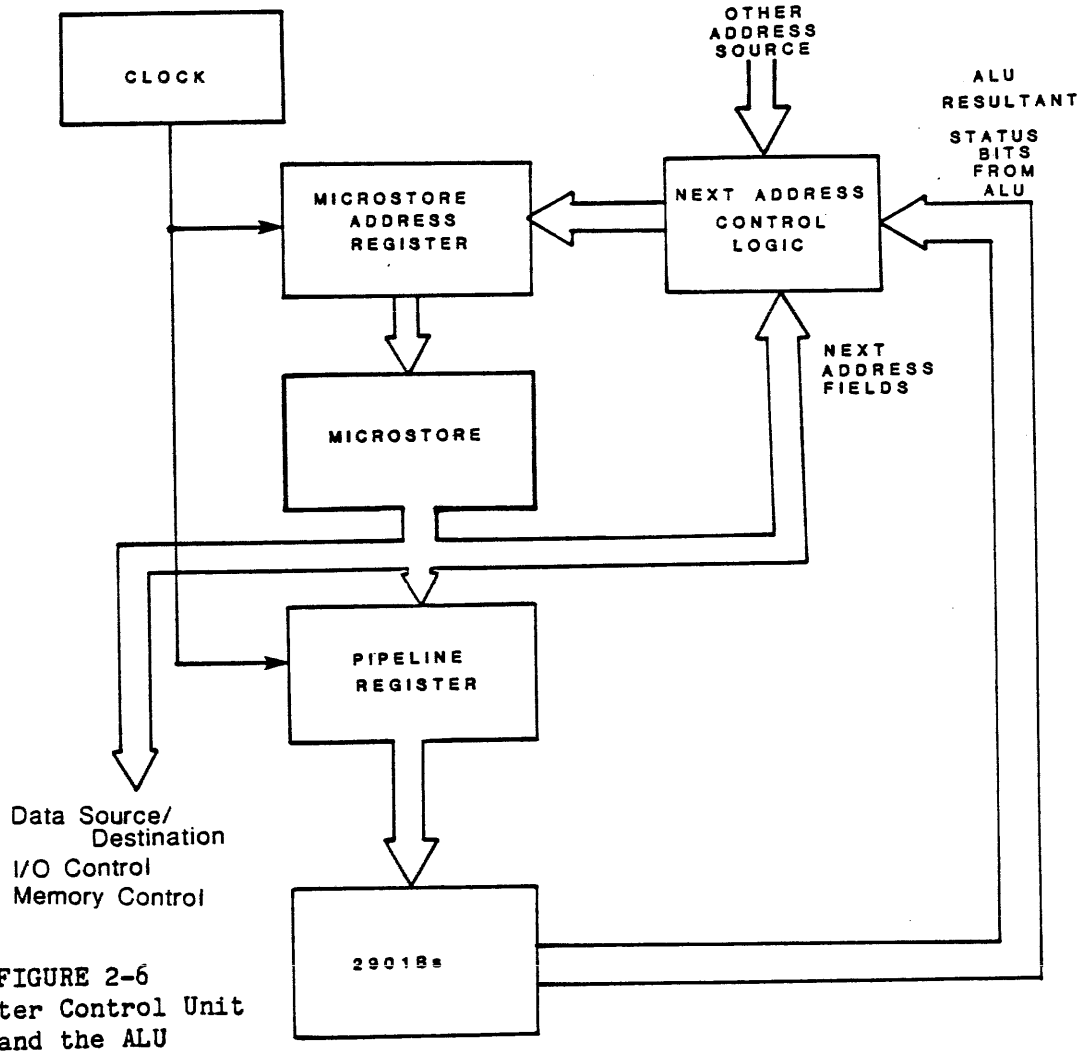


FIGURE 2-6
Computer Control Unit and the ALU

A macroinstruction is performed by executing several microinstructions in sequence. Each microinstruction contains not only bits to control the data hardware, but also bits to define the location in the Microstore of the next microinstruction to be executed. The Next Address Fields of the microword determine how the address in the Next Address Register will be modified to generate the address of the next microword.

The Microstore contains firmware routines of machine instructions. Microprogram flow is achieved through the Next Address Control Logic which determines if the address of the next instruction will be the next sequential instruction in the Microstore(incrementing the address of the instruction currently being executed), taken from the micro stack, or if some form of branch(conditional or unconditional) is required.

For example, the Microstore contains tables of branch instructions, each pointing to the starting address of firmware routines. By adding the offset determined by a specific machine instruction to the table starting address, the Next Address Control Logic is able determine the address of the appropriate branch instruction in the branch table.

Conditional branches require the testing of status bits from the 2901Bs or some other source by the Next Address Control Logic to determine the address of the next microinstruction. A series of decoders and selector/multiplexors are used to select the appropriate address to be loaded into the Next Address Register.

Figure 2-7 is a detailed block diagram of the Q64 Next Address Logic. Specific branches and conditions are discussed in Section 3.5.

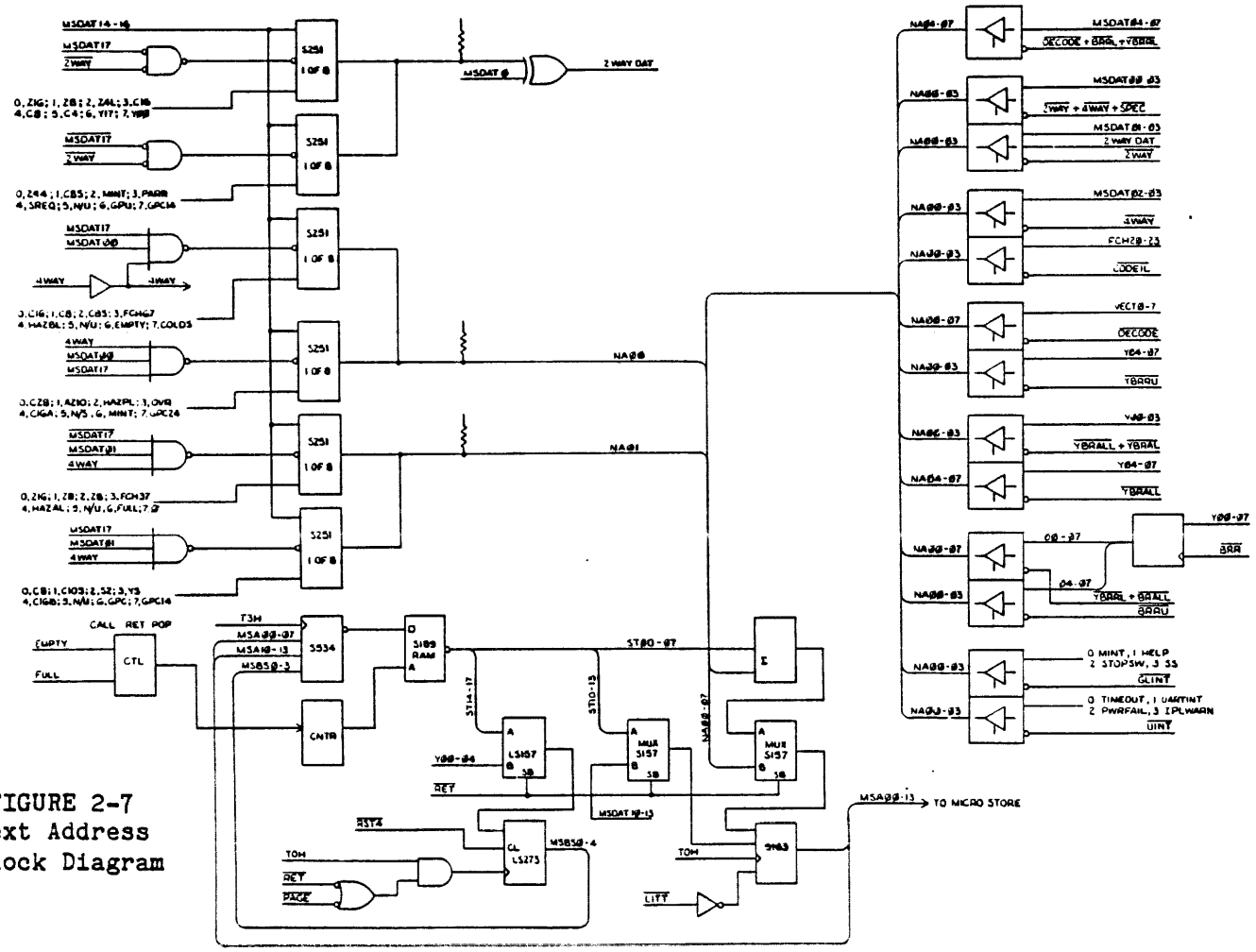


FIGURE 2-7
Next Address
Block Diagram

2.3.1 External Control

The Q64 Microstore ROMs may be disabled and control of the CPU assumed by external devices for diagnostic or development purposes. Hardware on the Q64 includes 2K of static RAM which makes up the Writeable Control Store(WCS). WCS is used in conjunction with the Q64 Test Panel which is connected to the System 64 through one of two serial diagnostic ports on the Auxilliary(AUX) Board. The two ports on the AUX Board allow the Q64 Test Panel to be connected to the system on site via a standard RS232 connector, or through a modem from a remote site. The Q64 Test Panel may be set up to monitor certain CPU functions or run diagnostics using WCS. The ROM Simulator is a special board which is installed in a host system and connected to a port on the ALU Board for development purposes.

2.3.2 WCS

In the diagnostic mode, a microprogram is written to WCS and then executed. The Microstore ROM is the only part of the Q64 CPU that is not utilized in the execution of the WCS microprogram. The 2901Bs, Next Address Logic and other related logic are used by the microprogram in WCS in the same manner as they are used by the resident microprogram in Microstore.

2.4 PIPELINING

The address of a microinstruction is clocked to the outputs of the Microstore Address Register on the rising edge of a clock signal. The address goes to the Microstore ROMs, and an access time later, the microinstruction is at the ROM outputs.

The Pipeline Register is put at the output of the Microstore to essentially split the system in two and speed processing by overlapping the execution of one instruction with the fetch of the next instruction. The Pipeline Register is clocked on the rising edge of the same clock signal as the Microstore Address Register(T0H), while the Next Address Fields of the Microword are not pipelined (see Figure 2-6).

The Next Address Fields of the Microword are directly input to the Next Address Logic, so that the next instruction is sitting at the inputs of the Pipeline Register while the instruction currently in the Pipeline Register is being executed. The presence of the Pipeline Register allows the microinstruction fetch to occur in parallel with the data operation, rather than serially, allowing the clock frequency to be approximately doubled.

2.5 SYSTEM TIMING

The Q64 clocks are generated from a free running 20 megahertz oscillator. The oscillator output is used as a "1/4 Clock" which has a period of 50 nanoseconds (NS) and a pulse of 25NS.. The 25NS 1/4 Clock is used to generate a 1/2 Clock which has a 100NS period and a 50NS pulse.

The 1/2 Clock is then used to generate four clocks with 400NS periods and 100NS pulses. The four 100NS free running clocks - T0, T1, T2 & T3 - define the 400NS machine cycle of the Q64 CPU. Figure 2-8 is a representation of the free running system clocks.

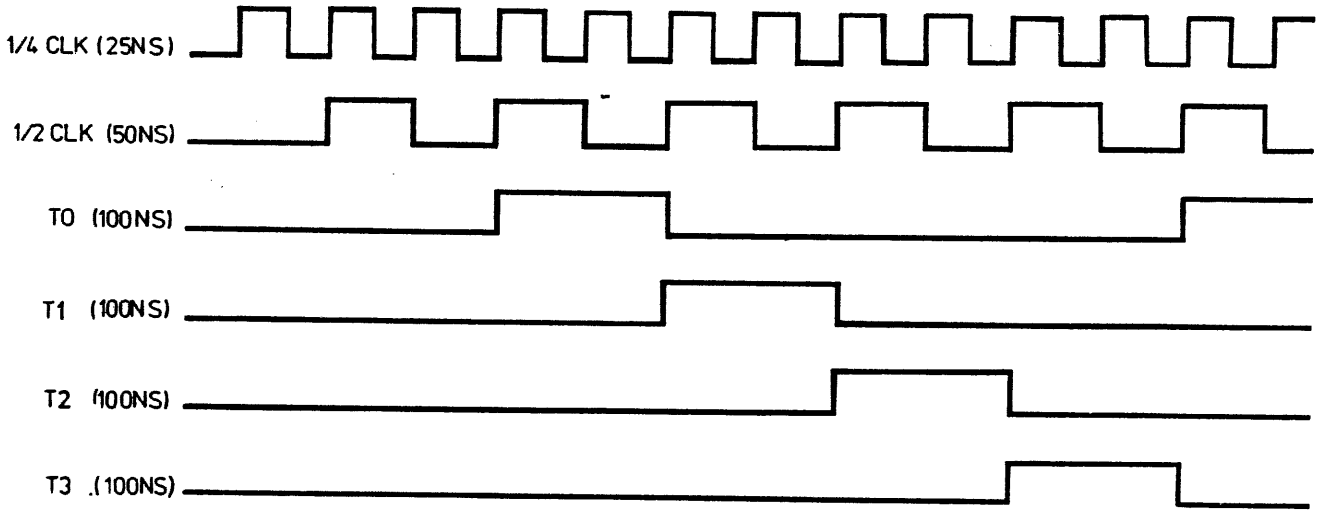


FIGURE 2-8
Discrete Clocks

The free running 100NS clocks are input to a MUX to generate "hold" clocks - T0H, T1H, T2H & T3H - which can be suspended to allow for mapping of the physical address register preceding a memory access using that address register. When a clock hold is asserted, the MUX selects inputs tied to ground as the outputs. Since certain CPU functions are enabled by the "hold" clocks, CPU activity is suspended when a clock hold is asserted.

When a clock hold is required, it is asserted on the rising edge of the 1/4 clock after T2 following the request(T2.5). The clocks will be held for one full machine cycle(400NS) and will resume at the point they left off. In other words, a clock hold will last from T2.5 until the following T2.5. T3H, T0H, T1H, and T2H will be suspended during this cycle, which will resume with the last 50NS of T2H.

Table 2-3 lists some of the functions directly enabled by system clocks. Note the functions that will be suspended when a clock hold is asserted(all functions listed in the lower half of the table).

T0	T1	T2	T3
Programmable Interval Timer clocked UART Interrupt clocked IO Data clocked Refresh Counter incremented Fetcher Sequence reset RAS and CAS reset Micro Halt asserted (clocks held) Micro Continue asserted (clocks restarted) D Source Decode T0.5 Map Hold set	LAR clocked T1.5 LAR loaded by Fetcher or ALU Bypass T1.5-2 LAR incremented or decremented	IPL Resets clocked (2nd T2 after IPL) ALU output valid T2.5 Clock Hold asserted Clock Hold reset	Service Request reset Map Hold reset LAR clocked PAR clocked Map Flags reset T3.5 RAR Mapped WCS Address valid
T0H	**T1H**	**T2H**	**T3H**
Pipeline clocked Microstore Address valid Fetcher started Fetch Sequence valid Memory Controls clocked Base Pointer clocked	WCS Write enabled Write to Stack Indirect Counter reset Fetch Vector Valid LARPO Readback	Fetch Sequence valid Fetcher Start reset	ALU Clocked Save Register clocked Carry Controls clocked Next Address Mode & Select decode Address read from Stack WCS Address incremented Y Destination decode GPC clocked Indirect Counter clocked Fetch Vector valid T3.5H LAR Load from ALU ("Load Late") Map Pending set on "Load Late"

TABLE 2-3
System Timing

Intervals between the 100NS clocks are split in a variety of ways. The interval between T0 and T2 is divided into 50NS segments for Map Control and Timing through the use of a shift register. A T3.5H clock signal is generated by inputting T3H and 1/2CLK to a NAND Gate. The clock hold is asserted and reset at T2.5 by latching the signal through two flip-flops, first at T2, then on the 1/4CLK.

Figure 2-9 shows the overlapping clocking of the ALU and the Pipeline discussed previously. Note that the Pipeline is clocked on the rising edge of T0H, while the ALU is clocked on the trailing edge of T3H(the rising edge of the inverted clock signal), giving nearly one full machine cycle between the clocking of the Pipeline and the clocking of the ALU.

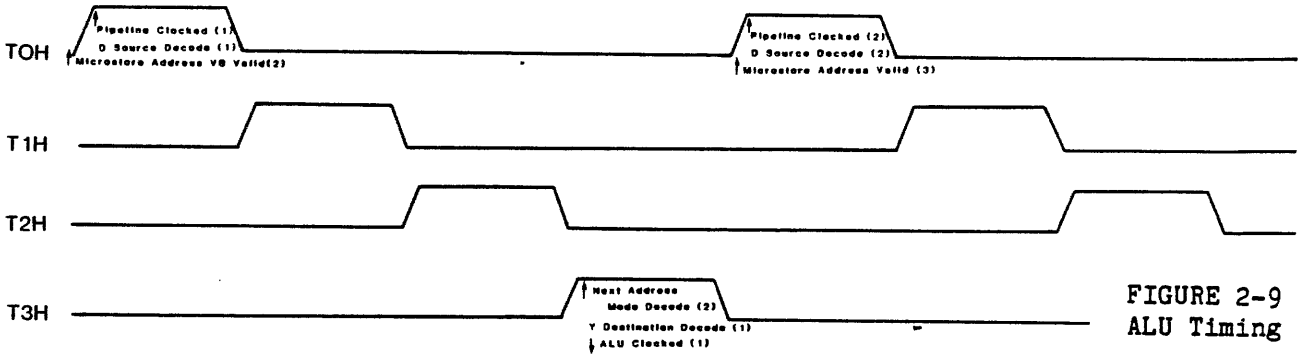


FIGURE 2-9
ALU Timing

Specific timing is discussed in more detail in the sections of the manual dealing with the Fetcher(Section 2.6.5), Map Control and Timing(Section 2.8), and the Memory Data Interface(Section 2.9).

2.6 THE FETCHER

The Fetcher is a microstate machine that speeds up the fetching and decoding of REAL instructions. The Fetcher resolves all indirect addresses on the Operand A and executes branch instructions without participation by the microcode.

The Fetcher is started by the microcode specifying one of three special memory controls(*FETCH, *WAFETCH or *WBFETCH). It can also be started in the "decode only" mode by a bit in the Local Control Register. The Fetcher interfaces to the microcode by supplying a "Decode Vector" to be interpreted by the microcode.

The following is a list of Fetcher capabilities:

1. Read data from memory into the Fetch Register using address registers B or P.

2. Decode REAL instructions and produce an 8 bit vector for the next address logic of the microprocessor.

3. Increment, reset and test a 4-bit Indirect Address Counter.

4. Examine the contents of the Qantel Flag Register and perform branch decisions.

5. Step through 32 microstates.

6. Load address registers A, B and P with selected operands from the Fetch Register. It may also load the constant $000F into address register A.

7. Increment address register P by factors of 3, 6 or 8.

8. Save the current value of the Qantel Program Counter (LARP) in the "old" program counter register(LARPO).

Note that the Fetcher cannot perform memory write operations.

2.6.1 Memory Fetch

With the exception of the fetching of an indirect address, the Fetcher performs all memory accesses using LARP. On a fetch, the content of LARP(the Program Counter) is moved to LARPO(the "Old Program Counter).

If there is a base or bank hazard on LARP(the address in LARP is within 8 bytes of a 2K boundary), the Fetcher will terminate with a vector of "Hazard in Instruction". If there is a memory parity error, it will terminate with a vector of "Parity Error."

In the absence of the above two conditions, a 64-bit memory word is read into the Fetch Register. The byte at the address specified in LARP is read into the first byte of the Fetch Register, LARP + 1 to the second byte, etc. Table 2-4 lists the macrocode partitions of the Fetch Register.

BYTE	LABEL
1&2	Operand 1(OP1)
3	Opcode 1(CODE1)
4&5	Operand 2(OP2)
6	Opcode 2(CODE2)
7&8	Operand 3(OP3)

TABLE 2-4
Fetch Register
Macrocode Partitions

2.6.2 Fetcher Processing

After the memory data is in place, the Fetcher will decode the opcodes and resolve any indirect addressing on Operand 1.

2.6.2.1 Opcode Decode

Based on the value of CODE1 and CODE2, the fetcher interprets the instruction as one of four types: branch; single operand; double operand; or triple operand.

2.6.2.2 Branch Instructions

Branch Instructions are decoded as having one of the values in column 1 of Table 2-5 in the CODE1 Register.

CODE	MNEMONIC	INSTRUCTION
$A0	NOP	No Operation
$A1	BOV	Branch on Overflow
$A2	BMI	Branch on Minus
$A3	BNZ	Branch on Nonzero
$A4	BZ	Branch on Zero
$A5	BNM	Branch on Not Minus
$A6	BNO	Branch on Not Overflow
$A7	BRU	Branch Unconditionally
$AA	BP	Branch on Plus (Not minus and not zero)
$AF	BNP	Branch on Not Plus (Minus or zero)

TABLE 2-5
Fetcher
Branch Codes

The Fetcher will increment LARP by 3. It determines whether each branch is taken or not taken by examining the Flag Register which contains the Qantel condition flags. BRU is treated as a branch always taken, and NOP is treated as a branch never taken.

2.6.2.2.1 Branch Not Taken

If the branch is not taken and the Fetcher is not in the Single Step mode, the Fetcher starts another memory fetch using the incremented address in LARP and the Decode Vector is set in the "Running" state. If the Fetcher is in the Single Step mode, it will complete with a "Complete" vector.

2.6.2.2.2 Branch Taken

If the Branch is taken, Operand 1 is the "branch to" address. The Decode Vector is set to "Busy", unless in the Single Step mode as noted above.

The most significant bit of the OP1 is checked to see if an indirect address is specified(bit 2**15 is 1). If so, the Fetcher will resolve the indirect by decrementing the the address by one before loading it into LARP(indirects are right-hand or low-byte addressed). Two bytes at LARP and LARP+1 are read into OP1 of the Fetch Register. This address is then checked for further indirection and loaded into LARP.

The Fetcher will continue to resolve up to 16 levels of indirection, after which the chain is declared illegal and the Fetcher will terminate with a Decode Vector of "Excessive Indirect Chain."

A parity error detected during indirect resolution will cause the Fetcher to terminate with a "Parity Error" Decode Vector, while a base or bank hazard on LARP during indirect resolution will cause the Fetcher to terminate with a Decode Vector of "Hazard on Indirect P." In the latter case the microcode will resolve the indirect, load the value into LARP, and start the Fetcher on the next microinstruction.

2.6.2.3 Single Operand Instructions

Single operand instructions are decoded by having CODE1 not equal to $Fx and not being one of the branch instructions listed above.

The Fetcher increments LARP by 3 and maps the constant $000F into LARA. OP1 is mapped into LARB. If it is an indirect address, it is resolved. The Decode Vector is set to the appropriate state and the Fetcher completes.

Indirect addressing is resolved in the same manner as with branch instructions; however if a base or bank hazard on LARB occurs, the Fetcher will hand off control to the microcode with the Decode Vector "Hazard on Indirect B." The microcode will resolve the indirect chain, loading the value into LARB, then restart the Fetcher for decode only.

2.6.2.4 Double Operand Instructions

Double operand instructions are decoded as having CODE1 equal to $Fx, and CODE2 not equal to $87 or $8B.

The Fetcher increments LARP by 6. OP1 is mapped to LARB and indirect addresses are resolved as described for single operand instructions. OP2 is mapped to LARA. The Fetcher hands off control to the microcode with a Decode Vector determined by CODE2 and the indirect bit of OP2.

2.6.2.5 Triple Operand Instructions

Triple operand instructions are decoded by having CODE1 equal to $Fx and CODE2 equal to $87 or $8B.

The Fetcher increments LARP by 8. OP1 is mapped to LARB and indirect addresses are resolved as described for single operand instructions. OP2 is mapped to LARA. Control is handed off to the microcode with the Decode Vector determined by CODE2 and the indirect bits on OP2 and OP3.

2.6.3 Fetcher Microstates

Figure 2-10 is a block diagram of Fetcher operation. Each Fetcher microstate (sequence) is 200NS(half a microcycle).

The mnemonic listed to the left of each block is the microstate name. Each block describes all the events that occur in a single microstate. Blocks connected by a vertical line indicate flow from one microstate to the next.

The numbers to the right of each block are the hexidecimal representation of the microstate. During a given microstate, the Fetcher Sequence Latch will contain the value shown. The Fetcher Sequence Latch is shown on the Memory Data Board at 11C3.

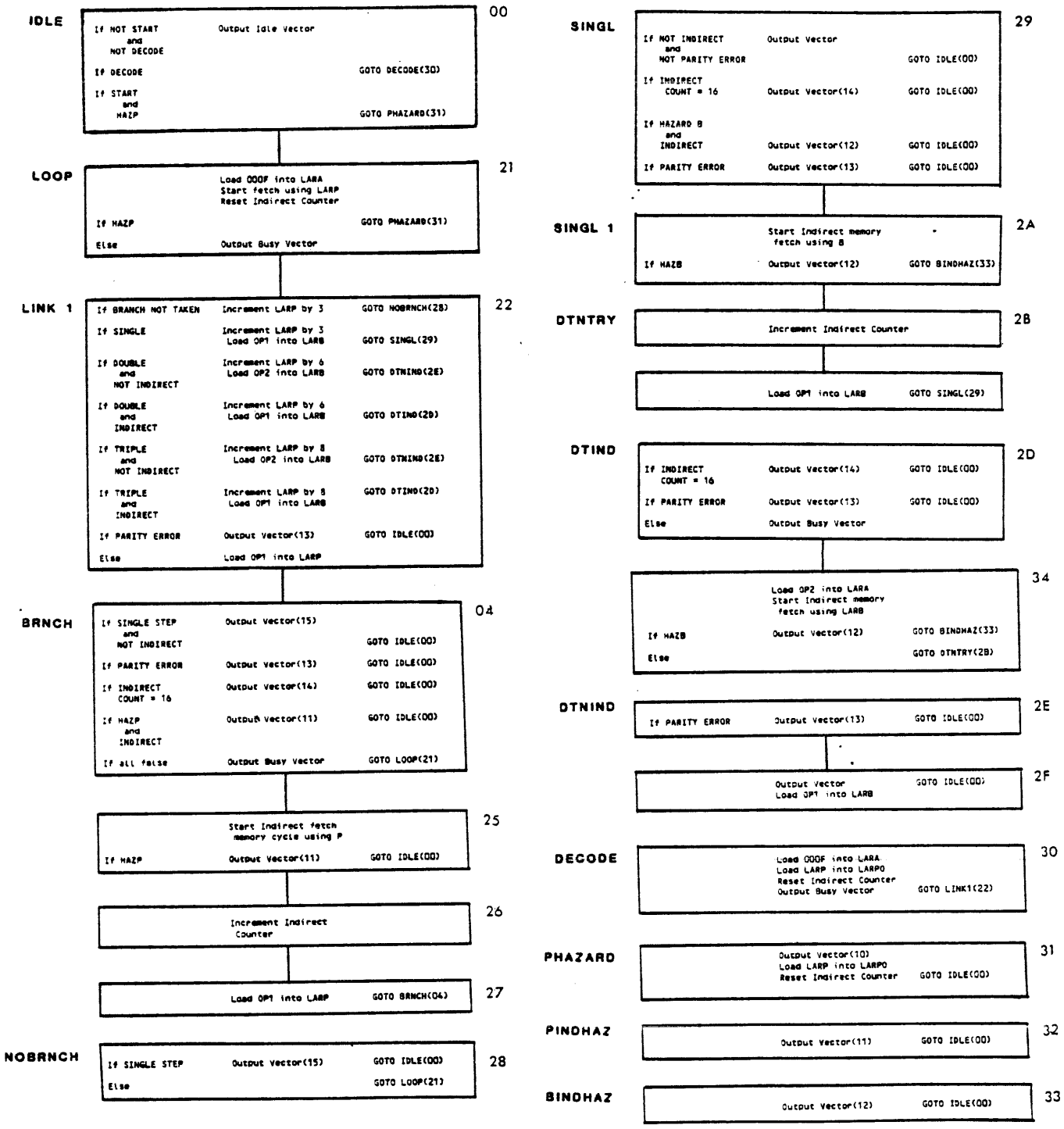


FIGURE 2-10
Fetcher Microstate
Block Diagram

2.6.4 Fetcher Decode Vectors

Table 2-6 lists the Decode Vectors(in hex) output by the Fetcher and the meaning of the vectors. The hexidecimal values listed under the meanings are object code for REAL instructions. These values will be contained in either the CODE1 or the CODE2 Register depending upon the type of the instruction(single, double or triple).

DECODE VECTOR	MEANING	DECODE VECTOR	MEANING	DECODE VECTOR	MEANING	DECODE VECTOR	MEANING
00-0F	Running & GLINT	40	98	76	83, Indirect	96	91, Indirect
10	Hazard on Instruction	41	99	77	83	97	91
11	Hazard on Indirect P	42	9A	78	84, Indirect	98	92, Indirect
12	Hazard on Indirect B	43	9B	79	84	99	92
13	Parity Error	44	9C	7A	85, Indirect	9A	93, Indirect
14	Excessive Indirect Chain	45	9D	7B	85	9B	93
15	Complete and Single Step	46	9E	7C	86, Indirect	9C	94, Indirect
16-1F	Not defined	47	9F	7D	86	9D	94
20	Single Operand, CODE1 = 00-0F	48	A0-A7	7E	Triple Operand 87, OP2 & OP3 Indirect	9E	95, Indirect
21	10-1F	49	A8	7F	Triple Operand 87, OP3 Indirect	9F	95
22	20-2F	4A	A9	80	Triple Operand 87, OP2 Indirect	A0	96, Indirect
23	30-3F	4B	AA	81	Triple Operand 87, No Indirects	A1	96
24	40-4F	4C	AB	82	Double Operand 88, OP2 Indirect	A2	97, Indirect
25	50-5F	4D	AC	83	Double Operand 88, No Indirect	A3	97
26	60-6F	4E	AD	84	89, Indirect	A4	98, Indirect
27	70-7F	4F	AE	85	89	A5	98
28	80	50	AF	86	8A, Indirect	A6	99, Indirect
29	81	51	B0-BF	87	8A	A7	99
2A	82	52	C0-CF	88	Triple Operand 8B, OP2 & OP3 Indirect	A8	9A, Indirect
2B	83	53	D0-DF	89	Triple Operand 8B, OP3 Indirect	A9	9A
2C	84	54	E0-EF	8A	Triple Operand 8B, OP2 Indirect	AA	9B, Indirect
2D	85	55-5F	Not defined	8B	Triple Operand 8B No Indirects	AB	9B
2E	86	60	Double Operand 00-0F, OP2 Indirect	8C	Double Operand 8C, OP2 Indirect	AC	9C, Indirect
2F	87	61	Double Operand 00-0F, OP2 not Indirect	8D	Double Operand 8C	AD	9C
30	88	62	10-1F, Indirect	8E	8D, Indirect	AE	9D, Indirect
31	89	63	10-1F	8F	8D	AF	9D
32	8A	64	20-2F, Indirect	90	8E, Indirect	B0	9E, Indirect
33	8B	65	20-2F	91	8E	B1	9E
34	8C	66	30-3F, Indirect	92	8F, Indirect	B2	9F, Indirect
35	8D	67	30-3F	93	8F	B3	9F
36	8E	68	40-4F, Indirect	94	90, Indirect	B4	A0-AF, Indirect
37	8F	69	40-4F	95	90	B5	A0-AF
38	90	6A	50-5F, Indirect			B6	B0-BF, Indirect
39	91	6B	50-5F			B7	B0-BF
3A	92	6C	60-6F, Indirect			B8	C0-CF, Indirect
3B	93	6D	60-6F			B9	C0-CF
3C	94	6E	70-7F, Indirect			BA	D0-DF, Indirect
3D	95	6F	70-7F			BB	D0-DF
3E	96	70	80, Indirect			BC	E0-EF, Indirect
3F	97	71	80			BD	E0-EF
		72	81, Indirect			BE	F0-FF, Indirect
		73	81			BF	F0-FF
		74	82,Indirect			C0-CF	Not defined
		75	82				

Note: All numerals are hexidecimal notation.

TABLE 2-6
Fetcher
Decode Vectors

The Decode Vector Latch is shown on the Memory Data Board schematics at 12C2. The CODE1 Register is also shown on the Memory Data Board schematics at 4B3. The CODE2 Register is at 5B3 on the Memory Data Board schematics.

2.6.5 Fetcher Timing

The Fetcher is clocked every 200NS by the clocks T0H and T2H, which creates restrictions on any one microstate. To accomodate the restrictions, all loops have an even number of states. The signal to start the Fetcher is gated with T2H and the decode signal is gated with T3H, and therefore the Fetcher will always start on a T0H step.

2.6.5.1 Fetcher Timing Limitations

Microstates which occur during T0H have no limitations and the Fetcher has the following capabilities:

1. Start main memory cycles;
2. Load logical address registers;
3. Set Decode Vector;
4. Increment address registers; and
5. Increment or reset Indirect Address Counter.

The following limitations are placed on the T2H microstates:

1. May not start main memory cycles;
2. May not load address registers; and
3. May not increment address registers.

The Fetcher has the following capabilities during T2H:

1. Set Decode Vector; and
2. Increment or reset Indirect Address Counter.

2.7 MEMORY ADDRESS INTERFACE

The Memory Address Interface provides automatic mapping of logical addresses to physical addresses. Required arithemetic calculation is performed by a 24-bit adder which is separate from the Arithemetic Logic Unit(ALU). The Base Register File contains 256 24-bit base registers(16 sets of 16) designed for fast context switching (only one set of 16 base registers is presently used). The 24-bit physical address provides an addressing capability of up to 16 megabytes of main memory.

2.7.1 Address Conversion

Figure 2-11 represents the address conversion process.

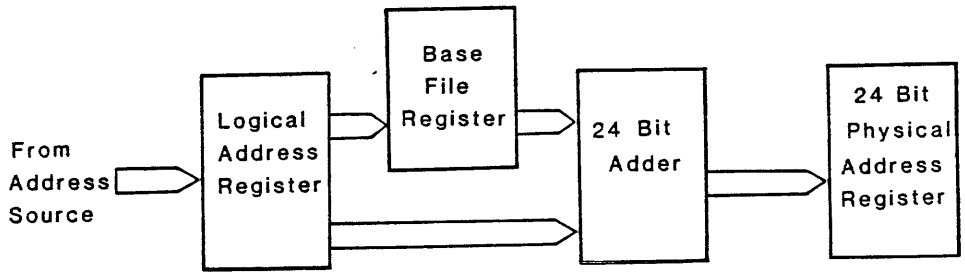


FIGURE 2-11
Address Conversion
Process

2.7.2 Address Registers

In the Q64 memory interface there are three logical address registers(LARA, LARB, and LARP) and four physical address registers(PARA, PARB, PARP, and RAR). Three of the four physical address registers are paired with logical address registers (A,B & P). The physical address register in each pair contains the physical equivalent of the partner logical address register. The remaining physical address register(RAR) is loaded directly by the microprocessor.

2.7.3 Logical Address

Figure 2-12 gives the bit format for the logical address.

IBBB BDDD DDDD DDDD

I - Indirect Address Flag
B - Base register select bits
D - Displacement

FIGURE 2-12
Logical Address

The most significant bit of the logical address is the Indirect Address Flag, which indicates that an address is indirect when it is set(high). The indirect address flag is resolved by the Fetcher of microcode and is not used by the hardware.

The next four bits refer to one of 16 Base Register Files which contains a 24-bit value used to calculate the physical address. The lower 11 bits of the logical address are the offset from the base value. The 11 bits give a range of 2K addresses for a single base value. The automatic mapping of physical addresses through the 16 Base Registers provides for addressing a memory block of 32K before the Base Register Files must be reloaded.

A logical address may be incremented or decremented, and it is possible that the change would carry into the base number. When this occurs, the base number is changed, and the new base number is used to obtain the new base value. Crossing the 2K base boundary is called a "base fault".

2.7.3.1 Logical Address Registers

There are three logical address registers in the Q64 memory interface(A,B,and P). To allow for the increment and decrement of the address, the least significant 3 bits of the logical address are implemented using a combination of a register and a ROM, while the remaining bits are loaded into counters. The counters allow for cascading of the increment to higher bits in the address.

2.7.3.2 Loading Logical Address Registers

The logical address registers may be loaded by the Fetcher or the microprocessor. There are two methods of loading by the microprocessor, one via the 2901 outputs, and the other through a 2901 Bypass operation. Loading a logical address register sets up the mechanism for mapping the corresponding physical address register.

When the logical address register is loaded by the Fetcher or a 2901 Bypass, the data is available in sufficient time to allow the mapping of the physical address register within the same microcycle. However, when the data must come from the 2901 outputs, the data is not available in time for mapping in the same microcycle, and mapping is deferred until the next microcycle. The timing delay is accomplished in part through tying the load enable generated by the Fetcher and the 2901 Bypass to a T1.5 clock signal, and tying the load enable for an operation involving the 2901 outputs to a T3.5 clock signal.

The data input to the logical address registers is clocked to the outputs on rising edge of the T1 or T3 clock signals. Data loaded at T1.5 will be valid at T3, while data loaded at T3.5 will not be valid until T1 in the following microcycle. Since the physical address registers are mapped on the trailing edge of T3, map controls set a pending flag and a map hold flag for the affected physical address register when the logical address register is loaded from the 2901 outputs(see Map Control and Timing Section 2.8).

2.7.3.3 Increment/Decrement of Logical Addresses

A lookup ROM is used to control the increment or decrement of a logical address by the Fetcher or the microprocessor. The Fetcher is limited to increment or decrement of LARP, while the microprocessor is capable of incrementing or decrementing all of the address registers.

The least significant 3 bits of the logical address presently in the register, a 3 bit increment select field(from the Fetcher or the Pipeline) and a direction flag are the address inputs to the lookup ROM. Coding of the increment select field is shown in Table 2-7.

CODE	MEANING
000	ALU increment/decrement by 1.
001	ALU increment/decrement by 2.
010	ALU increment/decrement by 8.
011	ALU increment/decrement by 5
100	Fetcher increment/decrement by 3.
101	Fetcher increment/decrement by 6.
110	Fetcher increment/decrement by 8.
111	Not used.

TABLE 2-7
Increment Select
Field Coding

Increment or decrement of the logical address is tied to a T1.5-2 clock and the incremented address is valid at the end T1. Increment or decrement of the physical address occurs concurrently with the logical address, but it is not valid until the end of T3H. If a base fault is encountered in the increment or decrement or the logical address, there is sufficient time for the fault to be resolved and the address mapped into the physical address register, overwriting the invalid incremented or decremented physical address.

2.7.4 Base Register File

The Base Register File is a 256 X 24-bit RAM file that contains base boundary values for resolving Qantel Logical Addresses. Since a logical address can only reference 16 base boundary values, only one set of Base Registers within the file is used. The remainder of the Base Registers may be used in future applications(e.g. P Codes).

A Base Register is accessed for three reasons: (1) to resolve(map) a logical address into a physical address; (2) to load or change the contents of a Base Register; or (3) to read the contents of a Base Register.

Only 4 of the 8-bit Base Register File Address used in mapping physical addresses come from the logical address. The 4 bits from the logical address are the low nibble of the Base Register Address which selects which register within a set is used. The remaining 4 bits which select the set of registers used is generated by the microprocessor. (NOTE: Since only one set of Base Registers is presently used, the 4 bits generated by the microprocessor are 0s).

2.7.4.1 Base Pointer

Prior to writing to or reading from a Base Register, the Base Register Address must be set up by the microprocessor. The address is loaded into the Base Pointer, which then gates the address to the Base RAMs. The low nibble of the Base Pointer is a 4-bit binary counter which increments on each write to a Base Register, eliminating the need to redefine the Base Register Address for each write. The Base Register Address is not affected on a read operation and the address must be respecified on each read.

2.7.4.2 Writing to the Base Registers

Writing to a 24-bit Base Register from a 16-bit microprocessor is accomplished through the use of a holding register(the TEMP Register). The least significant 8 bits of data is loaded into the TEMP Register. The upper 16 bits of data and the contents of the TEMP Register are then simultaneously written to the Base RAMs.

2.7.4.3 Reading from the Base Registers

As noted in the discussion of the Base Pointer, the Base Register Address must be loaded into the Base Pointer prior to each read. The problem of reading a 24-bit value into a 16-bit machine is solved by providing two separate data sources for the read operation. One of the sources provides the lower 16 bits of the Base Register content, and the other source provides the upper 8 bits. Note that two machine cycles are required to examine the entire content of one Base Register.

2.7.4.4 Base Adders

The Base Adders allow for rapid addition of the offset to the base value without the involvement of the microprocessor. Carry Lookahead Generators provide for full synchronous carry across the 24-bit value.

2.7.5 Physical Address

Bit format for the 24-bit Physical Address is shown in Figure 2-13.

XXBBMCCCCCCCCRRRRRRRROOO

X - Expansion bits for 256K RAMS
B - Board select
M - Module select
C - Column address
R - Row address
O - Byte offset

FIGURE 2-13
Physical Address

For addressing purposes, the Q64 memory may be conceptualized as 64K blocks of 8 byte words(see Figure 2-14). The byte offset from the physical address identifies the byte addressed within the word.

With byte addressable memory, the possibility exists that the data on a read of more than one byte will be located in two memory words. For example, six bytes may be read starting in Word N, Byte 3(see Figure 2-14). The sixth byte of data will be located in Word N+1, Byte 0. To facilitate such circumstances, the physical address is incremented by one. The starting address, the incremented address(A+1 address) and the address range(total bytes to be read or written) are all sent to the Mem64A where they are decoded and the appropriate enables are generated. The same principles apply to write functions.

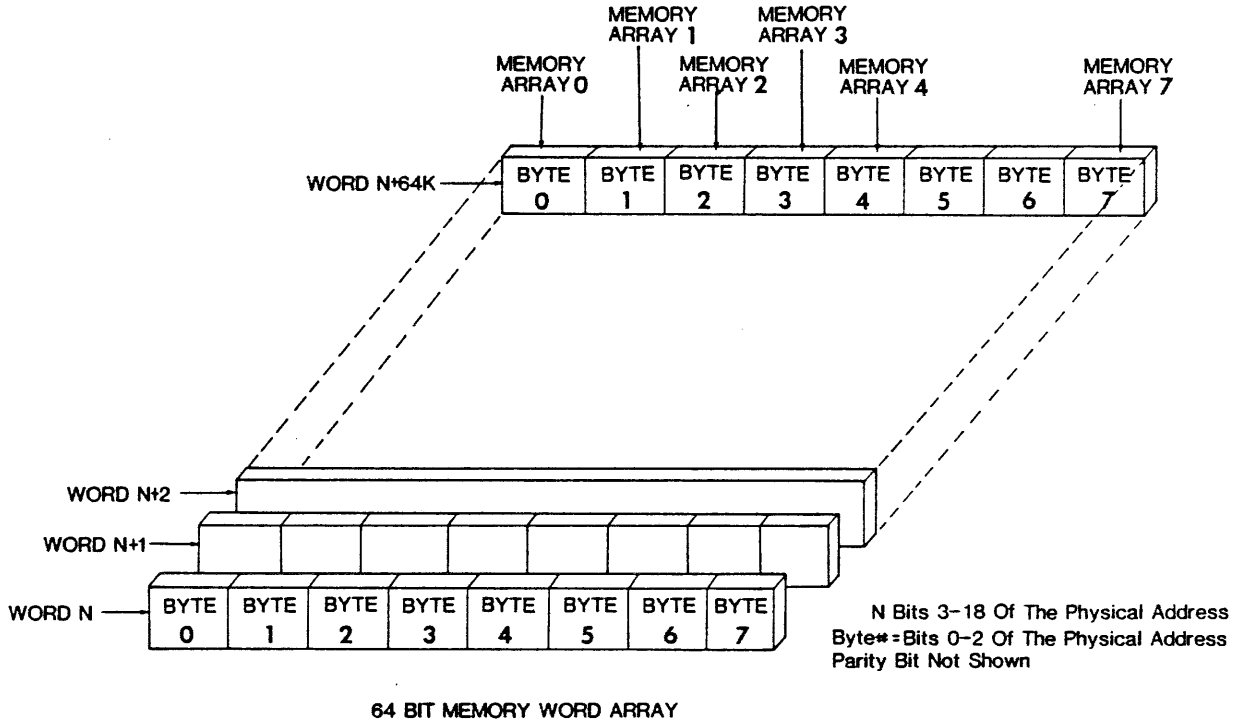


FIGURE 2-14

2.7.5.1 Physical Address Registers

Three of the physical address registers are paired with logical address registers (PARA, PARB and PARP) and are loaded(mapped) by the outputs of the Base Adders. A fourth physical address register, the "Real" Address Register (RAR) provides for direct memory access by the microprocessor.

To facilitate the parallel increment and decrement of the physical address with the corresponding logical address, the three least significant bits of the physical address are also implemented using a combination of a register and a ROM, while the remaining bits are loaded into counters. The counters allow the increment or decrement to be cascaded to the higher bits in the address.

2.7.5.2 Mapping the Physical Address Registers

The three physical address registers that are paired with the logical address registers are mapped subsequent to the loading of the corresponding logical address registers. The load enable for the logical address register generates the map enable for the physical address register.

A physical address register is also mapped when a base fault is detected in the increment or decrement of the corresponding logical register. Mapping or remapping occurs at T3.5H, and the address is valid on the trailing edge of T3.

Mapping of the Real Address Register requires two microcycles and is tied to the T3H clock signal. The most significant 8 bits of the address are first loaded into a temporary register (TEMP). On the following microcycle, the lower 16 bits of the address is loaded into RAR from the Y Bus concurrently with the output of the TEMP Register. The load enable for RAR is generated at T3H and the address is valid on the trailing edge T3H.

2.7.5.3 Increment/Decrement of Physical Addresses

Increment or decrement of the three physical address registers that are paired with logical address registers occurs concurrently with the corresponding logical address register. Note again that the Fetcher can only increment or decrement PARP.

The concurrent increment or decrement of register pairs allows the physical address to track the logical address without the need to always utilize the Base Register mapping process.

Increment and decrement of the physical address registers are also controlled by a lookup ROM. The increment select field is the same for the physical address registers as it is for the logical address registers.

Increment or decrement of the physical address is not tied to a clock. It is solely dependent upon the increment and address register controls from the microprocessor or the Fetcher which are generated during T0. Increment or decrement of the physical address occurs whether or not there is a need for remapping due to a base fault from the corresponding logical address register. If remapping is required, the incremented or decremented physical address is overwritten by the resolved address from the Base Adders at T3.5.

RAR is incremented or decremented directly by the memory controls from the microprocessor or the Fetcher. The Base Register is not used to map RAR.

If the physical address is within 16 bytes of a 64K block of memory, a Bank Hazard is detected. Bank Hazards are handled by the microcode and may require reloading of the Base Register as well as remapping of the physical address register.

2.8 Map Control and Timing

The use of multiple address registers creates contention for the mapping resources of the Base Register. Map Controls arbitrate the requests by establishing priorities. The priority sort is done using time delays. The interval between T0 and T2 is divided into half clocks: T0.5, T1.0, T1.5 and T2.0. The following is a description of the priorities for each clock:

T0.5 This request is for an address that must be used in a memory cycle that will take place in the current microcycle. If the memory cycle were allowed to occur, the results would be disastrous since the address in the physical address register is wrong. The microcycle is suspended and the memory cycle is aborted through a clock and map hold mechanism while the physical address register is mapped. setting a pending flag.

T1.0 All deferred(pending) requests are resolved. There may be more that one pending request. Priority within this level has been abritrarily assigned to A, then B, then P. Setting the request latch clears the pending flag.

T1.5 Remapping due to base fault is resolved. Since only one address register can do this at one time, there is no further prioritization.

T2.0 Mapping of physical address register paired with the logical address register which was loaded at T1.5 (from Fetcher or ALU Bypass). This allows mapping to occur in last half of the same microcycle.

Figures 2-15, 2-16 and 2-17 are timing diagrams for three common mapping conditions.

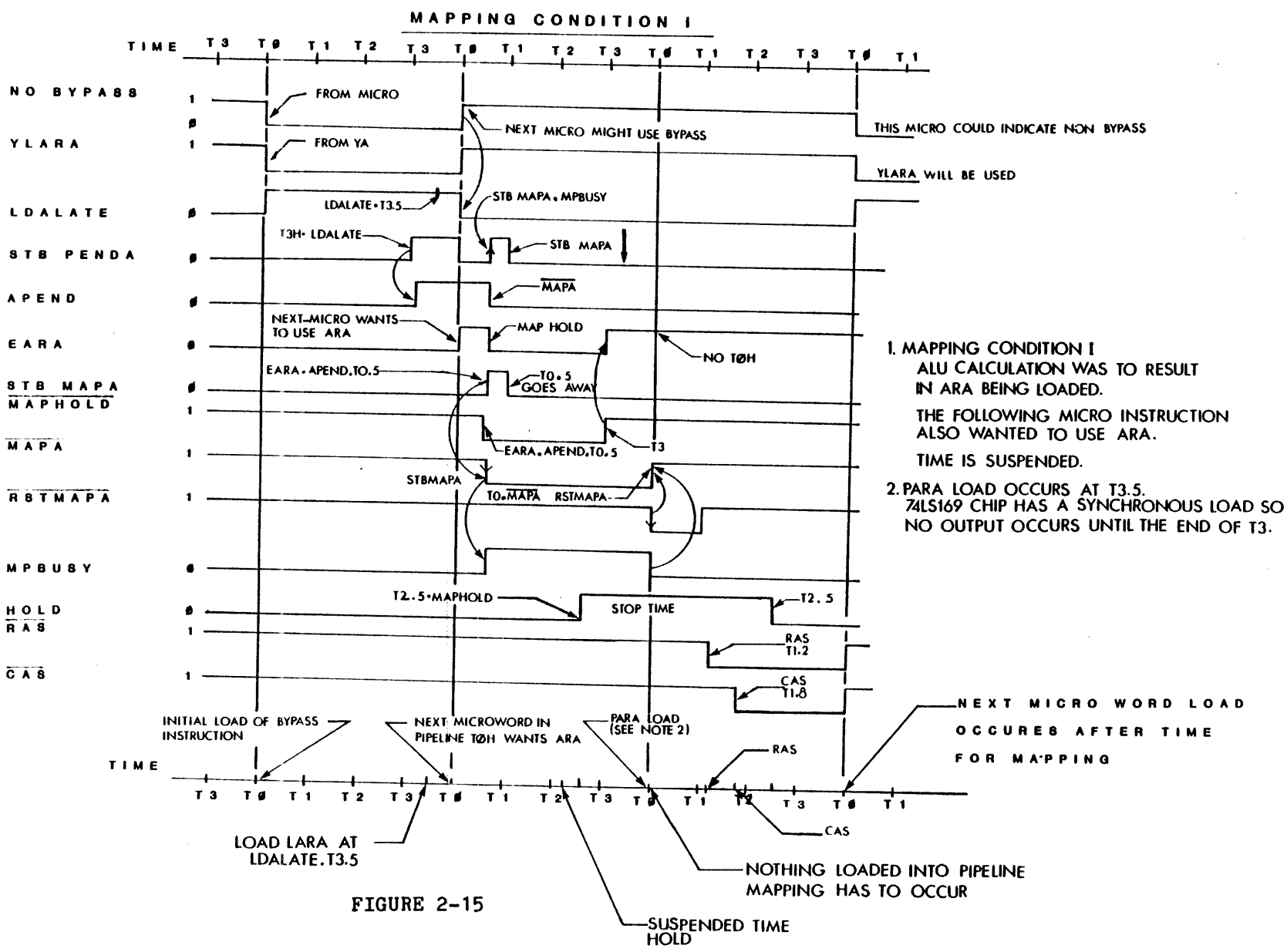
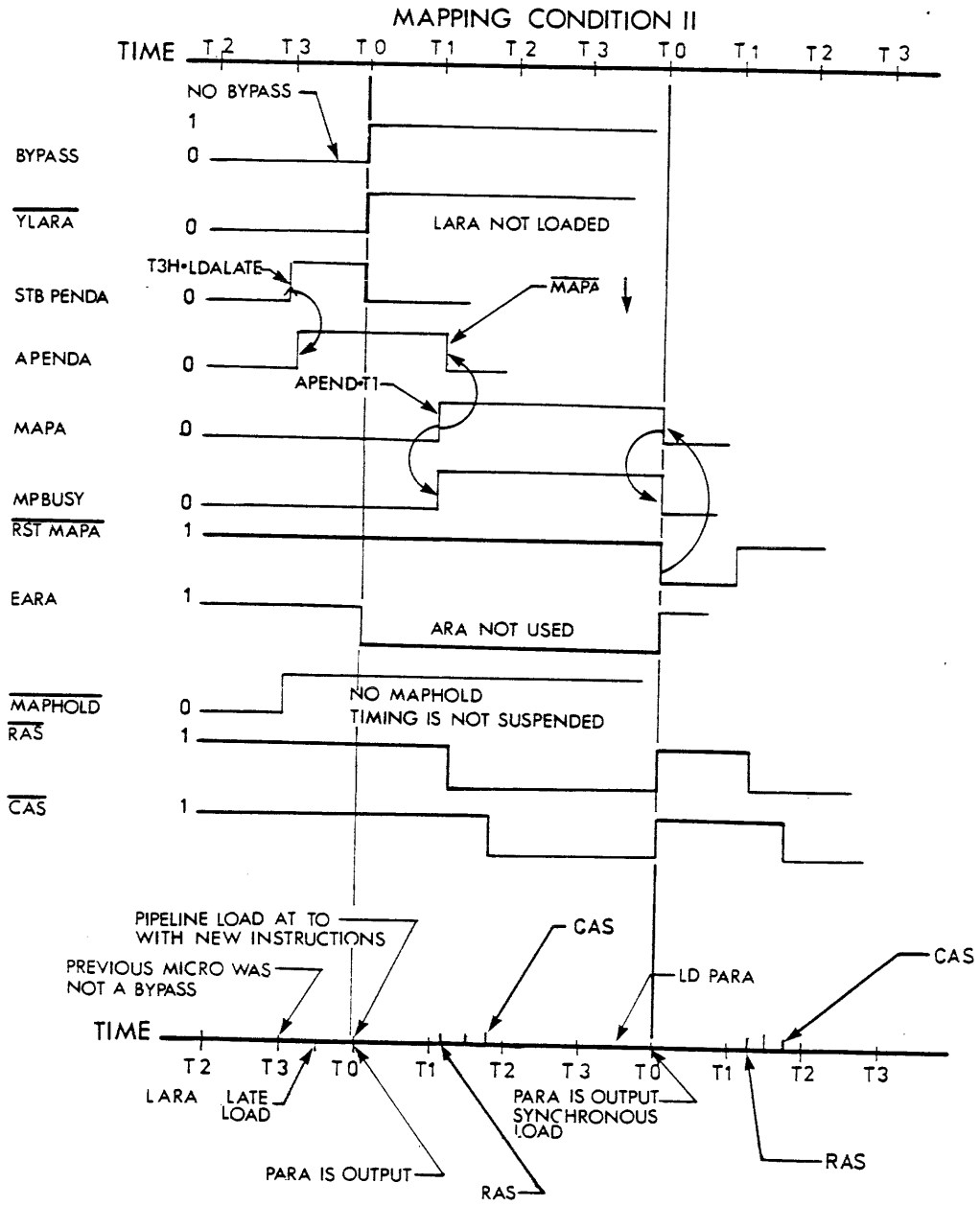
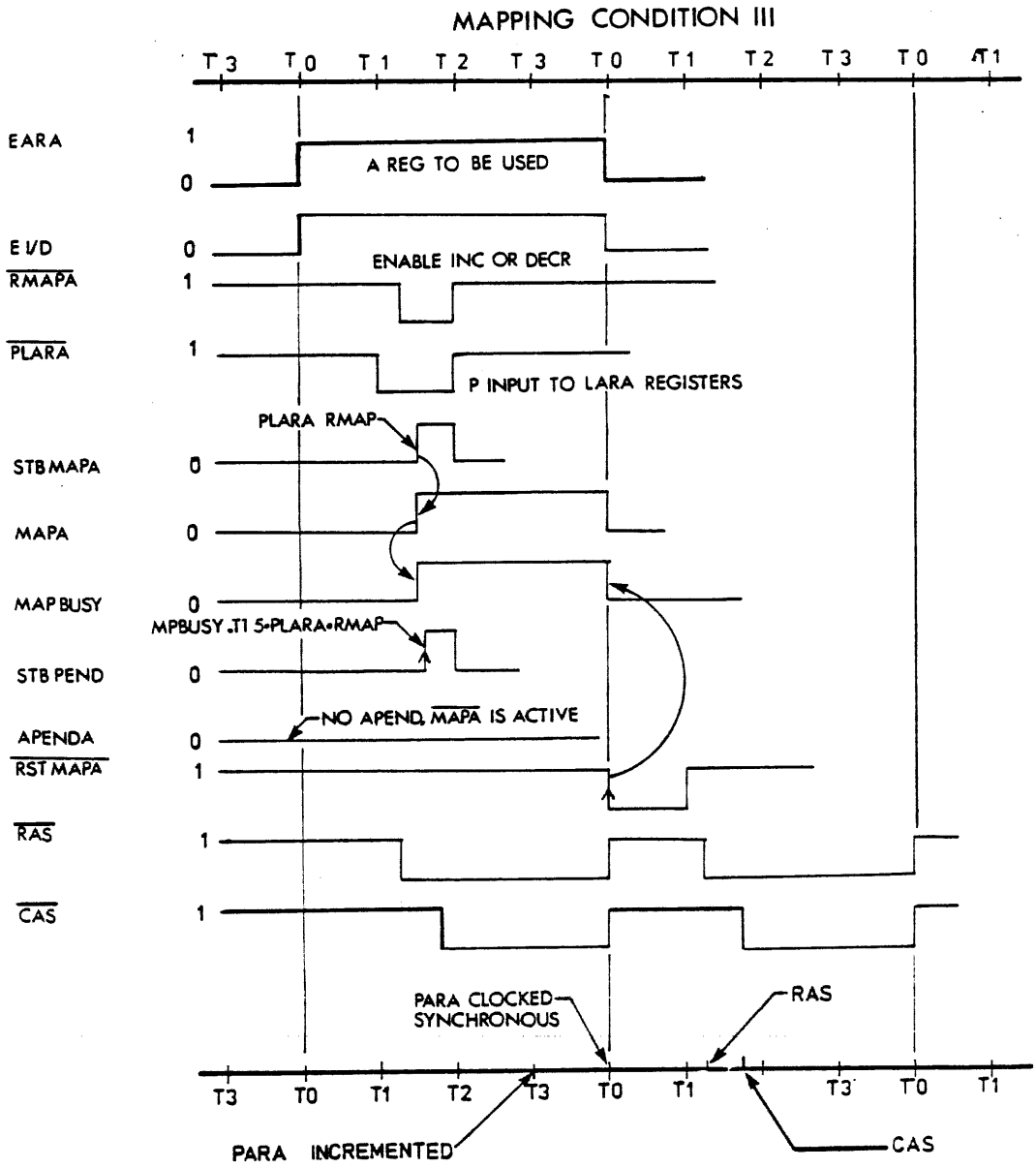

MAPPING CONDITION I
TIME
T3 TØ T1 T2 T3 TØ T1 T2 T3 TØ T1 T2 T3 TØ T1
NO BYPASS
YLARA
LDALATE
STB PENDA
APEND
EARA
STB MAPA
MAPHOLD
MAPA
RSTMAPA
MPBUSY
HOLD
RAS
CAS
FROM MICRO
FROM YA
NEXT MICRO MIGHT USE BYPASS
THIS MICRO COULD INDICATE NON BYPASS
YLARA WILL BE USED
LDALATE•T3.5
STB MAPA•MPBUSY
T3H• LDALATE
STB MAPA
MAPA
NEXT-MICRO WANTS TO USE ARA
MAP HOLD
NO TØH
EARA•APEND•T0.5
T0.5 GOES AWAY
EARA•APEND•T0.5
T3
STBMAPA
T0•MAPA RSTMAPA
T2.5•MAPHOLD
STOP TIME
T2.5
RAS T1.2
CAS T1.8
1. MAPPING CONDITION I
ALU CALCULATION WAS TO RESULT IN ARA BEING LOADED.
THE FOLLOWING MICRO INSTRUCTION ALSO WANTED TO USE ARA.
TIME IS SUSPENDED.
2. PARA LOAD OCCURS AT T3.5. 74LS169 CHIP HAS A SYNCHRONOUS LOAD SO NO OUTPUT OCCURS UNTIL THE END OF T3.
INITIAL LOAD OF BYPASS INSTRUCTION
NEXT MICROWORD IN PIPELINE TØH WANTS ARA
PARA LOAD (SEE NOTE 2)
NEXT MICRO WORD LOAD OCCURES AFTER TIME FOR MAPPING
TIME
RAS
CAS
LOAD LARA AT LDALATE.T3.5
NOTHING LOADED INTO PIPELINE MAPPING HAS TO OCCUR
SUSPENDED TIME HOLD

FIGURE 2-15



MAPPING CONDITION II
PREVIOUS MICRO USED ARA.
THIS MICRO DOES NOT USE ARA.
TIMING IS NOT SUSPENDED.
RAS AND CAS OCCURS NORMALLY BECAUSE
MAPHOLD IS NOT TURNED ON.

FIGURE 2-16



MAPPING CONDITION III
RMAP IS ACTIVE INDICATING LARA IS LOCATED AT A 2K BOUNDRY.
THE LOGICAL ADDRESS REGISTER IN THIS MICRO CYCLE IS NOT IN USE.
INCREMENT PAST THE PAGE BOUNDRY

FIGURE 2-17

2.9 MEMORY DATA INTERFACE

As noted previously, the Q64 memory is organized in arrays of 64-bit memory "words"(refer to Figure 2-14). Each byte within the memory word is directly addressable by the Q64 CPU.

2.9.1 Byte-wise Barrel Shifting

The 64-bit byte addressable memory word necessitates some manipulation of the data to ensure proper allignment. Any time more than one memory word is accessed in one operation, the direct readout will not be in the proper sequence. On writes to memory, the data must also be shifted to ensure that it is in the proper sequence to be written in sequential memory locations.

Figure 2-18 contains two graphic representations of "byte-wise barrel shifting" used to allign the data on a read from memory. The memory offset is defined by the memory increment and control logic, as is the range(number of bytes read). An offset of 0 and a range of 8 would indicate that the starting address is byte 0 in the memory word and that 8 bytes will be read.

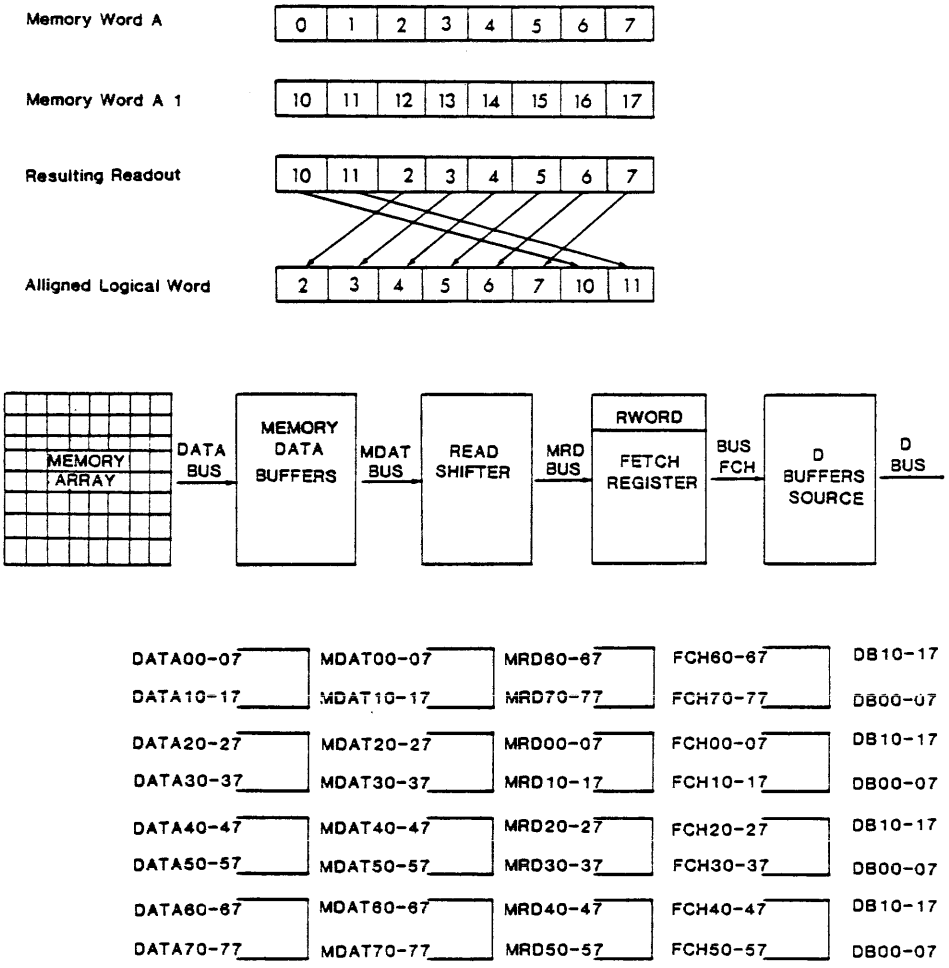


FIGURE 2-18
Read Barrel Shift

In the example in Figure 2-18, the offset is 2 and the range is 8. The starting address of the operation is Byte 2. The last six bytes of Word A and the first two bytes of Word A+1 will be read from memory. The reallignment of the data is shown in the upper section of Figure 2-18. The lower section of Figure 2-18 shows the data flow from the memory array through to the D Bus. Below the block diagram is a listing of the bit designation(in octal) of the data on each of the busses in the data path(with the exception of the D Bus which is 16-bits wide, all of the busses es shown are 64-bits wide. Busses are discussed in more detail in Section 4.1). Note the 2 byte shift in bit designation between the MDAT Bus and the MRD Bus. MDAT20-27 corresponds to the data from Byte 2 in Memory Word A(the starting address), and it is shifted to the least significant byte on the MRD Bus(MRD00-07).

A similar reallignment of data takes place on a write and is shown in Figure 2-19.

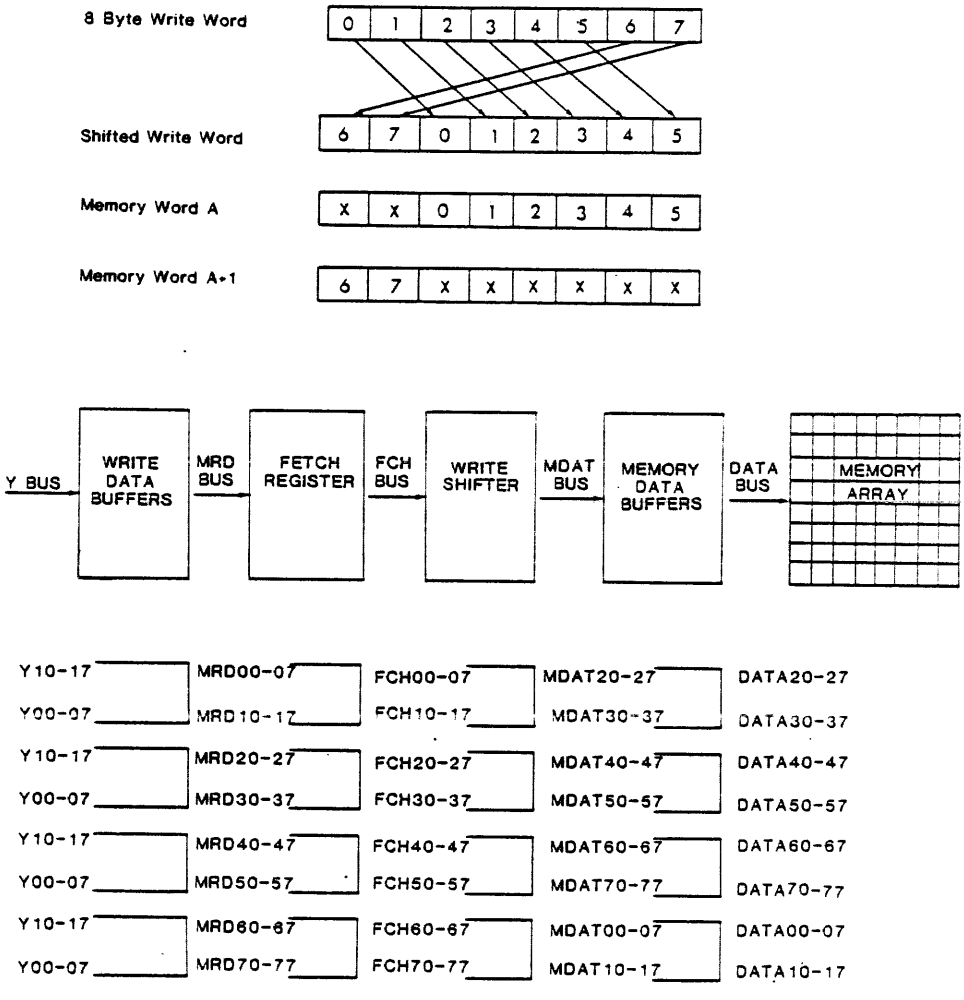


FIGURE 2-19
Write Barrel Shift

2.9.2 Parity

A parity bit is generated for each byte written to memory. The number of high bits in a byte are summed, and if it is even, the parity bit will be high, generating odd parity. On a read from memory, each byte and the parity bit generated on write are input to a parity detector. If the sum of the bits of a byte and the parity bit is even, then a parity error is detected and the starting address of the memory operation is latched.

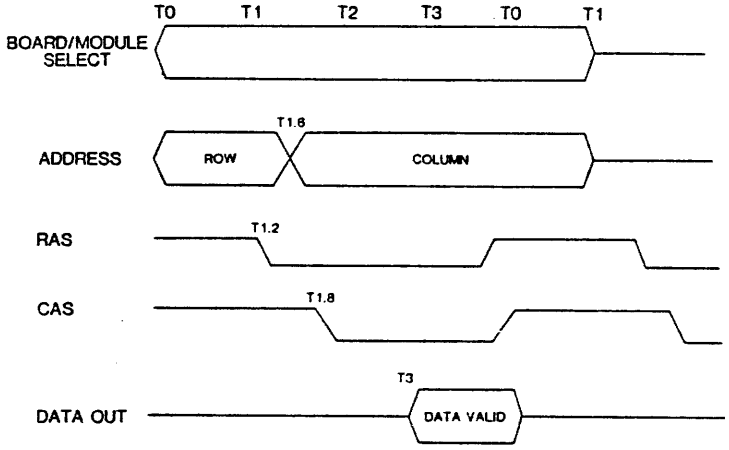


FIGURE 2-20
Memory Interface Timing
Read Cycle

2.9.3 Memory Data Timing

As noted previously, the Fetch Register is also the Read and Write Data Register for memory operations. The Fetch Register is clocked at T0 and T2; however, data to or from memory is valid only at T0.

Figures 2-20 and 2-21 show the timing of memory data at the backplane connector to the Memory Data Board which contains the Fetch Register.

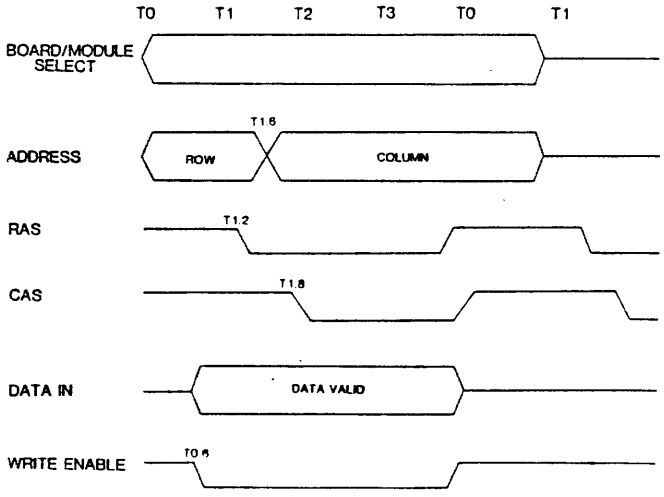


FIGURE 2-21
Memory Interface Timing
Write Cycle

3.0 Q64 MICROWORD

The microword is 64 bits in length. It consists of three parts: the Arithmetic/Logic Section; Discrete Functions Section; and the Next Address Section. Figure 3-1 is a composite picture of the microword.

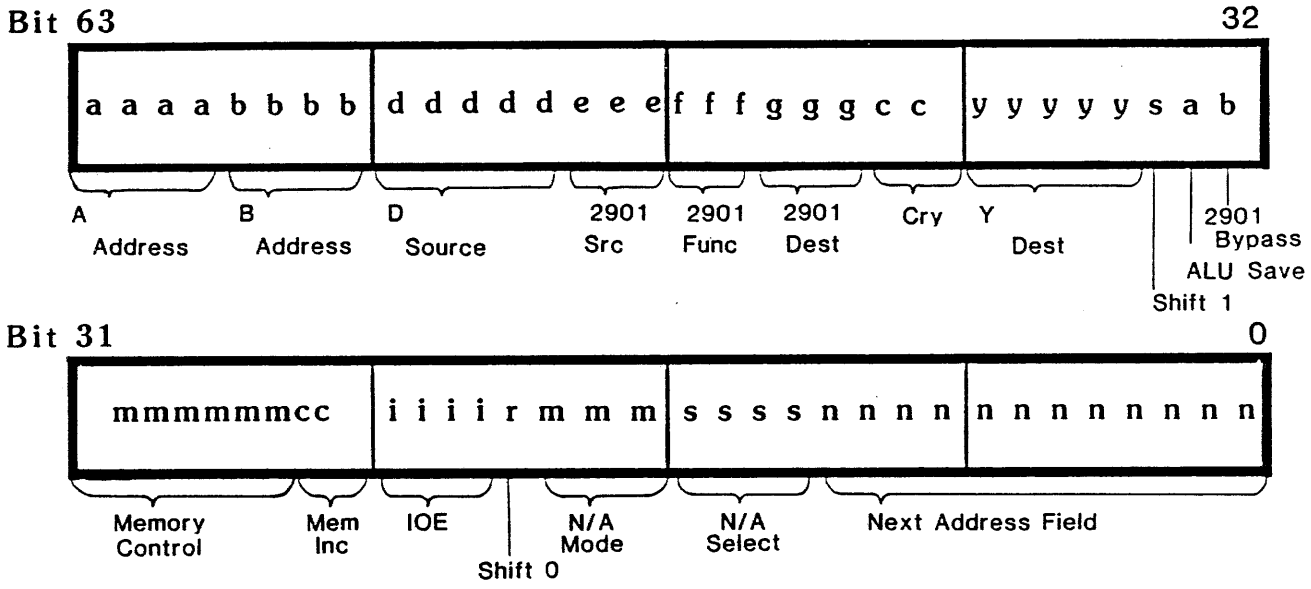


FIGURE 3-1
Q64 Microword

The Arithmetic/Logic Section controls the manipulation of the ALU. The Discrete Function Section controls miscellaneous functions of the micro-processor, such as memory read/write, status saving, etc. The Next Address Section directs program flow. It is also used to insert literals into the system.

In the discussion which follows, the specific fields will be described together with some assembly language mnemonics used in the microinstruction set. Some assembly language mnemonics are included to provide some assistance in reading the diagnostic microcode listings(such as Board Bug). The mnemonics are designed to optimize REAL emulation. While there are no microcode manuals available, a reader with some knowledge of REAL and the information discussed in this section should be able to review microcode listings with little difficulty.

The fields are discussed in descending order of significance in the Microword (from left to right), paralleling the object code generated by the assembler.

Following the discussion of the specific fields of the Microword are some simple examples of microcode which are examined. A thorough understanding of the microcode is necessary to understand the Theory of Operation of the Q64 CPU.

3.1 ARITHMETIC/LOGIC SECTION

This section controls part of the Control Store, the ALU, source addressing, destination addressing and the pipeline register. An instruction from the Control Store goes through the pipeline first. The arithmetic part of an instruction from the Control Store is executed one clock cycle after the "next instruction" part is executed. When the instruction is delayed for a clock cycle, this is called a "pipelined" operation.

3.2 ARITHMETIC/LOGIC FIELDS

The Arithmetic/Logic Fields of the microinstruction control the manipulation of data through the ALU. These fields are specified by an expression to be evaluated and the destination(s) for the result. From the statement, the Assembler generates a number of sub-fields in the microword which are listed below and discussed in the following paragraphs.

A Address	Selects one of the sixteen 16-bit file registers internal to the ALU.
B Address	Same as A Address; may also be used as the destination.
D Source	Selects the external data source to be placed on the D Bus to the ALU.
2901 Source	Controls what combination of A, B, Z, and Q is used in the ALU function.
2901 Function	Selects the function to be performed by the ALU.
2901 Destination	The results of the ALU operation can be routed to several destinations defined later. This may include a shift operation.
Carry Control	Determines the carry input to the ALU.
Y Destination	Each instruction places either the A Source, the D Source, or the ALU result on the Y output which is routed to the destination specified by this field.
Shift Control	Determines the value of the bits vacated by shift operations.

3.2.1 A And B Addresses

These 4-bit fields are used to determine which of the sixteen available registers are to be operated by the ALU. Both addresses may point to different registers at the same time; however, only the B address will define a "write-to"

register. They are designated by the names R0, R1,..., RF. If only one register is specified, the Assembler generates a value for both A and B.

3.2.2 D Source

This is a 5-bit field that is used to address the various sources of data that may be accessed by the ALU. They select the source of a 16-bit value to be presented on the D Bus to the ALU. Table 3-1 is a list of D Sources showing the coding(in Hex), the Assembler mnemonic and the D Source.

CODE	MNEMONIC	D SOURCE
$00	LIT	Literal Register
$01	SWAP	Byte Swap Register
$02	LCL	Local Control Register
$03	UDATA	Data and Status from UART Interface
$04	IODATA	Data from I/O backplane
$05	THREE	Constant $0003
$06	BCDRES	BCD Adder Result Register
$07	TIMER	Programmable Interval Timer
$08	WCSD	WCS Data
$09	FRPNL	Front Panel Data
$0A		Not used
$0B	FLAGS	Qantel Flag Register
$0C	RWORD	Read Word from memory
$0D	RWORD2	2nd Read Word from memory
$0E	RWORD3	3rd Read Word from memory
$0F	RWORD4	4th Read Word from memory (aka OP3)
$10	OP1	Operand 1
$11	OP2	Operand 2
$12	CODE1	Opcode 1
$13	CODE2	Opcode 2
$14	LCAS	Cascade Register(left to right read)
$15	RCAS	Cascade Register(right to left read)
$16	RBYTE	Read Byte from memory
$17	PERR	Parity Error Address Register
$18	LARA	Logical Address Register A
$19	LARB	Logical Address Register B
$1A	LARP	Logical Address Register P (Program Counter)
$1B	LARPO	Old Logical Address Register P
$1C	BASEL	Lower eight bits Base Register
$1D	BASE	Upper 16 bits Base Register
$1E	MOVLEN	Length from MOV instruction formed from opcode bytes
$1F	FFFF	Constant $FFFF

TABLE 3-1
D Source Coding

3.2.2.1 Literal Register(LIT)

The 16-bit Literal Register contains the constant previously loaded by the microcode(see Next Address Section 3.5). The literal can be used in the same statement that loads it.

3.2.2.2 Byte Swap Register(SWAP)

This register contains the result of exchanging the two halves of a 16-bit word. For example, if $1234 is output to SWAP, $3412 will be returned upon reading SWAP. It can be used in any statement following the output to the swapper.

3.2.2.3 Local Control Register(LCL)

This 16-bit register is the readback of the local resource control register. Table 3-2 lists the bit definitions.

BIT	DEFINITION
2**15	Not I/O Mode
2**14	Interrupt Daisy Chain(Not implemented)
2**13	Halt
2**12	Interrupt Acknowledge(Not implemented)
2**11	Set P-code mode(Not implemented)
2**10	Enable Macro interrupts
2**9	Set Stop light
2**8	Single step mode
2**7	Enable Fetcher decode
2**6	Stop Fetcher
2**5	Reset Programmable Interrupt Timer
2**4	Disable Programmable Interrupt Timer
2**3	Reset parity error latch
2**2	Start WCS readback
2**1	WCS Readback Register select bit 1
2**0	WCS Readback Register select bit 0

TABLE 3-2
Local Control Register
Bit Definition

The WCS Readback Register select bits are used to determine which part of the microword is input by accessing WCS data. The bit format is shown in Section 3.2.2.9.

3.2.2.4 UART Data Register(UDATA)

The UART Data Register contains both the status of the UART and any received data. Table 3-3 lists the bit definition.

BIT	DEFINITION
2**15	Data Set Ready
2**14	Clear to Send
2**13	Able to Transmit
2**12	Transmit Interrupt
2**11	Receive Interrupt
2**10	Receive overrun
2**9	Receive Framing Error
2**8	Receive Parity Error
2**0...2**7	Receive Data

TABLE 3-3
UART Data Register
Bit Definition

3.2.2.5 I/O Received Data Register(IODATA)

This register contains the last data or status read from the input/output unit (IOU)interface. The register is 16 bits for the Q64 Expansion Controllers(not implemented) and 8 bits for the old controllers.

3.2.2.6 Constant 3(THREE)

This register contains the constant $0003.

3.2.2.7 BCD Adder Result(BCDRES)

This register contains the result of the BCD Adder. It is always in the form $003d, where d is the result digit 0-9.

3.2.2.8 Programmable Interval Timer(TIMER)

The Programmable Interval Timer is a 16-bit counter which can be loaded by the microcode. It is decremented after each microcycle(400NS) and generates a micro interrupt when it underflows. The counter continues to run after the interrupt is posted so that the exact amount of time that has elapsed can be determined. The timer readback register contains the current timer value remaining before an interrupt. The overflow value is presented as a two's complement number.

3.2.2.9 Writeable Control Store Data(WCSD)

This 16-bit register contains the data read from WCS. Which 16 of 64 bits from the microword are read is determined by bits 1 and 0 in LCL. Table 3-4 lists the coding of the LCL bits and the corresponding WCS Word read(the microword bit designations are shown in parenthesis).

CODE	WORD SPECIFIED
00	WCS Word 1(2**63 - 2**48)
01	WCS Word 2(2**47 - 2**32)
10	WCS Word 3(2**31 - 2**16)
11	WCS Word 4(2**15 - 2**0)

TABLE 3-4
WCS Select Bit
Coding

3.2.2.10 Front Panel Register(FRPNL)

This register contains the 8-bit data input from the Diagnostic Panel.

3.2.2.11 Qantel Flag Register(FLAGS)

This is the readback register for the macrocode condition flags. Table 3-5 lists the bit definition.

BIT	QANTEL FLAG
2**2	Overflow
2**1	Minus
2**0	Nonzero

TABLE 3-5
Qantel Flag Register
Bit Definition

3.2.2.12 Memory - Fetch Register

The Fetch Register is a 64-bit read data register. When data is read from memory into the Fetch Register, the addressed byte is placed in byte 0, address+1 in byte 1, etc. Table 3-6 is a list of the various combinations of D Sources available to the microcode through the Fetch Register.

MNEMONIC	D SOURCE	BYTE(S)
OP1	Operand 1	Bytes 0 and 1
CODE1	Opcode 1	Byte 2
OP2	Operand 2	Bytes 3 and 4
CODE2	Opcode 2	Byte 5
OP3	Operand 3	Bytes 6 and 7
RWORD2	Read Word 2	Bytes 2 and 3
RWORD3	Read Word 3	Bytes 4 and 5
RWORD4	Read Word 4	Bytes 6 and 7

TABLE 3-6
Fetch Register
D Sources

3.2.2.13 Memory Read Word(RWORD)

This register is separate from the Fetch Register but is loaded whenever the Fetch Register is loaded. However, it is also possible to load this register without altering the contents of the Fetch Register.(It may or may not contain Operand 1).

3.2.2.14 Memory Read Byte(RBYTE)

This register contains the addressed byte read from memory. It is always $00rr, where RWORD contains $rrxx.

3.2.2.15 Memory Read Cascade Registers(RCAS, LCAS)

These registers contain the last two bytes read from memory. If RBYTE contains $00aa and the next byte is $00bb, then RCAS contains $bbaa and LCAS contains $aabb.

3.2.2.16 Memory Address Registers(LARA, LARB, LARP, LARPO, RAR & PERR)

The contents of the logical address registers can be read by accessing LARA, LARB, and LARP. LARP is used as the Qantel Program Counter(PC), while LARA and LARB are the A and B operands of an instruction. LARPO contains the last valid Qantel PC(old contents of LARP) when the Fetcher is operating. PERR is used to read the Parity Error address or the last real memory address used, since PERR always tracks any memory reference. The least significant bit of the Local Control Register is used to select the lower 16 bits of the register(if LCL0=0) or the upper 8 bits(if LCL0=1).

3.2.2.17 Base Register Contents(BASE, BASEL)

These registers contain the 24-bit contents of the base register pointed to by the Base Pointer(BPTR). BASE contains the upper 16 bits, and BASEL contains the lower 8 bits.

3.2.2.18 Move Length(MOVLEN)

The Move Length Register is loaded from bytes 2 and 5 of the Fetch Register and contains $00ab where Byte2 = $xa and Byte5 = $yb. If a and b are 0s, then MOVLEN contains $0100.

3.2.2.19 Minus One(FFFF)

This register contains the constant $FFFF.

3.2.3 2901 Source

This 3-bit field selects which of the possible sources available to the ALU in the 2901 will be used for the present operation. Table 3-7 lists the coding of the 2901 Source Field.

CODE	MNEMONIC	INSTRUCTION
000	AQ	Use the register specified by A and Q register.
001	AB	Use the registers specified by A and B.
010	ZQ	Use the constant 0 and Q register.
011	ZB	Use the constant 0 and register specified by B.
100	ZA	Use the constant 0 and register specified by A.
101	DA	Use external data and register specified by A.
110	DQ	Use external data and Q register.
111	DZ	Use external data and constant 0.

TABLE 3-7
2901 Source
Coding

3.2.4 2901 Function

This 3-bit field is used to determine the operation that will be performed on the data sources that have been selected. Table 3-8 lists the coding of the 2901 Function Field.

CODE	MNEMONIC	INSTRUCTION
000	R+S	Add R to S.
001	S-R	Subtract R from S.
010	R-S	Subtract S from R.
011	R.S	Logical OR R and S.
100	R&S	Logical AND R and S.
101	^R&S	Complement R, logical AND with S.
110	R:S	Logical Exclusive OR R and S.
111	R#S	Complement of (Exclusive OR R and S).

TABLE 3-8
2901 Function
Coding

Table 3-9 summarizes the unique expressions which can be performed by the 2901. It also shows what value the Assembler will generate for a given expression.

FUNCTION	SOURCE AQ	AB	ZQ	ZB	ZA	DA	DQ	DZ
R+S								
(c=0)	a+Q	a+b	Q	b	a	d+a	d+Q	d
(c=1)	a+Q+1	a+b+1	Q+1	b+1	a+1	d+a+1	d+Q+1	d+1
(c=C8,C16)	a+Q+c	a+b+c	Q+c	b+c	a+c	d+a+c	d+Q+c	d+c
S-R								
(c=0)	Q-a-1	b-a-1	Q-1	b-1	a-1	a-d-1	Q-d-1	-d-1
(c=1)	Q-a	b-a				a-d	Q-d	-d
(c=C8,C16)	Q-a-c	b-a-c	Q-c	b-c	a-c	a-d-c	Q-d-c	-d-c
R-S								
(c=0)	a-Q-1	a-b-1	-Q-1	-b-1	-a-1	d-a-1	d-Q-1	d-1
(c=1)	a-Q	a-b	-Q	-b	-a	d-a	d-Q	
(c=C8,C16)	a-Q-c	a-b-c	-Q-c	-b-c	-a-c	d-a-c	d-Q-c	d-c
R.S	a.Q	a.b				d.a	d.Q	
^R&S	^a&Q	^a&b				^d&a	^d&Q	
R:S	a:Q	a:b				d:a	d:Q	
R#S	a#Q	a#b	^Q	^b	^a	d#a	d#Q	^a

TABLE 3-9
Combined 2901
Coding

3.2.5 2901 Destination

This 3-bit field determines the disposition of the data that has been manipulated by the ALU. The destination of the ALU is implicitly descr- ibed by the form of the source statement. Also available is a "2901 Bypass" in which the D Source is directly routed to a Y Destination.

Table 3-10 lists the coding of the 2901 Destination Field. If no Y Destination is specified, the Assembler will generate the "NOP" destination.

CODE	INSTRUCTION
000	Result of ALU is placed into the Q register and on the Y output.
001	Result of ALU is placed on Y output.
010	Result of ALU is placed into B Source register. A Source is placed on Y output.
011	Result of ALU is placed into B Source register and on Y output.
100	Result of ALU is placed on Y output. The result is shifted right one bit and placed into B Source register. Q is shifted right(double right shift).
101	Result of ALU is placed on Y output. The result is shifted right one bit and placed into the B Source register(single right shift).
110	Result of ALU is placed on Y output. The result is shifted left one bit and placed into the B Source register. Q is shifted left(double left shift).
111	Result of ALU is placed on Y output. The result is shifted left one bit and placed into the B Source register(single left shift)

TABLE 3-10
2901 Destination
Coding

3.2.6 Carry Control

This 2-bit field does not directly control the 2901; however, its content does affect the behavior of the 2901. This field controls the state of the carry input to the 2901 ALU. It is generated by the Assembler while examining the ex- pression field. Table 3-11 lists the coding of the Carry Control Field.

CODE	MNEMONIC	MEANING
00	0	Forced 0 into carry input.
01	1	Forced 1 into carry input.
10	C16	Saved 16-bit carry output into carry input.
11	C8	Saved 8-bit carry output into carry input.

TABLE 3-11
Carry Control
Coding

3.2.7 Y Destination

This 5-bit field directs the 2901 result(the Y output) to some external target. At most one Y Destination may be specified per statement. Table 3-12 lists the coding of the Y Destination Field.

CODE	MNEMONIC	Y DESTINATION
$00	NOP	No target
$01	BRR	Branch Register
$02	WCDS1	WCS Data Word 1
$03	WCSD2	WCS Data Word 2
$04	WCSD3	WCS Data Word 3
$05	WCSD4	WCS Data Word 4*
$06	WCSA	WCS Address and page
$07	LCL	Local Control Register
$08	UDATA	UART Data
$09	UCTL	UART Control Register
$0A	JUNK	Junk Register
$0B	SWAP	Byte Swap Register
$0C	GPC	16-bit General Purpose Counter
$0D	GPC2	4-bit General Purpose Counter
$0E	BCD1	BCD Adder Operand 1
$0F	BCD2	BCD Adder Operand 2**
$10	PAGE	WCS Page Register
$11	TIMER	Programmable Interval Timer
$12	IODATA	Data to backplane
$13	FLAGS	Qantel Flag Register
$14	WWORD	Word 1 in Fetch Register
$15	RWORD2	Word 2 in Fetch Register(aka WWORD2)
$16	RWORD3	Word 3 in Fetch Register(aka WWORD3)
$17	RWORD4	Word 4 in Fetch Register(aka WWORD4)
$18	LARA	Logical Address Register A
$19	LARB	Logical Address Register B
$1A	LARP	Logical Address Register P
$1B	RAR	Real Address Register(lower 16 bits)
$1C	TEMP	Temporary Register (actually 2 - one for upper 8 bits of Real Address Register and one for lower 8 bits of Base Register)
$1D	BPTR	Base Register Pointer
$1E	BASE	Base Register(upper 16 bits)
$1F	FRPNL	Front Panel

TABLE 3-12
Y Destination
Coding

3.2.7.1 No Operation(NOP)

No register is modified.

3.2.7.2 Branch Register(BRR)

The Branch Register accepts 8 bits of data to be used in Next Address computation. The microcode may branch on part or all of the contents of the Branch Register.

3.2.7.3 WCS Data Registers(WCSD1, WCSD2, WCSD3 & WCSD4)

The four WCS Data Registers accept 16-bit words from the ALU to be written to WCS. Loading the fourth register(WCSD4) implicitly starts the write to the address specified by WCSA.

3.2.7.4 WCS Address Register(WCSA)

This register accepts a 16-bit address for WCS. It is used to read from and write into WCS.

3.2.7.5 Local Control Register(LCL)

The Local Control Register is used to control a number of miscellaneous functions. The bits are defined in Table 3-2.

3.2.7.6 UART Data(UDATA)

This register accepts 8 bits of data from the ALU to be output to the UART.

3.2.7.7 UART Control(UCTL)

Bit definition of the UART Control Register is listed in Table 3-13.

BIT	DEFINITION
2**7	Not used.
2**6	Light IPL Light
2**5	I/O Service Request Enable: 0 forces I/O Service Request true for seek on IOU24; 1 is normal operation.
2**4	Reset UART: 0 is idle; 1 is reset.
2**3	Receive interrupt: 0 is disable; 1 is enable.
2**2	Reset transmit interrupt: 0 is reset; 1 is idle.
2**1	Reset data available: 0 is reset; 1 is idle.
2**0	Transmit data strobe: 0 is idle; 1 is transmit.

TABLE 3-13
UART Control Register
Bit Definition

3.2.7.8 "Junk" Register(JUNK)

The "Junk" Register(for lack of a better name) controls a number of miscellaneous functions pertaining to binary and decimal arithmetic.

Setting Junk to 0000 initializes addition by:

1. Setting saved C8, C16, and C10 to false(0);
2. Setting accumulated Z4 and Z8 to true(1); and
3. Setting BCD Adder to add mode.

Setting Junk to 0001 initializes subtraction by:

1. Setting saved C8, C16 and C10 to true(1);
2. Setting accumulated Z4 and Z8 to true(1); and
3. Setting BCD Adder to subtract mode.

NOTE: Unlike most Y Destinations, the Junk Register is not a physical entity such as a buffer or a latch. Enables for the functions described above are generated either directly by the Junk Register enable or the ANDing of the Junk Register enable and the least significant bit on the Y Bus.

3.2.7.9 Byte Swap Register(SWAP)

This register accepts 16 bits of data from the ALU and exchanges the upper and lower 8 bits. The result is available as the D Source SWAP.

3.2.7.10 General Purpose Counters(GPC & GPC2)

The 16-bit GPC(GPC) can be tested for either all zeros or low 4 bits equal to zero. The four-bit GPC(GPC2) can also be tested for zero.

Both counters are tested by means of the Next Address field of the microword and are decremented by 1 after each test. They cannot be tested until the second statement following the load.

3.2.7.11 BCD Adder(BCD1 & BCD2)

The BCD Adder performs decimal addition or subtraction on two 4-bit numbers and an input decimal carry. The first operand is input to BCD1 and the second operand to BCD2. When BCD2 is loaded, the operation is performed. The result is available in BCDRES.

3.2.7.12 Page Register(PAGE)

The WCS Page Register determines whether the operating microword originates from the Microstore ROM or from a 4K page of WCS. Table 3-14 lists the coding of the Page Register.

CODE	PAGE SPECIFIED
$00	Microstore ROM
$10	WCS Page 0 ($0000 - $0FFF)
$11	WCS Page 1 ($1000 - $1FFF)
$12	WCS Page 2 ($2000- $2FFF)
...	
$1F	WCS Page 15 ($F000 - $FFFF)

TABLE 3-14
Page Register Coding

3.2.7.13 Programmable Interval Timer(TIMER)

This 16-bit register defines the count for the Programmable Interval Timer. It is described in Section 3.2.2.8.

3.2.7.14 I/O Data(IODATA)

The I/O Data Register accepts either 8 or 16 bits of data to be output to the I/O interface.

3.2.7.15 Qantel Flag Register(FLAGS)

This register holds the value of the current Qantel condition flags so that the Fetcher interface may execute conditional branches dependent upon the flags. The flag register bits are defined in Table 3-5.

3.2.7.16 Write Data Registers(WWORD, WWORD2, WWORD3 & WWORD4)

The Fetch Register is also used as the write data path to memory. The microcode may specify that the entire Fetch Register be written to memory, or that all of WWORD, or just the low or high byte of WWORD be written to memory. Data must be output to WWORD or WWORD, WWORD2, WWORD3 and WWORD4 at least one statement before the write command.

3.2.7.17 Logical Address Registers(LARA, LARB & LARP)

The 16-bit Logical Address Registers are loaded with an 11-bit displacement value, a 4-bit Base Register index and an indirect address flag. Note that the indirect address flag is decoded by the Fetcher or microcode and is not used by the hardware. When LARx is loaded, the index and the Base Pointer select the appropriate Base Register and its contents are added to the displacement. This 24-bit value is loaded into the Physical Address Register associated with each LAR.

The logical-to-physical mapping process takes one microcycle and the LAR cannot be used until the second statement following the load. As an exception, the LAR may be used on the next statement following the load if loaded on a 2901 Bypass operation. The hardware will stall the microcode if it attempts to use an unmapped address register.

As noted previously, LARP is used by the microcode and the Fetcher as the Program Counter. LARA is used as the A(source) operand, and LARB is used as the B(destination) operand.

3.2.7.18 Real Address Register(RAR)

The 24-bit Real Address Register is loaded from the 16-bit Y Bus in two sequential operations. The high 8 bits are first loaded into a temporary register (TEMP) which outputs to the high byte of the RAR. The loading of the low 16 bits directly to the RAR then causes the contents of the temporary register to be loaded to the high byte of the RAR.

3.2.7.19 Base Registers - The Base Pointer(BASE - BPTR)

There are 256 24-bit Base Registers specified by an 8-bit pointer(BPTR). A Base Register is loaded in two sequential operations similar to the RAR. The low 8 bits of the value to be loaded to a Base Register are first loaded into the Temporary Register(TEMP) which outputs to the Base Register. When the high 16 bits are loaded directly to the Base Register, the low 8 bits are simultaneously loaded from TEMP. The Base Pointer is incremented(modulo 16) when a Base Register is loaded.

3.2.7.20 Front Panel Register(FRPNL)

This register accepts an eight bit value which is converted to hexadecimal and displayed on the Front Panel.

3.2.8 2901 Shift Control

This 2-bit field determines how the shift inputs and outputs are connected to the 2901. They are invoked by the mnemonic code in the discrete function field of the assembly source statement. The default is S0. Table 3-15 lists the coding of the Shift Control Field.

CODE	MNEMONIC	FUNCTION
00	S0	Logical(or Zero) mode Right,single 0 -> RAM -> SR Right,double 0 -> RAM ->QREG -> SR Left,single SR <- RAM <- 0 Left,double SR <- RAM <- QREG <- 0
01	S1	Cyclic Mode Right,single --> RAM -> SR Right,double --> RAM -> QREG -> SR Left,single SR <-- RAM <- Left,double SR <-- RAM <- QREG <-
10	S2	Arithemetic/One Mode Right,single --> RAM -> SR Right,double --> RAM -> QREG -> SR Left,single SR <- RAM <- 1 Left,double SR <- RAM <- QREG < 1
11	S3	Extended Mode Right,single -> RAM --> SR Right,double SR -> RAM -> QREG -> Left,single -- RAM <- SR Left,double -- RAM <- QREG <- SR <-

TABLE 3-15
2901 Shift Control Coding

CODE	MNEMONIC	MEANING
$00	(blank)	No request for memory activity.
$04	MA+1	Increment address register A by 1.
$05	MA+2	" " " " " 2.
$06	MA+8	" " " " " 8.
$07	MA+5	" " " " " 5.
$08	MA-1	Decrement " " " " 1.
$09	MA-2	" " " " " 2.
$0A	MA-8	" " " " " 8.
$0B	MA-5	" " " " " 5.
$0C	MB+1	Increment " " B " 1.
$0D	MB+2	" " " " " 2.
$0E	MB+8	" " " " " 8.
$0F	MB+5	" " " " " 5.
$10	MB-1	Decrement " " " " 1.
$11	MB-2	" " " " " 2.
$12	MB-8	" " " " " 8.
$13	MB-5	" " " " " 5.
$14	FA	Read into Fetch Register using A.
$18	FA+1	" " " " " ", increment by 1.
$19	FA+2	" " " " " ", " " 2.
$1A	FA+8	" " " " " ", " " 8.
$1C	FA-1	" " " " " ", decrement " 1.
$1D	FA-2	" " " " " ", " " 2.
$1E	FA-8	" " " " " ", " " 8.
$20	FB	Read into Fetch Register using B.
$24	FB+1	" " " " " ", increment by 1.
$25	FB+2	" " " " " ", " " 2.
$26	FB+8	" " " " " ", " " 8.
$28	FB-1	" " " " " ", decrement " 1.
$29	FB-2	" " " " " ", " " 2.
$2A	FB-8	" " " " " ", " " 8.
$2C	WAH	Write high byte using A.
$30	WAH+1	" " " " ", increment by 1.
$34	WAH-1	" " " " ", decrement " ".
$38	WBH	" " " " B.
$3C	WBH+1	" " " " ", increment by 1.
$40	WBH-1	" " " " ", decrement " ".
$44	WA	" (low) " " A.
$45	WA2	" 2 bytes " ".
$46	WA8	" 8 " " ".
$48	WA+1	" (low) byte " ", increment by 1.
$49	WA2+2	" 2 bytes " ", " " 2.
$4A	WA8+8	" 8 " " ", " " 8.
$4C	WA-1	" (low) byte " ", decrement " 1.
$4D	WA2-2	" 2 bytes " ", " " 2.
$4E	WA8-8	" 8 " " ", " " 8.
$50	WB	" (low) byte " B.
$51	WB2	" 2 bytes " ".
$52	WB8	" 8 " " ".
$54	WB+1	" (low) byte using B, increment by 1.
$55	WB2+2	" 2 bytes " ", " " 2.
$56	WB8+8	" 8 " " ", " " 8.
$58	WB-1	" (low) byte " ", decrement " 1.
$59	WB2-2	" 2 bytes " ", " " 2.
$5A	WB8-8	" 8 " " ", " " 8.
$5C	RA	Read into RWORD using A.
$60	RA+1	" " " " ", increment by 1.
$61	RA+2	" " " " ", " " 2.
$62	RA+8	" " " " ", " " 8.
$63	RA+5	" " " " ", " " 5.
$64	RA-1	" " " " ", decrement " 1.
$65	RA-2	" " " " ", " " 2.
$66	RA-8	" " " " ", " " 8.
$67	RA-5	" " " " ", " " 5.
$68	RB	" " " " B.
$6C	RB+1	" " " " ", increment by 1.
$6D	RB+2	" " " " ", " " 2.
$6E	RB+8	" " " " ", " " 8.
$6F	RB+5	" " " " ", " " 5.
$70	RB-1	" " " " ", decrement " 1.
$71	RB-2	" " " " ", " " 2.
$72	RB-8	" " " " ", " " 8.
$73	RB-5	" " " " ", " " 5.
$74	*FETCH	Start Fetcher.
$78	*WAFETCH	Write (low) byte using A, start Fetcher.
$79	*WA2F	" 2 bytes " ", " " .
$7C	*WBFETCH	" (low) byte " B, " " .
$7D	*WB2F	" 2 bytes " ", " " .
$80	MP+1	Increment P by 1.
$81	MP+2	" " " 2.
$82	MP+8	" " " 8.
$83	MP+5	" " " 5.
$84	MP-1	Decrement " " 1.
$85	MP-2	" " " 2.
$86	MP-8	" " " 8.
$87	MP-5	" " " 5.
$88	MR+1	Increment Real Address Register by 1.
$89	MR+2	" " " " 2.
$8A	MR+8	" " " " 8.
$8C	MR-1	Decrement " " " 1.
$8D	MR-2	" " " " 2.
$8E	MR-8	" " " " 8.
$90	FP	Read into Fetch Register using P.
$94	FP+1	" " " " ", increment by 1
$95	FP+2	" " " " ", " " 2.
$96	FP+8	" " " " ", " " 8.
$98	FP-1	" " " " ", decrement " 1.
$99	FP-2	" " " " ", " " 2.
$9A	FP-8	" " " " ", " " 8.
$9C	FR	" " " " R.
$A0	FR+1	" " " " ", increment by 1.
$A1	FR+2	" " " " ", " " 2.
$A2	FR+8	" " " " ", " " 8.
$A4	FR-1	" " " " ", decrement " 1.
$A5	FR-2	" " " " ", " " 2.
$A6	FR-8	" " " " ", " " 8.
$A8	WPH	Write high byte using P.
$AC	WPH+1	" " " " ", increment by 1.
$B0	WPH-1	" " " " ", decrement " ".
$B4	WRH	" " " " R.
$B8	WRH+1	" " " " ", increment by 1.
$BC	WRH-1	" " " " ", decrement " 1.
$C0	WP	" (low) " " P.
$C1	WP2	" 2 bytes " ".
$C2	WP8	" 8 " " ".
$C4	WP+1	" (low) byte " ", increment by 1.
$C5	WP2+2	" 2 bytes " ", " " 2.
$C6	WP8+8	" 8 " " ", " " 8.
$C8	WP-1	" (low) byte " ", decrement " 1.
$C9	WP2-2	" 2 bytes " ", " " 2.
$CA	WP8-8	" 8 " " ", " " 8.
$CC	WR	" (low) byte " R.
$CD	WR2	" 2 bytes " ".
$CE	WR8	" 8 " " ".
$D0	WR+1	" (low) byte " ", increment by 1.
$D1	WR2+2	" 2 bytes " ", " " 2.
$D2	WR8+8	" 8 " " ", " " 8.
$D4	WR-1	" (low) byte " ", decrement " 1.
$D5	WR2-2	" 2 bytes " ", " " 2.
$D6	WR8-8	" 8 " " ", " " 8.
$D8	RP	Read into RWORD using P.
$DC	RP+1	" " " " ", increment by 1.
$DD	RP+2	" " " " ", " " 2.
$DE	RP+8	" " " " ", " " 8.
$DF	RP+5	" " " " ", " " 5
$E0	RP-1	" " " " ", decrement " 1.
$E1	RP-2	" " " " ", " " 2.
$E2	RP-8	" " " " ", " " 8.
$E3	RP-5	" " " " ", " " 5.
$E4	RR	" " " " R.
$E8	RR+1	" " " " ", increment by 1.
$E9	RR+2	" " " " ", " " 2.
$EA	RR+8	" " " " ", " " 8.
$EC	RR-1	" " " " ", decrement " 1.
$ED	RR-2	" " " " ", " " 2.
$EE	RR-8	" " " " ", " " 8.
$FF	HOLD	Refresh holdoff (DIAGNOSTIC USE ONLY).

TABLE 3-16
Memory Control and
Memory Increment Coding

3.3 DISCRETE FUNCTION SECTION

The Discrete Function section of the microword controls a number of miscellaneous functions related to memory, I/O, and ALU status and bypass.

3.4 DISCRETE FUNCTION FIELDS

The fields contained in the Discrete Function section are:

ALU Status Save	Causes status conditions such as carry to be saved.
2901 Bypass	Reroutes data around ALU.
IOE Lines	Controls the I/O interface.
Memory Control and Memory Increment	Handles reading and writing to the memory interface.

3.4.1 ALU Status Save(ALU)

This single bit is set to save the current status flags from the ALU(C8, AZ8, etc.) and the BCD Adder(C10, AZ10).

3.4.2 2901 Bypass

When this bit is set, the D Bus is connected directly to the Y Bus, and the output of the 2901s is disabled. This allows data to be routed around the 2901s while the 2901s perform some other operation. The 2901s may still use data on the D Bus.

3.4.3 Memory Control and Memory Increment

These two fields are encoded to cover all the possible memory request situations that may occur. Table 3-16 shows the coding of the Memory Control and Memory Increment Fields.

3.4.4 IOE Lines

This 4-bit field controls the IOE lines to the backplane. Only one control can be specified per instruction. The default code is Read Status 0, which is supplied by the Assembler. Table 3-17 lists the coding of the IOE Field.

CODE	MNEMONIC	MEANING
0	RDD	Read Data
1	RS0	Read Status 0
2	WRD	Write Data
3	WRC	Write Control
4	RIO	Reset I/O
5	SRD	Set Read
6	SWR	Set Write
7	TERM	Set Terminate
8	RS1	Read Status 1
9	RS2	Read Status 2
A	CYL	Set Cylinder (disk only)
B	HEAD	Set Head (disk only)
C	DIFF	Set Difference (disk only)
D	DCTL	Disk Control
E	SECT	Set Sector (disk only)
F	SEL	Select Controller

TABLE 3-17
IOE Coding

3.5 NEXT ADDRESS SECTION

This field is used to specify the address of the next microword to be executed, to load literal constants, test the General Purpose Counters, and handle the stack.

3.6 NEXT ADDRESS FIELDS

The three Next Address fields in the microword operate unpipelined. They are: Next Address Mode, Next Address Select, and Next Address.

3.6.1 Next Address Mode

This 3-bit field indicates the mode that the Next Address Logic will assume for an operation. Some of the modes are further modified by the contents of the Next Address Select field. Table 3-18 lists the coding of the Next Address Mode Field.

CODE	MODE	MEANING
000	Jump Absolute	Transfer Next Address(N/A) field intact to N/A Register.
001	2-Way Branch	2-Way modified address passed to N/A Register.
010	4-Way Branch	4-Way modified address passed to N/A Register.
011	Special Branch	Modified address passed to N/A Register.
100	Call (Push & Jump)	Write current address to the Stack. Transfer Next Address Field intact to N/A Register.
101	Return	Address from stack added to offset specified in Next Address field. Result passed to N/A Register.
110	Pop	Transfer Next Address field intact to N/A Register. Discard last address written to stack.
111	Literal	Increment address in N/A Register. Load value in Next Address field and Next Address Select field into Literal Register.

TABLE 3-18
Next Address Mode
Coding

3.6.2 Next Address Select

This 4-bit field is used to modify the Next Address Mode. Depending on the mode present, the Next Address Select field will specify the conditions to modify the Next Address.

On a literal, the Next Address Select field will contain the high nibble of the high byte of a constant to be loaded into the literal register.

3.6.2.1 2-Way Branch Selects

In the 2-Way Branch mode, the least significant bit of the Next Address field is replaced by a condition flag. The displaced bit is used to determine the polarity of the bit which is added to the address.

The condition can be used directly or inverted. The Assembler tests the flags address depending upon the mnemonic used.

Table 3-19 lists the coding of the Next Address Select Field on 2-Way Branches showing the assembler mnemonic, the condition(s) tested, and the status of the condition flag used in determining the Next Address on 2-Way Branches. The mnemonic for the inverted condition is shown in parenthesis. The status of the condition flag will be inverted for the mnemonics shown in parentheses.

CODE	MNEMONIC	CONDITION TESTED	REPLACEMENT BIT VALUE
0	Z16 or Z (NZ)	ALU 16-Bit Zero	0 if nonzero 1 if zero
1	Z8 (NZ8)	ALU 8-Bit Zero	0 if nonzero 1 if zero
2	Z4 (NZ4)	ALU 4-Bit Zero	0 if nonzero 1 if zero
3	C16 or C (NC)	ALU 16-Bit Carry	0 if no carry 1 if carry
4	C8 (NC8)	ALU 8-Bit Carry	0 if no carry 1 if carry
5	C4 (NC4)	ALU 4-Bit Carry	0 if no carry 1 if carry
6	Y15 (NY15)	Y Bus Bit 2**15 (indirect)	0 if 2**15 = 0 1 if 2**15 = 1
7	Y0 or E0 (NY0)	Y Bus Bit 2**0 (even/odd)	0 if 2**0 = 0 1 if 2**0 = 1
8	Z4H (NZ4H)	4-bit Zero, High Nibble, Least Significant Byte ALU	0 if nonzero 1 if zero
9	SC8 (NSC8)	ALU Saved 8-Bit Carry	0 if no carry 1 if carry
A	INT (NINT)	Micro Interupt	0 if none 1 if interrupt
B	PAR (NPAR)	Memory Parity Error	0 if no error 1 if error
C	SRQ (NSRQ)	IO Service Request	0 if none 1 if Ser. Req.
D		Not defined	
E	GPNZ (GPZ)	Test General Purpose Counter	0 if nonzero 1 if zero
F	GPNZ4 (GPZ4)	Test Low Nibble General Purpose -	0 if nonzero 1 if zero

TABLE 3-19
Next Address
2-Way Branch Coding

3.6.2.2 4-Way Branch Select

In the 4-Way Branch mode, the least significant two bits of the Next Address are replaced by two external condition flags. A subset of this is called Type 2 2-way branching where one of the flags is selected and the other is replaced by a 1. The selection is determined by the replaced bits in the Next Address field. The conditions are shown most significant bit first.

Table 3-20 shows the coding of the Next Address Select Field for 4-Way Branches, showing the code, the assembler mnemonic, the condition tested, and the value of the replacement bits for each flag status.

CODE	MNEMONIC	CONDITION TESTED	REPLACEMENT BIT VALUES
0	ZC ZX	ALU 16-Bit Zero; ALU 16-Bit Carry ALU 16-Bit Zero; 1	+0 Nonzero;no carry +1 Nonzero;carry +2 Zero;no carry +3 Zero;carry
1	Z8C8 Z8XX	ALU 8-Bit Zero; ALU 8-Bit Carry ALU 8-Bit Zero; 1	Same as ZC
2	Z8SC8	ALU 8-Bit Zero; 8-Bit Saved Carry	Same as ZC
3	IB	Indirect Bits from OP2 and OP3	+0 No indirects +1 OP3 indirect +2 OP2 indirect +3 Both indirect
4	HAZL HAZLA HAZLB	Base/Bank Hazard LARA; Base/Bank Hazard LARB Base Hazard LARA; 1 1; Base Hazard LARB	+0 No Hazards +1 Hazard on LARB +2 Hazard on LARA +3 Hazard on both
5		Not defined	
6	STACK	Stack overflow; Stack underflow	+0 Neither +1 Underflow +2 Overflow +3 Both
7	COLD	0; Coldstart	+0 Warm start +1 Cold start
8	C8CZ8	ALU 8-Bit Carry; ALU Combination 0 (Least Significant Bit = 0 and Least Significant Byte = 0)	+0 No carry;nonzero +1 No carry;zero +2 Carry;nonzero +3 Carry;zero
9	SCAZ10	Saved BCD Carry; Accumulated decimal 0	Same as C8CZ8
A	SOHAZP	Shift Out; Base/Bank Hazard LARP	+0 SO=0;no hazard +1 SO=0;hazard +2 SO=1;no hazard +3 SO=1;hazard
B	NRMOVR	Y Bus normalized ALU Overflow	+0 NRM=0;no overflow +1 NRM=0;overflow +2 NRM=1;no overflow +3 NRM=1;overflow
C		Not defined	
D		Not defined	
E	IGPZ	Micro Interrupt; General Purpose Counter Zero	+0 No int;zero +1 No int;nonzero +2 Int;zero +3 Int;nonzero
F	GPNZB4	4-Bit Zero 16-bit General Purpose Counter(GPC); 4-Bit General Purpose Counter (GPC2) Zero	+0 Both nonzero +1 GPC>0;GPC2 zero +2 GPC zero;GPC2>0 +3 Both zero

TABLE 3-20
Next Address
4-Way Branch Coding

3.6.2.3 Special Branch Selects

On a 16-Way Branch, the low nibble of the Next Address is replaced, while on a 256-way Branch, the low byte of the Next Address is replaced. The contents of the Branch Register, the Y Bus or a combination of flags determine the replacement value. Table 3-21 is a list of the coding of the Next Address Select Field for Special Branches showing the code, the assembler mnemonics and the condition tested.

CODE	MNEMONIC	MEANING
0	BRRL	16-way branch on lower 4 bits of Branch Register
1	BRRU	16-way branch on upper 4 bits of Branch Register
2	BRALL	256-way branch on all 8 bits of Branch Register
3	YBRRL	16-way branch on low nibble of the least significant byte on Y Bus
4	YBRRU	16-way branch on high nibble of the least significant byte on Y Bus
5	GLINT	16-way branch on: Single step mode; Macro Interrupt; STOP button pressed; Any Micro Interrupt
6	UINT	16-way branch on: IPL Warning; Power Fail Warning; UART Interrupt;
7	YBRALL	256-way branch on low byte on Y Bus
A-F		Not defined

TABLE 3-21
Next Address
Special Branch Coding

3.6.3 Next Address

The 12-bit Next Address field may contain the specific address of the next microinstruction, an offset value for to be added to an address taken from the stack on a return from a subroutine, or lowest 12 bits of a constant to be loaded into the Literal Register.

3.7 MICROCODE DECODE

Q64 MICROWORD WORKSHEET

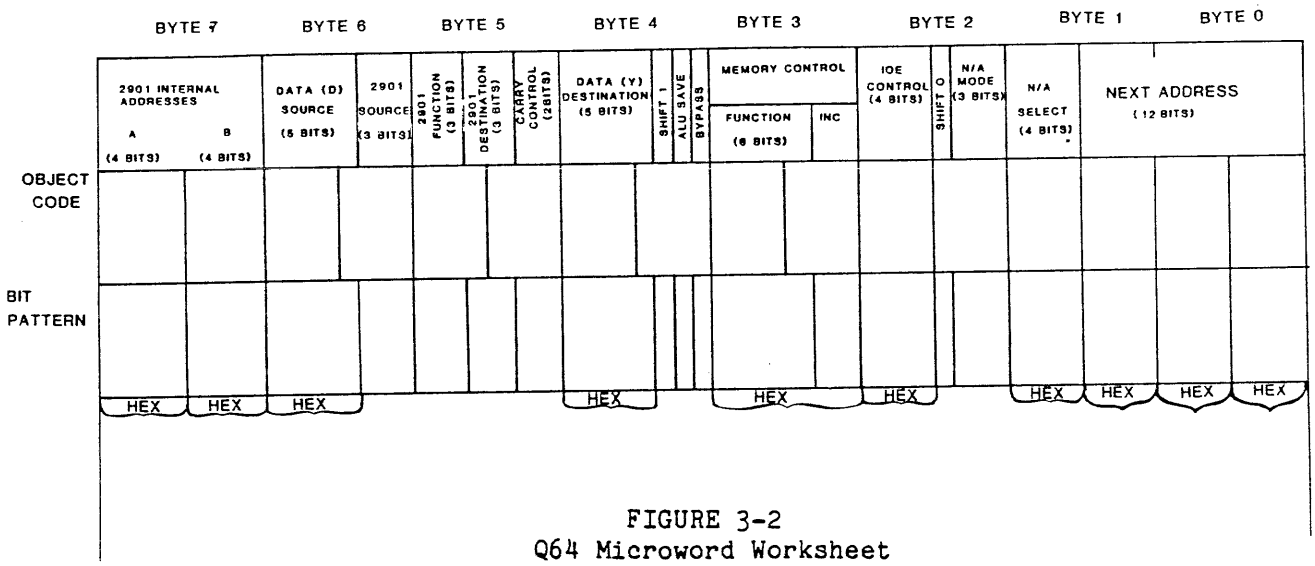


FIGURE 3-2
Q64 Microword Worksheet

Figure 3-2 was developed to provide a method for readily decoding microcode in troubleshooting the Q64 CPU. In the example which follows, part of a Board Bug subroutine is decoded. Board Bug is a diagnostic microprogram which is contained in a special set of PROMs. The Board Bug PROMs are substituted for the Microstore PROMs on the ALU Board in troubleshooting the Q64 CPU.

PCTR	OBJECT CODE	LABEL	NEXT ADDRESS	EXPRESSION
0449	0002840000100447		BASE2A	
0447	33040D000011E441	BASE2A	GPZ,BASEADD3	R3+1->R3
0441	330404E8001003F3			R3->
03F3	33E5C400001003F2			R3->BPTR
03F2	3304040000111422		Z8,	R3->
			RES 2	
0443	0002840000117444		Y0,	
			RES 2	
0445	33040458001003F1			R3->SWAP
03F1	440F0C00001003F0			SWAP->R4
03F0	44EDC400001003EF			R4:BASE->
03EF	0002840000110446		Z16,	
			RES 2	
0446	0015643800100CE0		ERROR	R0 . LCL -> LCL
0CE0	00056C0000171000	ERROR	=$1000	LIT . R0 -> R0

TABLE 3-22
Sample Microcode
(Board Bug)

Table 3-22 is a listing of the Board Bug code which is decoded in Figures 3-3 through 3-14. Table 3-23 is a reference guide to the specific fields of the microcode, showing the sections in this manual which contain the coding for the specific field of the microword.

FIELD	REFERENCE SECTION
2901 INTERNAL ADDRESS	
A	3.2.1
B	3.2.1
D SOURCE	3.2.2
2901 SOURCE	3.2.3
2901 FUNCTION	3.2.4
2901 DESTINATION	3.2.5
CARRY CONTROL	3.2.6
Y DESTINATION	3.2.7
SHIFT CONTROL	3.2.8
ALU SAVE	3.4.1
BYPASS	3.4.2
MEMORY CONTROL AND INCREMENT	3.4.3
IOE CONTROL	3.4.4
NEXT ADDRESS MODE	3.6.1
NEXT ADDRESS SELECT	3.6.2
NEXT ADDRESS	3.6.3

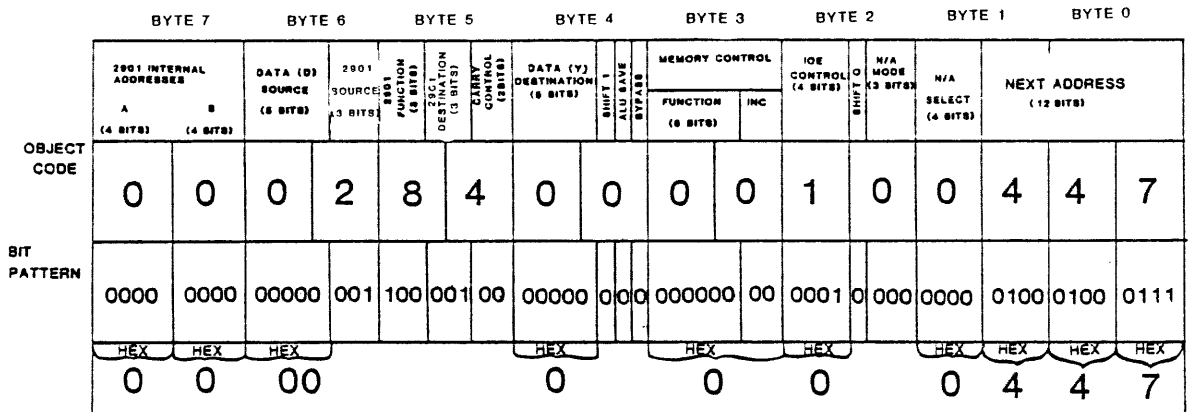
TABLE 3-23
Decode Reference
Guide

Figures 3-3 through 3-14 are completed composites of Figure 3-2 and Table 3-23. A line of microcode is inserted in the OBJECT CODE boxes. The BIT PATTERN is shown, and fields which are expressed in hexidecimal notation are then converted. The definition of each field is then given.

The code in the example is part of a subroutine which tests the loading and readback of the Base Register Files. Note that an error is detected in a 2-Way Branch and that the Next Address Mode Field of the last instruction is a Return from the subroutine. Since the Next Address Field contains $000, a $000 offset is added to the address previously written to the Stack. The starting address of the subroutine in which the error was detected becomes the Next Address. The Q64 CPU will loop on the subroutine until halted by the operator.

In the example used, there is no memory activity; however it is possible to gain some understanding of 2901B logic, Next Address Logic, and the microcode by studying the example. By following the steps taken in this example, and using the information in the sections cited in Table 3-23, any microword can be decoded.

Once the microcode has been decoded, it is possible to locate the hardware circuits involved in executing the code in the schematics. Troubleshooting the Q64 CPU is most often performed using a logic analyzer and a storage scope.



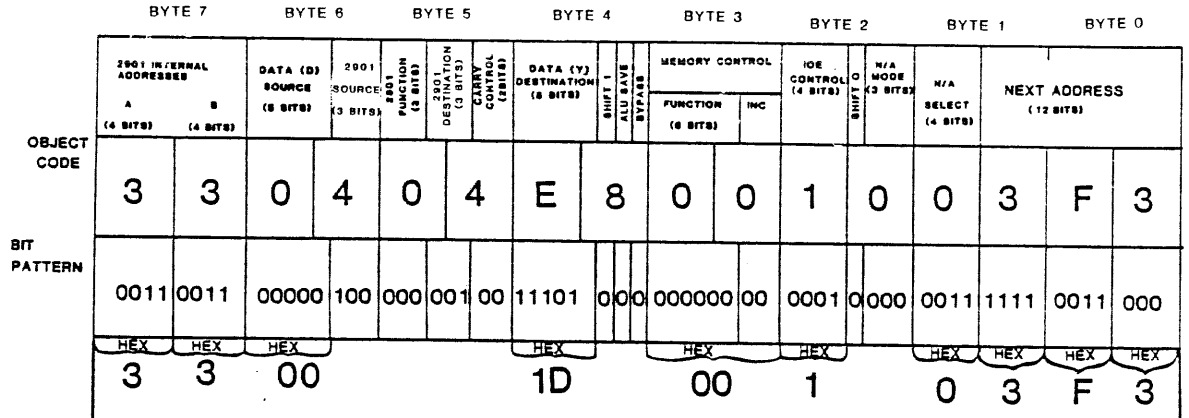
FIELD	INTERPRETATION
2901 ADDRESS	
A	R0
B	R0
D SOURCE	Literal Register(default)
2901 SOURCE	Use registers specified by A & B
2901 FUNCTION	Logical AND R & S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 unto carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or zero) mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0 (default)
NEXT ADDRESS MODE	Jump absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$447

FIGURE 3-3

	BYTE 7		BYTE 6		BYTE 5			BYTE 4				BYTE 3		BYTE 2		BYTE 1	BYTE 0		
	2901 INTERNAL ADDRESSES A (4 BITS)	2901 INTERNAL ADDRESSES B (4 BITS)	DATA (D) SOURCE (5 BITS)	2901 SOURCE (3 BITS)	2901 FUNCTION (3 BITS)	2901 DESTINATION (3 BITS)	CARRY CONTROL (2 BITS)	DATA (Y) DESTINATION (5 BITS)	SHIFT 1	ALU SAVE	BYPASS	MEMORY CONTROL FUNCTION (6 BITS)	MEMORY CONTROL INC	IOE CONTROL (4 BITS)	SHIFT	N/A MODE (3 BITS)	N/A SELECT (4 BITS)	NEXT ADDRESS (12 BITS)	
OBJECT CODE	3	3	0	4	0	D	0	0	0	0		1	1	E	4	4	1		
BIT PATTERN	0011	0011	00000	100	000	011	01	0000	0	0	0	000000	00	0001	0	001	1110	0100	0100	0001
	HEX	HEX	HEX					HEX				HEX		HEX			HEX	HEX	HEX	HEX
	3	3	00					0				00		1			E	4	4	1

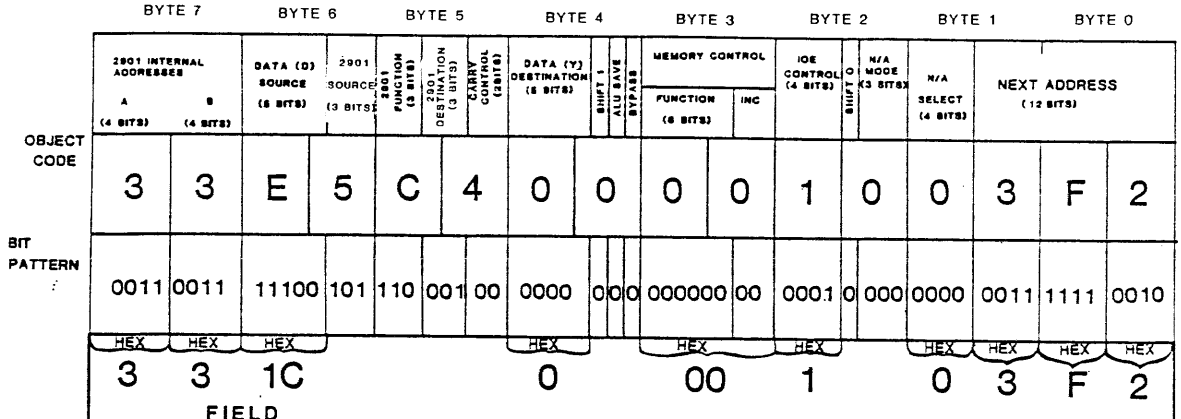
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R3
B	R3
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and Register specified by A
2901 FUNCTION	Add R to S
2901 DESTINATION	Result of ALU placed in B Source Register and on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	None
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	2-Way Branch
NEXT ADDRESS SELECT	General Purpose Counter Non-zero (GPNZ) - 0 if nonzero 1 if zero
NEXT ADDRESS	$441

FIGURE 3-4



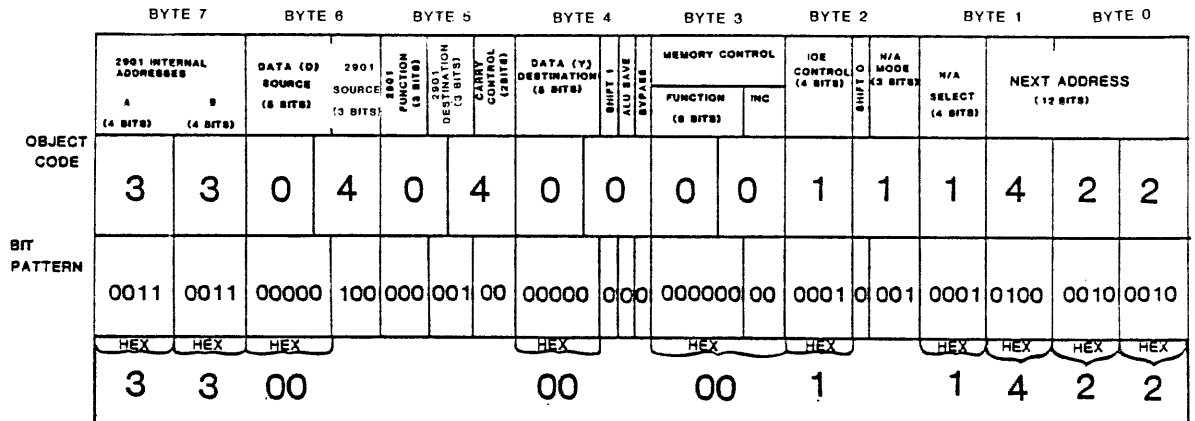
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R3
B	R3
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and register specified by A
2901 FUNCTION	Add R to S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	Base Pointer
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump Absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$3F3

FIGURE 3-5



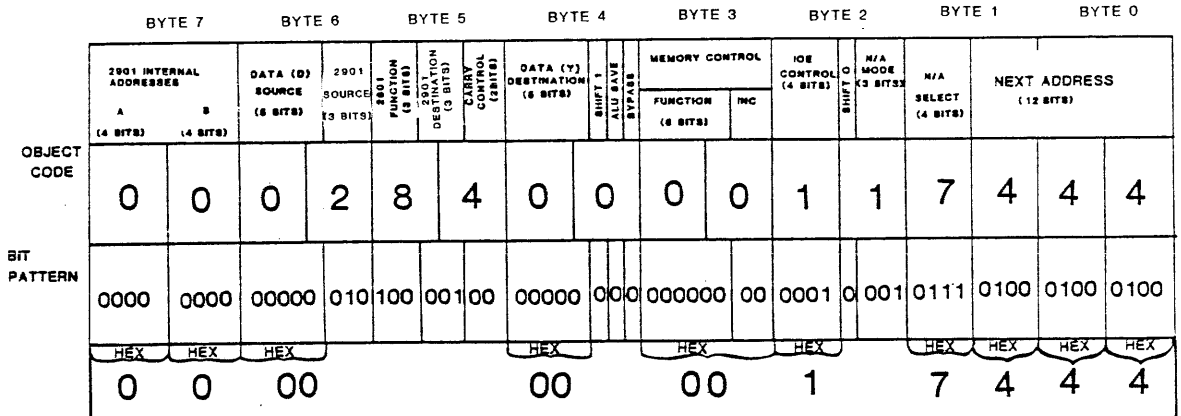
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R3
B	R3
D SOURCE	Lower 8 bits Base Register(BASEL)
2901 SOURCE	Use external data and register specified by A
2901 FUNCTION	Compliment R, Logical AND with S
2901 DESTINATION	Result placed on Y Bus. Result shifted one and placed in B source register. Q is shifted left.(Double shift left).
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	None
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump Absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$3F2

FIGURE 3-6



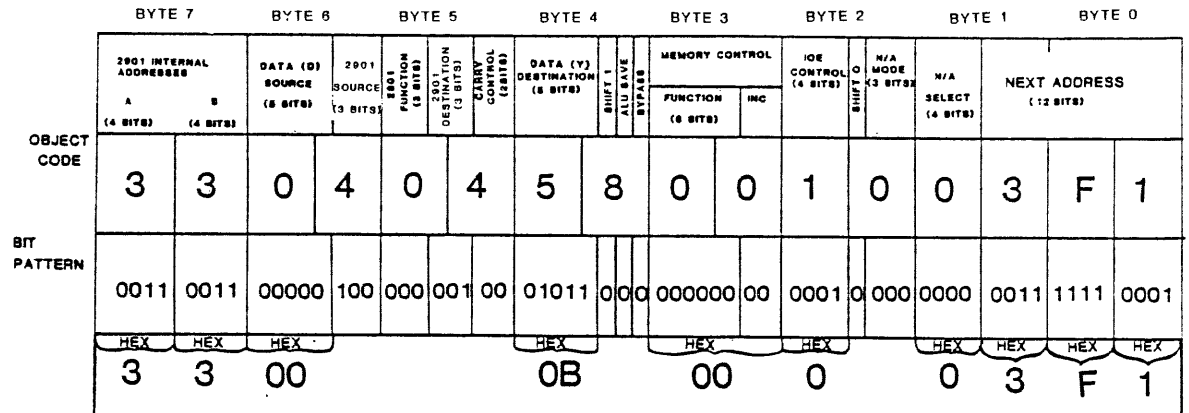
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R3
B	R3
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and register specified by A
2901 FUNCTION	Add R to S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	2-Way Branch
NEXT ADDRESS SELECT	ALU 8 bit zero(Z8) - 0 if nonzero 1 if zero
NEXT ADDRESS	$422

FIGURE 3-7



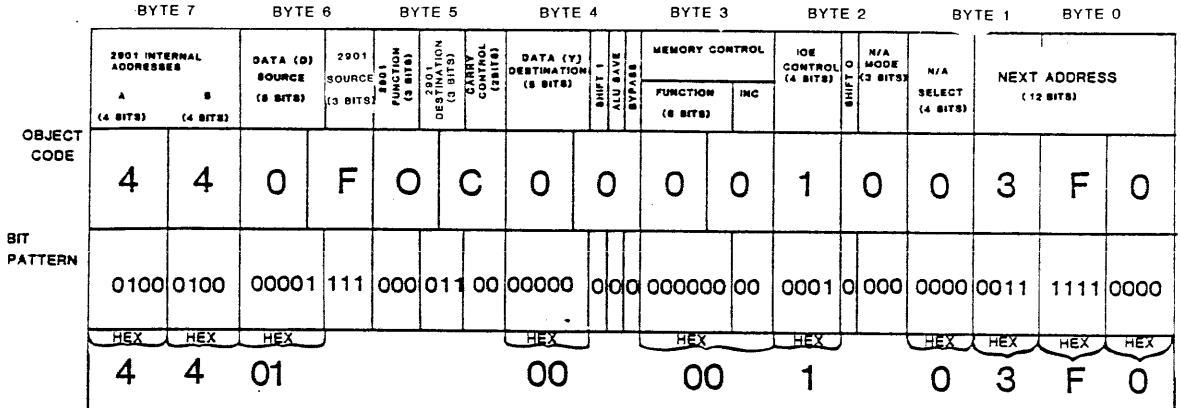
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R0
B	R0
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and Q Register
2901 FUNCTION	Logical AND R & S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	2-Way Branch
NEXT ADDRESS SELECT	Y Bus 2**0(odd/even) - 0 if 0 1 if 1
NEXT ADDRESS	$444

FIGURE 3-8



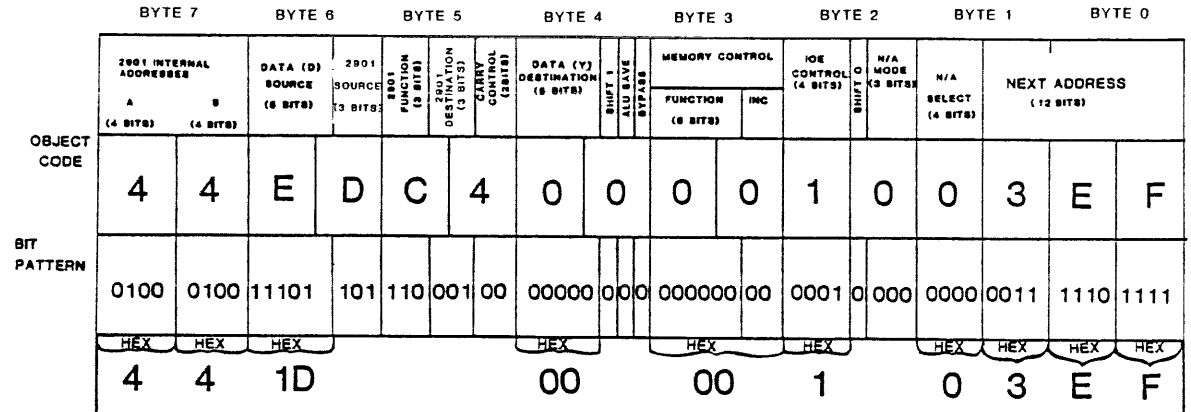
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R3
B	R3
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and register specified by A
2901 FUNCTION	Add R & S
2901 DESTINATION	Result of ALU to Y outputs
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	Byte Swap Register
SHIFT CONTROL	Logical(or zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump Absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$3F1

FIGURE 3-9



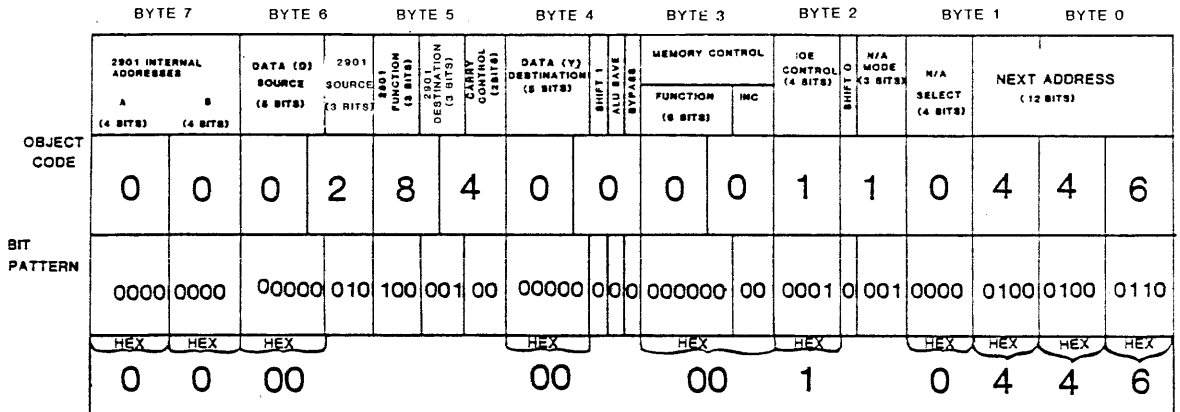
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R4
B	R4
D SOURCE	Byte Swap Register
2901 SOURCE	Use external data and register specified by A
2901 FUNCTION	Add R & S
2901 DESTINATION	Result of ALU placed in B source register and on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump Absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$3F0

FIGURE 3-10



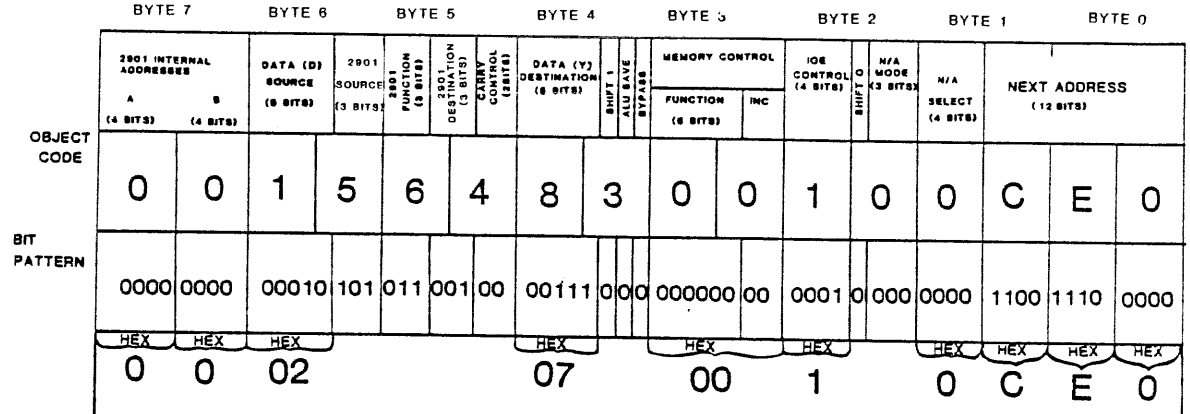
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R4
B	R4
D SOURCE	Upper 16 bits Base Register(BASE)
2901 SOURCE	Use external data and register specified by A
2901 FUNCTION	Exclusive OR R & S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump Absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$3EF

FIGURE 3-11



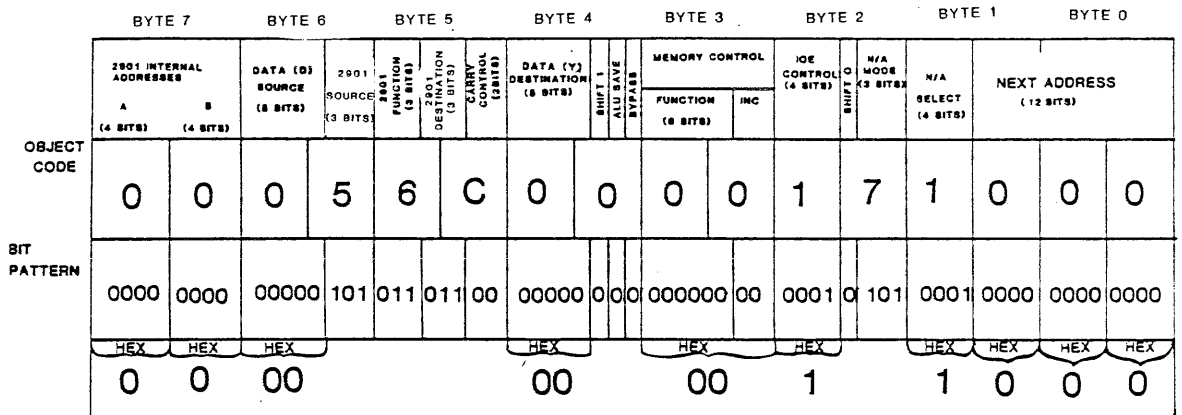
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R0
B	R0
D SOURCE	Literal Register
2901 SOURCE	Use Constant 0 and Q register
2901 FUNCTION	Logical AND R & S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	2-Way Branch
NEXT ADDRESS SELECT	ALU 16 Bit Zero(Z16) - 0 if nonzero 1 if zero
NEXT ADDRESS	$446

FIGURE 3-12



FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R0
B	R0
D SOURCE	Local Control Register
2901 SOURCE	Use external data and register specified by A
2901 FUNCTION	Logical OR R & S
2901 DESTINATION	Result of ALU placed on Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	Local Control Register
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read Status 0
NEXT ADDRESS MODE	Jump absolute
NEXT ADDRESS SELECT	None
NEXT ADDRESS	$CE0

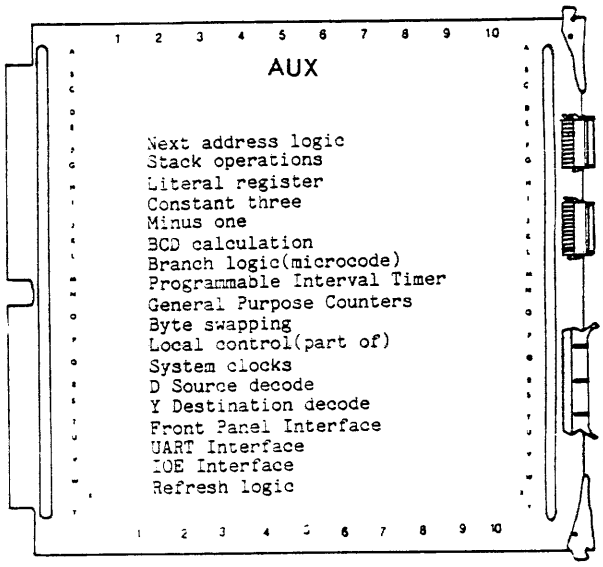
FIGURE 3-13

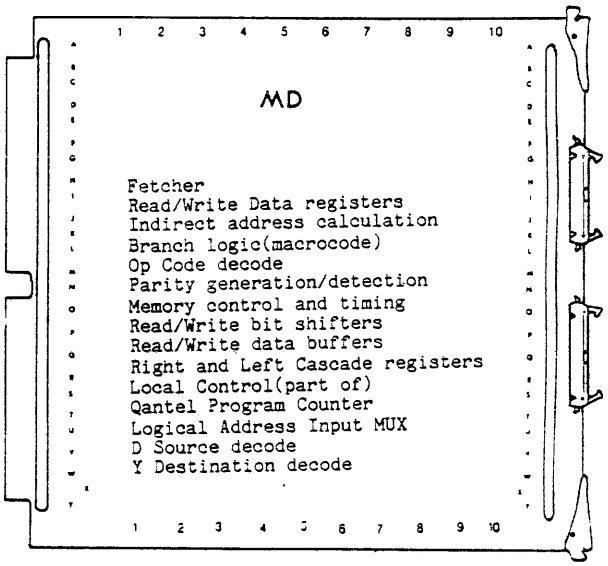


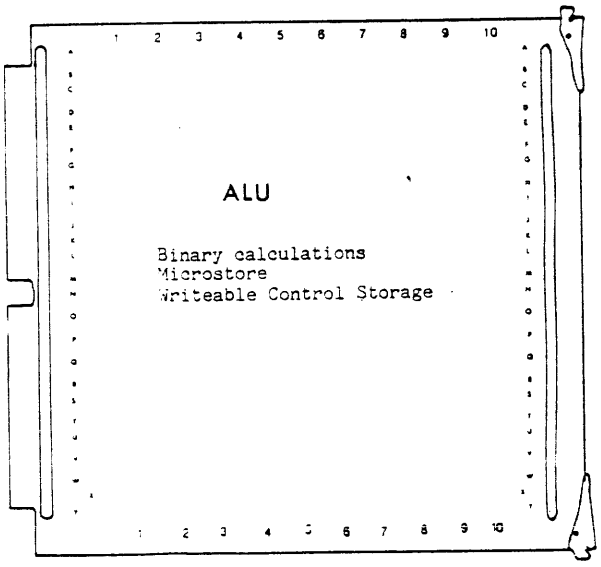
FIELD	INTERPRETATION
2901 INTERNAL ADDRESS	
A	R0
B	R0
D SOURCE	Literal Register
2901 SOURCE	Use external data and register specified by A
2901 FUNCTION	Logical OR R & S
2901 DESTINATION	Result of ALU placed in B Source Register and Y output
CARRY CONTROL	Forced 0 into carry input
Y DESTINATION	No target(NOP)
SHIFT CONTROL	Logical(or Zero) Mode
ALU SAVE	No
BYPASS	No
MEMORY CONTROL AND INCREMENT	No request for memory activity
IOE CONTROL	Read status 0
NEXT ADDRESS MODE	Return
NEXT ADDRESS SELECT	Contains Literal
NEXT ADDRESS	Contains Literal - Also 0 offset to add to address off stack.

FIGURE 3-14

4.0 LOGICS OVERVIEW







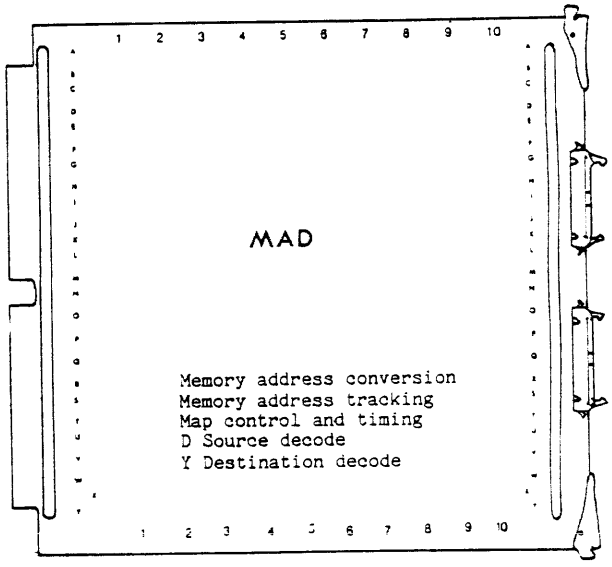


FIGURE 4-1
CPU Boards

Each of the four Q64 CPU Boards contains discrete circuits which perform specific functions somewhat in isolation from the other logics. The decoding of microcode, binary arithmetic and logic are performed on the Arithmetic Logic Unit(ALU) Board: Next Address calculation and miscellaneous IO and descrete functions are performed on the Auxilliary(AUX) Board: memory address calculation and tracking is performed on the Memory Address(MAD) Board; and memory data interchange and some macrocode decode is performed on the Memory Data(MDAT) Board.

Figure 4-1 is a representation of the four PCBs in the Q64 CPU, showing all of the discrete functions performed on each board. Data flow between all four of the boards is accomplished primarily through the D(source) Bus and the Y(destination) Bus. The two busses are common to all four boards through the backplane connectors. The MAD and MDAT boards are also connected to each other via a ribbon connector providing direct throughput for memory address calculations and synchronous parity generation and detection.

Following a discussion on busses, each CPU board is discussed in detail. One chapter is devoted to each board. The Diagnostic Panel logics are included and discussed in the final chapter.

D SELECT	REGISTER[mnemonic]	SCHEMATIC LOCATION BOARD	SHEET
$00	Literal[LIT]	AUX	5
$01	Byte Swapping[SWAP]	AUX	13
$02	Local Control[LCL]	AUX	13
$03	UART Data[UDATA]	AUX	12
$04	IOU Receive Data[IODATA]	AUX	16
$05	Constant Three[THREE]	AUX	13
$06	BCD Adder Result[BCDRES]	AUX	14
$07	Programmable Interval Timer[TIMER]	AUX	11
$08	WCS Readback [WCSD]	ALU	4
$09	Service Panel[FRPNL]	ALU	22
$0B	Flag[FLAGS]	ALU	11
$0C	Readword 1[RWORD]	MD	8
$0D	Readword 2[RWORD2]	MD	8
$0E	Readword 3[RWORD3]	MD	8
$0F	Readword 4 - Operand 3[RWORD4]	MD	5
$10	Operand 1[OP1]	MD	4
$11	Operand 2[OP2]	MD	4,5
$12	Op Code 1[CODE1]	MD	4
$13	Op Code 2[CODE2]	MD	5
$14	Left Cascade[LCAS]	MD	8
$15	Right Cascade[RCAS]	MD	8
$16	Read Byte[RBYTE]	MD	8
$17	Parity Error Address[PERR]	MD	9
$18	Logical Address A[LARA]	MAD	4
$19	Logical Address B[LARB]	MAD	5
$1A	Logical Address P[LARP]	MAD	6
$1B	Old Logical Address P[LARPO]	MAD	7
$1C	Lower 8 Bits Base[BASEL]	MAD	7
$1D	Upper 16 Bits Base[BASE]	MAD	7
$1E	Move Length[MOVLEN]	MD	10
$1F	Minus One[FFFF]	ALU	12

TABLE 4-1
D Source Index

4.1 SYSTEM BUSSES

The only system-wide busses are the 16-bit D and Y busses. Other discrete circuits utilize busses ranging in widths up to 64-bits. Data exchanges between 8, 16 and 64-bit busses occurs through devices which latch the data through multiple machine cycles. Loading from or outputting to the smaller busses requires sequential operations defined by the microcode. The major busses on each board are discussed below.

4.1.1 ALU Board Busses

The primary system busses manipulated by the ALU are the D Bus and the Y Bus. The D Bus connects the 2901Bs to all Data Sources, and the Y Bus connects the 2901Bs to all Data Destinations. Tables 4-1 and 4-2 list the sources and destinations available to the CPU and show the logics location of the specific devices. D Sources are discussed in Sections 3.2.2, and Y Destinations are discussed in Section 3.2.7.

Y SELECT	REGISTER[mnemonic]	SCHEMATIC LOCATION BOARD	SHEET
$00	No Operation[NOP]		
$01	Branch[BRR]	AUX	4
$02	WCS Write Data Word 1[WCSD1]	ALU	5
$03	WCS Write Data Word 2[WCSD2]	ALU	5
$04	WCS Write Data Word 3[WCSD3]	ALU	5
$05	WCS Write Data Word 4[WCSD4]	ALU	5
$06	WCS Read/Write Address Pointer[WCSA]	ALU	5
$07	Local Control[LCL]	AUX	13
$08	UART Data Output[UDATA]	AUX	12
$09	UART Control[UCTL]	AUX	12
$0A	Junk[JUNK]	AUX	14
$0B	Byte Swapping[SWAP]	AUX	13
$0C	General Purpose Counter - 16 bits[GPC]	AUX	14
$0D	General Purpose Counter - 4 bits[GPC2]	AUX	14
$0E	BCD Adder Operand 1[BCD1]	AUX	14
$0F	BCD Adder Operand 2[BCD2]	AUX	14
$10	WCS Page[PAGE]	ALU	7
$11	Programmable Interval Timer[TIMER]	AUX	11
$12	IOU Interface Output Data[IODATA]	AUX	16
$13	Flag[FLAGS]	ALU	11
$14	Write Word 1[WWORD]	MD	4
$15	Write Word 2[WWORD2]	MD	4
$16	Write Word 3[WWORD3]	MD	5
$17	Write Word 4[WWORD4]	MD	5
$18	Logical Address Register A[LARA]	MAD	4
$19	Logical Address Register B[LARB]	MAD	5
$1A	Logical Address Register P[LARP]	MAD	6
$1B	Real Address Register[RAR]	MAD	14
$1C	Temporary Register[TEMP]	MAD	10
$1D	Base Read/Write Pointer[BPTR]	MAD	10
$1E	Base Register[BASE]	MAD	10
$1F	Front Panel[FPNL]	AUX	15

TABLE 4-2
Y Destination Index

The 64-bit bidirectional Microstore Data(MSDAT) Bus connects the Microstore ROMs with the Pipeline Register and other logic. The 64-bit bidirectional WCS Data (WCS DATA) Bus is connected to the MSDAT Bus through a 64-bit buffer.

Both the Microstore and WCS Address busses are 16-bits wide, although only 12 lines are presently used for addressing the Microstore and only 11 lines are used for addressing WCS. One of the lines is a WCS Flag which is used to select WCS or the Microstore as the microprogram source. The additional lines within the bus will be used when the Microstore or WCS is expanded.

Figure 4-2 is a block diagram of the ALU Board and shows the general data flow, including the breakdown of the 64-bit microword from the MSDAT Bus to the logic affected by specific fields of the Microword. Note that when the ROM Simulator is connected to the ALU Board, the Microstore and WCS are disabled and the ROM Simulator utilizes the MSDAT and other system busses in the same manner in which the Microstore does.

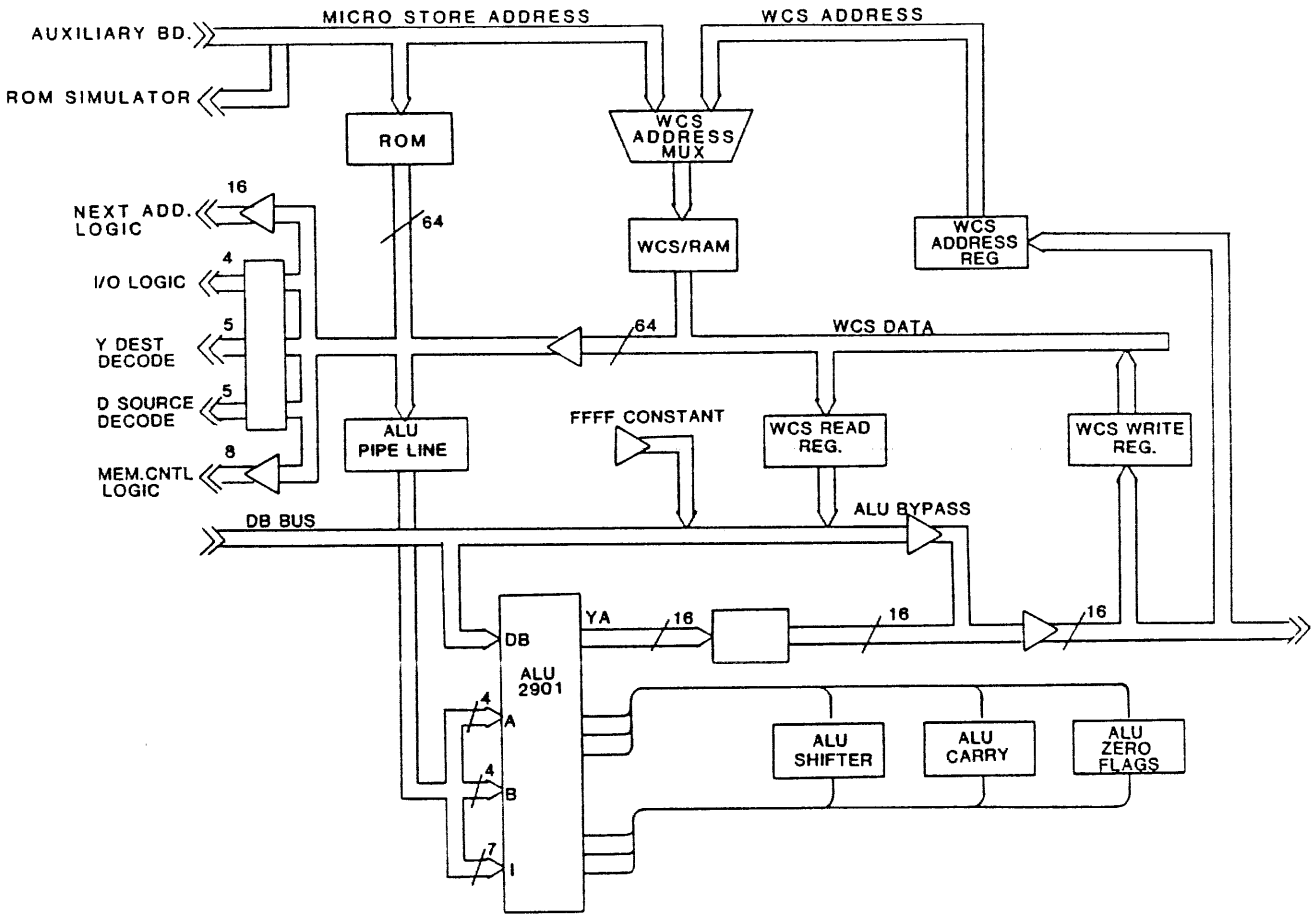


FIGURE 4-2
ALU-R Board
Block Diagram

4.1.2 AUX Board Busses

The AUX Board provides the primary data path between external devices and the CPU. Hardware is in place for a 16-bit data path using an expanded controller; however, only 8-bit controllers are presently used with the Q64. The 8-bit IO Data(DA) Bus requires sequential loading of the IO Write Data buffer from the low byte on the Y Bus and sequential reads from the IO Read Data buffer to the low byte of the D Bus.

The diagnostic serial ports on the AUX Board provide an 8-bit wide data path between the CPU and remote or local diagnostic devices(i.e. the Q64 Test Panel). The UART Transmitter Holding Register is loaded from the low byte on the Y Bus, and the UART Received Data Register outputs to the low byte on the D Bus(the high byte on the D Bus contains miscellaneous status signals from the UART).

Other primary busses on the AUX Board are related to the Next Address Logic and include the 16-bit Microstore Address (MSA), Next Address(NA) and Stack Address (ST) busses. Figure 4-3 is a block diagram of the Next Address Logic on the AUX Board. Other logics on the AUX Board are essentially discrete circuits that are functionally independent of each other.

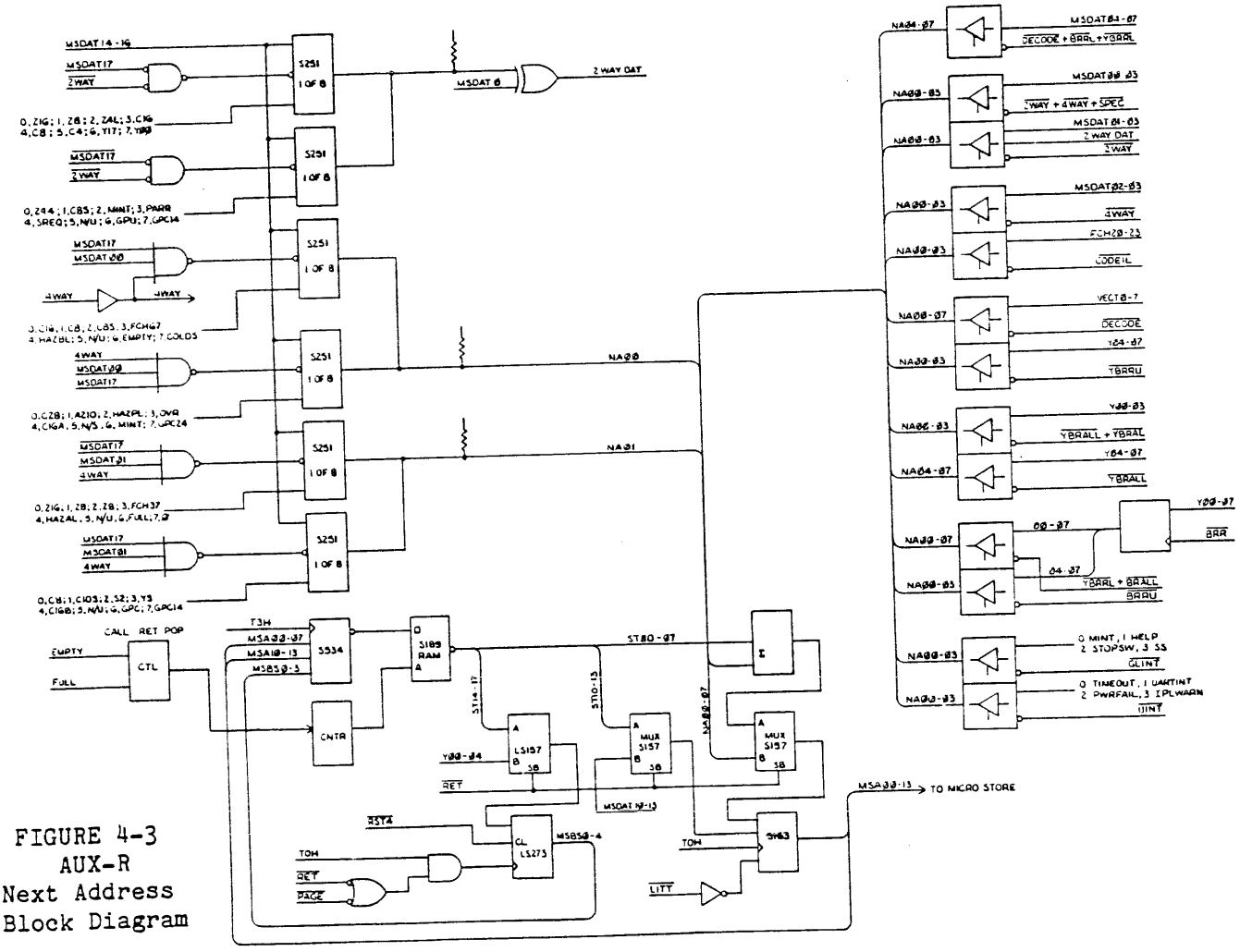


FIGURE 4-3
AUX-R
Next Address
Block Diagram

4.1.3 MAD Board Busses

There are two 16-bit Logical Address Input Busses which connect the Logical Address Input MUX on the MDAT Board with the Logical Address Registers on the MAD Board via the Ribbon Connector between the two boards. One input bus(YLARA) is used only for LARA, while the other(YLARBP) is shared by LARB and LARP. The 16-bit Logical Address may be output to the D Bus, or routed via a 16-bit Logical Address Register(LAR) Bus to the Base Adders(11 bits) and the Base Pointer(4 bits). The most significant bit of the logical address is the indirect address bit and is ignored in the calculation of the memory address after indirection has been resolved.

A 24-bit Base Register(BASE) Bus connects the Base Register with the Base Adders. The 24-bit Physical Address Register(PAR) Bus connects the Physical Address Registers with the Physical Address MUX which outputs to the Row and Column Address MUX and the Address A+1 Incrementer via the 24-bit "Real" Memory Address (RMA) Bus.

Row and column addresses are output through the backplane to main memory via a 9 bit Address(A) Bus. (The 9th address line is presently not used, but it is in place for implementation when 256K RAMs are installed on the MEM64B). The incremented address is output through the backplane via a separate 9-bit Address+1 (A+1) Bus. The memory refresh address(row only) is also output through the backplane via the A Bus. Board and module select data from the "Real" Memory Address are output through the backplane over separate lines, as are the Byte and Word Offsets which are generated and output to memory from the MDAT Board.

Figure 4-4 is a block diagram of the Memory Address Board, showing the general data flow on the board.

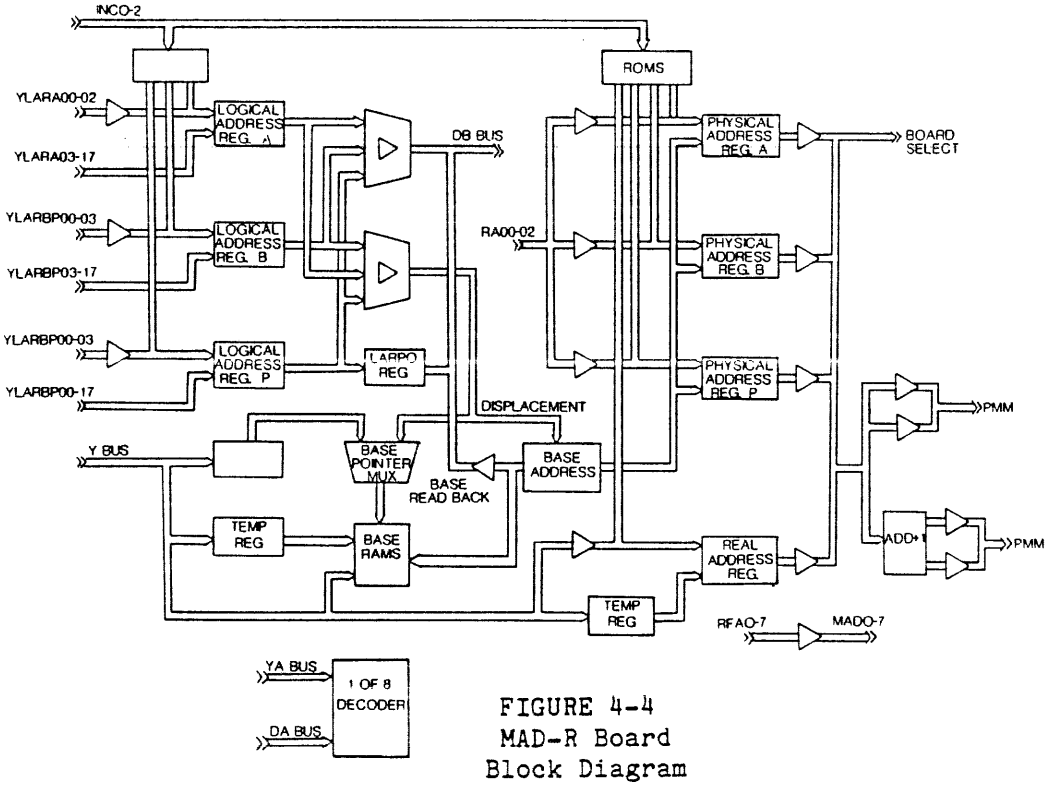


FIGURE 4-4
MAD-R Board
Block Diagram

4.1.4 Memory Data Board Busses

Data exchange between the CPU and main memory occurs over a 64-bit bidirectional Data(DATA) Bus. A 24-bit Address Bus(A) and an 8-bit bidirectional Parity Bus (DATxP) also connect the MDAT Board to main memory through the backplane.

Data to and from main memory is buffered between the Data Bus and a 64-bit bidirectional Memory (MDAT) Bus which feeds the Read Bit Shifter and is fed by the Write Bit Shifter. The Fetch Register is the Read/Write Data Register for all memory operations of more than two(2) bytes. Separate registers are provided for single byte and single word reads(RBYTE and RWORD1 respectively). The Fetch Register is connected to the Read Bit Shifter by the 64-bit Memory Read Data (MRD) Bus and to the Write Bit Shifter and Fetcher decode logic by the 64-bit Fetcher(FCH) Bus.

The MRD Bus is also a path for data to be written from the ALU or other data sources to memory. The MRD Bus is connected to the Y Bus through a 64-bit Write Data Buffer. Data from the Write Data Buffer propogates to the Fetch Register on writes to memory. The Fetch Register acts as a holding register on writes to memory, loading two bytes at a time from the Y Bus(via the Write Data Buffers), and outputting up to eight bytes at a time to the Write Bit Shifter. Several machine cycles may be required to load the Fetch Register prior to executing the memory write.

Figure 4-5 is a block diagram of the Memory Data Board, showing the general data flow on the board.

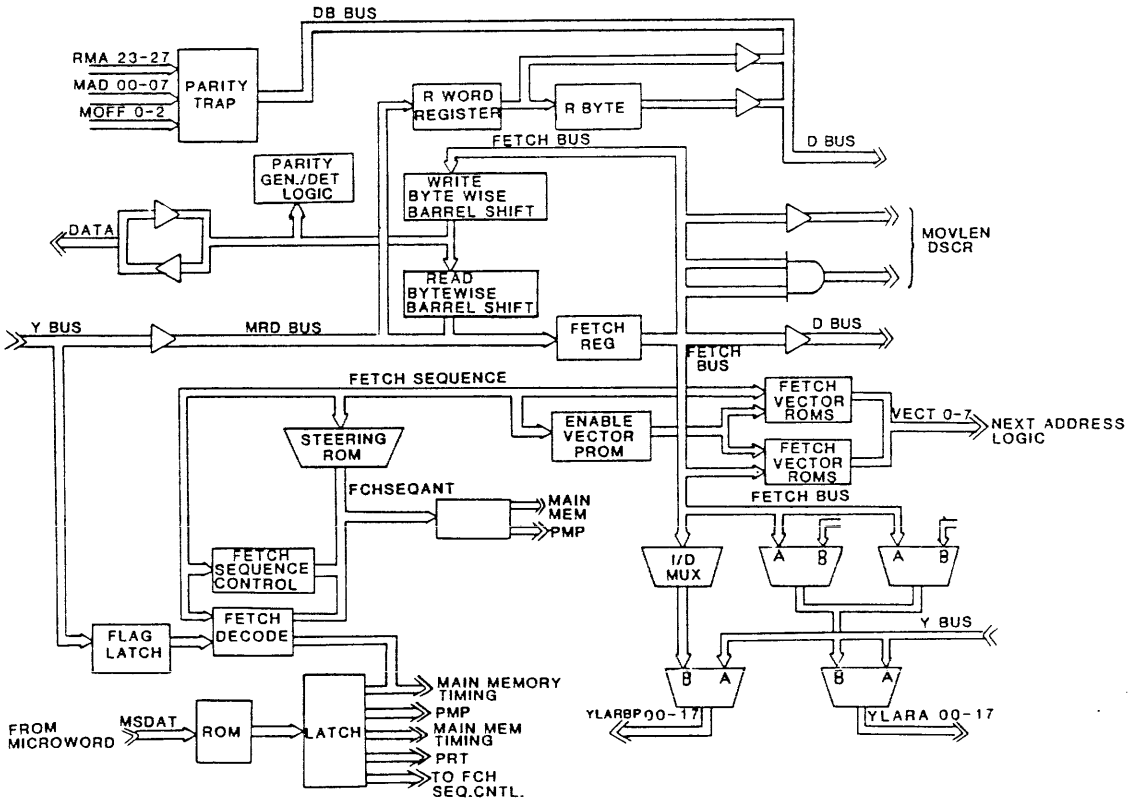


FIGURE 4-5
MDAT-R Board
Block Diagram

4.2 Signal Mnemonics and logic symbols

Appendix A is a listing of the signal mnemonics used on the Q64 Logics. Industry standard symbols are used throughout the logics. Low-active signals are identified on the logics by a solid line over the signal mnemonic.

4.3 Method of Logics Review

The general data flow on the logics is from left to right and top to bottom on each sheet. The discussion of each sheet follows the general data flow, with few exceptions. Where the logic of the circuits discussed does not follow the graphic representation, the flow of the logic is followed. For example, the discussion of Logical Address Register A(MAD Board Sheet 4) evolves around the enable signals and timing shown in the top center(D6) and the center of the right side(C2-C3) of the sheet.

Specific ICs are identified by their board location which is shown on the logics by a circled number on the IC. Signals are referred to by their mnemonics if one is designated on the logics. Otherwise the signal is referenced by the board identifier and the specific pin number of the input or output.

5.0 ALU-R LOGICS

SHEET	CONTENTS
1	IC Locations
2	ROM Simulator Connectors
3	Backplane Connectors
4	WCS Write Data and Address Registers
5	WCS Address Select/Readback Registers
6	WCS Data Buffering - WCS Control
7	WCS RAMs
8	Microstore ROMs
9	ALU Pipeline Register - WCS ROM Control
10	ALU(2901Bs)
11	ALU Shift MUX - ALU Carry Control - Saved Flags
12	ALU Bypass - Y Bus Buffering

TABLE 5-1
ALU-R Board Index

SHEET 1

The block diagram presented indicates IC locations on the board. The horizontal axis is lettered A-M while the vertical axis is numbered 1-9. Each block is divided into sections showing the manufacturer's code for the IC(s) in the narrow section at the top of the block. The sheet(s) on the schematics where the particular IC(s) is(are) located is(are) shown in the bottom section(s) of the block. For example the IC at board location A2 is a 74S244 and is shown on sheet 9 of the schematics. The three ICs at board location E2 are 74S04s and are shown on sheets 11, 9 and 9.

SHEET 2

Sheet 2 is a signal listing for the ROM Simulator connectors (J5&J6) showing the signal mnemonic on the left, the pin number in the center, and the sheet location of the signal source or destination on the right.

SHEET 3

Sheet 3 is a signal listing for the backplane connectors(P1&P2) showing the signal mnemonic, pin number and sheet location of the signal source or destination.

SHEET 4

The WCS Write Data Register(8A through 8F) is loaded two bytes at a time from the 16-bit Y bus and outputs all eight bytes at once to the 64-bit WCS Data Bus. Data is sequentially clocked into each register pair(WCS Word1, then WCS Word2, etc.) on the rising edge of low-active Y Destination enables(WCSD1-4). The clock enable for WCS Word 4(WCSD4) generates the low-active output enable(WCSWD) for the WCS Write Data Register (See Sheet 6).

The WCS Address Register is four cascaded 4-bit binary counters (8L,9L,8M&9M). The load enable(WCSA) is generated by the Y Destination Decode on the AUX Board at T3H, and the address will be clocked to the outputs on the rising edge of the inverted 1/2 Clock which parallels T3H(at T3.5H).

On any read or write to WCS, MUXSEL will be high, and at T3H, AND Gate 1M will generate a high-active count enable for the counters, incrementing the WCS Address. The incremented address will be valid on the rising edge of the inverted 1/2 Clock. A load enable to the counters disables the count function, preventing data contention.

Since the WCS RAMS are presently only 2K X 8, only 11 bits are required for addressing. Two bits(WR/W16 and WR/W17) are dropped from the most significant byte of the WCS Address Register(A1). Three additional bits are dropped from the address in other locations in the logic.

Note that WR/W14-17 correspond to the Microstore Bank Select bits(MSBS0-3) in the Microstore Address. Only one bank of Microstore ROMs are presently used, and with only one bank of WCS RAMs, the bank select bits will all be 0s.

SHEET 5

The WCS Address MUX(9J,9K,8J&8K) selects the WCS Address from the WCS Address Register or from the Microstore Address Register on the AUX Board. When executing from WCS, the Microstore Address will be selected since the WCS microcode utilizes the Next Address Logic and microstack operations located on the AUX Board.

The select input to the MUX(MUXSEL) will be high on any read or write to WCS and the B inputs(WR/W00-WR/W15) will be selected as the output. MUXSEL will be low when executing from WCS.

The jumper at B6(JMP2) is used to define the RAM size for WCS. Table 5-2 shows the settings for JMP2.

RAM	CONNECTED PINS
2K	2-8 5-11
4K	1-7 3-9 5-11
8K	1-7 3-9 6-12

TABLE 5-2
WCS RAM Switch
Settings(JMP2)

With the 2K RAM size, pins 1 and 7 are not connected and WCSA11 is dropped. The WCS Write Enable(WR) is connected across pins 2 and 8 and becomes WCSA11/WE, which is used as the write enable to the WCS RAMs. Pins 6 and 12 are also not connected, and WCSA13 is dropped from the addressing here. WCSA12 is connected to an unused pin on the WCS RAM socket. Larger RAMs will use this line.

A 64-bit WCS Data Word is clocked into the WCS Readback Register(7A-7H) on the rising edge of the WCS Readstrobe (WCSRDSTB) generated through the Local Control Register. The Local Control Register also generates the output enables for each register pair(RDWCSD0-3) in conjunction with the D Source Decode on the AUX Board. The Local Control Register generates two(2) WCS Select Bits(WCSDSEL0&1 - see Sheet 13 of the AUX Board) which are decoded into the WCS Readback enables (RDWCS0-3 - see Sheet 9) when WCS is specified as a D Source.

SHEET 6

The WCS Buffer(6A-6F) gates WCS data to the Pipeline and other logic when the microengine is executing from WCS. The WCS Buffer enable(WCSBUFEN) will be low when the WCS Flag(MSBS4) is set(high - see Sheet 9). The 64-bit WCS Word exactly parallels the microword in terms of bit designation and function.

Dual flip-flops with preset and clear are used to generate WCS controls. The 74LS74 dual flip-flops feature complementary outputs which can be controlled through clocking the single data input, or through the preset and reset functions. Preset drives the Q output high, while reset pulls the Q output low. Each of the two dual flip-flops contained in the 74LS74 functions independently of the other. Figure 5-1 is a block diagram of the 74LS74.

DUAL D-TYPE POSITIVE-EDGE-TRIGGERED FLIP-FLOPS WITH PRESET AND CLEAR

FUNCTION TABLE

INPUTS				OUTPUTS	
PRESET	CLEAR	CLOCK	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H	H
H	H	T	H	H	L
H	H	T	L	L	H
H	H	L	X	Q_0	Q_0

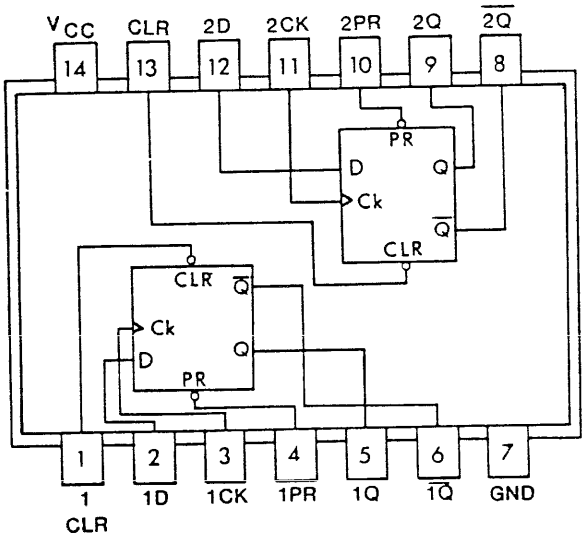


FIGURE 5-1
74LS74 Block Diagram

Throughout this manual the flip-flops are referred to as the lower and upper halves of the IC. The flip-flop using pins 1-6 inclusive is referred to as the lower half, while the flip-flop using pins 7-12 is refered to as the upper half.

The WCS Readstrobe is generated by the lower half of 6L which functions in parallel with the Local Control Register(see the AUX Board, Sheet 13). Local Control bit 2**2(Y02) is clocked into 6L on the rising edge of the low-active Local Control Register enable. When Y02 is high, 6L-6 is the low-active WCSRDSTB. 6L-5 is input to OR Gate 7L, and when high will drive MUXSEL high.

The WCS Write Enable(WCSWD) is generated by the upper half of 7M. Data input 7M-12 is pulled up and is clocked to the outputs of 7M on the rising edge of the load enable for most significant register pair of the WCS Write Data Register (WCSD4). 7M-8 is the low-active output enable for the WCS Write Data Register (WCSD), while 7M-9 becomes the read/write control for the WCS RAMs. The high-active WCSD will be high on a write, disabling the output control of the RAMs. 7M-9 is also input to OR Gate 7L, and when high will drive MUXSEL high.

The low-active output enable for the WCS Write Data Register is clocked into the lower half of 7M on the rising edge of T1H to generate the low-active write enable for the WCS RAMs(WR).

The WCS write enable is reset through the upper half of 7M and NAND Gate 6M. An inverted T3.5H clock is generated through AND Gate 6M. At T3.5H, 7M-4 will go low, presetting 7M-5 high, putting WR in its inactive state.

On the rising edge of the low-active T3.5H clock, MUXSEL is clocked into the upper half of 6L. The 6L-8 output resets the lower half of 6L and the upper half of 7M, making WCSRDSTB and WCSWD inactive. The 6L-9 output is input to NAND Gate 6M. The low-active T3.5H clock is the second input to NAND Gate 6M, and when it goes high, the upper half of 6L is also reset.

An inverted 1/2 CLK is generated through NAND Gate 6M, which has one input(6M-5) pulled up.

SHEET 7

Eight 2K X 8 bit RAM chips make up the WCS RAM. Eight bytes of data are input to or output from the RAMs via the bidirectional WCS Data Bus. The chip enable (WCSCE) is generated when the WCS Flag is set in the microword, or when a WCS Readstrobe or WCS Write enable is generated(see Sheet 9). Because of the switch settings at JMP2(5B6), Vcc/CNTL is high, and WCSA11/WE is the write enable.

NOTE: The pin numbers listed for the WCS RAMs are for the 28 pin socket, not the pin numbers designated for the 24 pin 6116 RAMs on IC data sheets. The 6116 RAMs are mounted as shown in Figure 5-2. The IC pin designation will be 2 less than the socket pin designation shown on the logics. Pin 3 on the logic corresponds to Pin 1 on the 6116 RAMs.

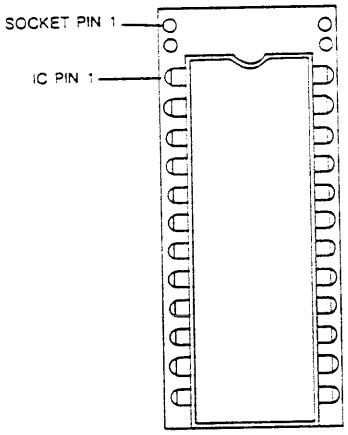


FIGURE 5-2
WCS RAM and Microstore ROM
Mounting

SHEET 8

The Microstore consists of eight 4K X 8 bit EPROMS(3A-3H) which output 8 bytes of data at a time to the pipeline. JMP3 is set according to the size of the Microstore. Table 5-3 gives the switch settings at JMP3.

ROM	CONNECTED PINS
4K	2-7
8K	1-8 3-6
16K	1-8 4-5

TABLE 5-3
Microstore ROM
Switch Settings(JMP3)

As noted previously, MSBS0-2 will be 0s because there is only one bank of Microstore ROMs.

The Microstore Address(MSA00-13) is input from the Microstore Address Register on the AUX Board. The Microstore Address Register is clocked at T0H.

The Microstore ROMs are enabled(ROMEN is low) unless the WCS Flag is set(MSBS4 is high), the ROM Simulator is connected (see Sheet 9), or the Test Panel is connected to the AUX Board.

NOTE: The pin numbers listed for the Microstore ROMs are for the 28 pin socket, not the pin numbers designated for the 24 pin 2732A ROMs on IC data sheets. The 2732A ROMs are mounted as shown in Figure 5-2. The IC pin designation will be 2 less than the socket pin designation shown.

SHEET 9

The WCS Select Bits(WCSDSEL0 & 1) from the Local Control Register on the AUX Board are decoded into the four WCS Readword enables(RDWCSD0-3) by a binary decoder(4A).

A jumper is placed on JMP1 when the ROM simulator is used. The connection will pull SWEXT down. When SWEXT is low, the outputs from NAND Gates at 2D(2D-6 and 2D-8) will be driven high, disabling the Microstore ROM and the WCS RAMs. When the ROM simulator is not connected, the polarity of the WCS Flag(MSBS4) will determine whether the Microstore ROMs or the WCS RAMs are enabled.

When the WCS Flag is set, the high input at 2D-10 will generate the low-active WCS Buffer enable(WCSBUFEN). The high MSBS4 is inverted at 2E, and the low output(2E-12) disables the Microstore ROMs by driving the ROM enable(2D-6). The low input to 2D-2 drives the output (2D-12) high. THe output is inverted to generate the low-active WCS RAM enable(WCSCE). Any read or write to WCS will also generate the WCS RAM enable since the low-active WCSRDSTB or WCSWD will drive 2D-12 high.

The Microstore ROM enable(ROMEN) also enables buffers 2A and 2B which buffer the Next Address Field of the Microword(MSDAT00-17) before it is output through the backplane to next address logic on the AUX Board.

Data from the Microstore ROMs or the WCS RAMs is input to the ALU Pipeline Register(4C,4D,4F,4G&4H), which is clocked at T0H(NOTE: the Test Panel T0H signal is not connected beyond the backplane of the Memory Data Board). Note that the Next Address and Memory fields of the Microword are not pipelined.

SHEET 10

The Q64 ALU contains four cascaded 2901Bs(4J-M). Section 2.1 is a detailed discussion of 2901B function and specifications. Coding of the ALU Controls is discussed in Section 3.2.

The open collector F=0 outputs of the high byte 2901Bs(4L & 4M) are connected together and to a pull-up resistor. Z8H will go high only if the output of both devices goes high.

Note that pin 29 on each device is an input from the Lookahead Carry Generator (see Sheet 11), and that the RAM(pins 8&9) and Q(pins 16&21) Shift lines are bidirectional. Shifts between the most significant 2901B(4M) and the least significant 2901B(4J) are accomplished through the Shift MUX(see Sheet 11). Shift Controls from the Microword are also handled through the Shift MUX.

Note also that the 2901Bs are clocked on the rising edge of the inverted T3H clock signal.

On an ALU Bypass, the Bypass Flag(BYPASS) will be high, and output from the 2901Bs will be disabled. A Bypass may be used any time that ALU calculations or logic are not required. Many functions such as loading the Literal Register or the Real Address Register may utilize the Bypass function. Bypassing the ALU speeds the operation since only one machine cycle is required to load a literal

value into a given Y Destination. The same operation would require an additional cycle if the ALU were not bypassed since the output of the ALU is not available until the end of T3H.

SHEET 11

The Shift MUX(1J,1G,1C,1D&1H) selects the value to be input to the Q Register and RAM at the ends of the 2901B array. ALU Instruction bit I7 selects the right shift multiplexers(1G&1D) when high and the left shift multiplexers(1J&1C) when low. I7 is also a select input to the Save Register MUX(1H) which determines the value to be shifted into the Save Register(1K). ALU Instruction bit I6 is the second select signal to the Save Register MUX and also provides one of the three select signals to the other MUXes. The two shift control bits from the microword are the other select signals to the Left and Right Shift MUXes.

The most significant bit from the most significant 2901B(F16 - the sign bit) and the bidirectional Q Register and RAM shift lines from the least significant (Q0&R0) and most significant(Q15&R15) 2901Bs are active data input lines to the MUXes. Other data input lines are pulled up(e.g. 1G-5) or tied down(e.g. 1J-4) to allow for forced 0 or 1 shifts.

Shifts beyond the 16-bit word length require more than one microcycle. The output of the Save Register MUX(SR) is output through the backplane to the next address branch logic on the AUX Board and latched in the Save Register(1K). ALU Instruction bit I8 will be high on a shift from the Save Register, and the contents of the register will be shifted on the rising edge of the low-active T3H.I8 signal from NAND Gate 7J.

Table 5-4 is a truth table for the MUXes. For a detailed discussion of shift controls, see Sections 2.2 and 3.2.8 of this document.

INPUTS				OUTPUTS	
SELECT			STROBE		
C	B	A	S	Y	W
X	X	X	H	Z	Z
L	L	L	L	D0	D0
L	L	H	L	D1	D1
L	H	L	L	D2	D2
L	H	H	L	D3	D3
H	L	L	L	D4	D4
H	L	H	L	D5	D5
H	H	L	L	D6	D6
H	H	H	L	D7	D7

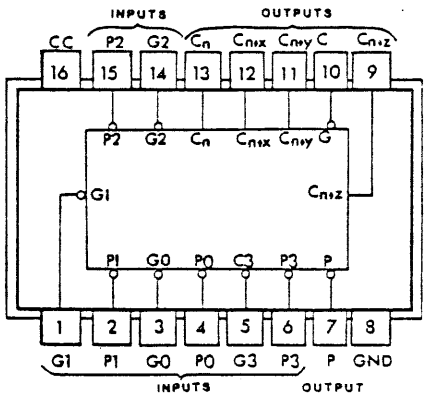
TABLE 5-4
Multiplexer('LS251)
truth table

Carry Controls from the Microword(CC0&CC1) provide the select signals to the Carry Control MUX(2G) which determines the status of the carry input(C0) to the least significant 2901B. Coding of the Carry Control bits is discussed in Section 3.2.6 of this document.

The Lookahead Carry Generator(3J) inputs Carry Generate(G0-3), Carry Propogate (P0-3) and Carry(C0) signals from the 2901Bs and the Carry Control MUX and outputs three carry out signals(C4, C8 & C12). It also outputs propogate and generate signals(P&G) for the next level of carry functions. Table 5-5 lists the logic equations and truth table for the Lookahead Carry Generator.

The logic equations provided at the outputs are:

$$C_{n+x} = G_0 + P_0 C_n$$
$$C_{n+y} = G_1 + P_1 G_0 + P_1 P_0 C_n$$
$$C_{n+z} = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_n$$
$$\overline{G} = \overline{G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_n}$$
$$\overline{P} = \overline{P_3 P_2 P_1 P_0}$$



TRUTH TABLE

INPUTS								OUTPUTS
C_n	$\overline{G}_0$	$\overline{P}_0$	$\overline{G}_1$	$\overline{P}_1$	$\overline{G}_2$	$\overline{G}_3$	$\overline{P}_3$	
X	H	H						L
L	H	X						L
X	L	X						H
H	X	L						H
X	X	X	H	H				L
X	H	X	H	X				L
X	X	H	H	X				L
X	X	X	L	X				H
X	X	X	X	L				H
X	X	X	X	X	H	H		L
X	X	X	X	X	H	X		L
X	X	X	X	X	H	X		L
X	X	X	X	X	X	L		H
X	X	X	X	X	X	X		H
	X	X	X	X	X	X	H	H
	X	X	X	X	X	X	L	L
	X	H	X	X	X	X	X	L
	X	X	L	X	X	X	X	L
		H	X	X	X	X	X	H
		X	H	X	X	X	X	H
		X	X	H	X	X	X	H
		X	X	X	L	X	X	H
		X	X	X	X	L	L	H
		X	X	X	X	X	L	H

TABLE 5-5
Lookahead Carry Generator

A low Carry Generate output from the Lookahead Carry Generator will generate a 16-bit Carry(C16) through AND-OR Invert Gate 7K. A low Carry Propogate will generate a 16-bit carry only if there is no carry input to the least significant 2901B(C0 is low).

The 4, 8, and 12-bit carry flags output by the Lookahead Carry Generator are input to the 2901Bs(see Sheet 10). The 4, 8, and 16-bit carry flags are output through the backplane to the Next Address Logic on the AUX Board.

The 8 and 16-bit carry flags are latched into dual flip-flop 1F. When the ALU Save Status Flag(ALU) is set in the microword, a low-active ALU Clock signal(ALU CLK) is generated by NAND Gate 7J at T3H. On the rising edge of the low-active signal, the C8 and C16 flags are clocked to the outputs of the flip-flops.

The "Junk" Register on the AUX Board generates controls for binary and decimal arithmetic(See Q64 Microword, Section 3.2.7.8). For subtration(SUB is low), both halves of 1F are preset, driving both C8S and C16S high. For addition(ADD is low), the flip-flops are both reset, pulling C8S and C16S low.

The ALU Clock signal(ALU CLK) is also used to latch Accumulated 4-bit and Accumalated 8-bit Zero Flags(AZ4 and AZ8). The "Junk" Register enable(JUNK) presets 1E on both addition and subtraction, driving AZ4 and AZ8 high(true).

AND Gates at 1L generate 8 and 16-bit Zero Flags from the outputs of the 2901Bs. Zero Flags from the two least significant devices(Z4L & Z4H) are ANDed to generate the 8-bit Zero Flag(Z8), which is then ANDed with a Zero Flag from the most significant devices(Z8H) to generate the 16-bit Zero Flag(Z16). Z4L, Z4H, Z8, Z16 and the Accumulated Zero Flags are output through the backplane to the next address logic on the AUX Board.

SHEET 12

The data inputs to the Minus One Register(2H&2K) are tied down so that $FFFF is output to the D Bus when the register is specified as a D Source. Unlike other D Source enables which are generated through decoders, the enable for the Minus One Register(FFFF) is generated through NAND Gate 3M.

On an ALU Bypass operation, the high active enable(BYPASS), disables the output of the ALU Result Register(6J&6K). BYPASS is inverted at 2E, and the inverted signal enables the Bypass Buffers which gate data from the D Bus to the Y Bus Buffers(3K&3L). The ALU Result Register is clocked on the rising edge of T2. The Y Bus Buffers are always enabled and accept data from the Bypass Buffers or the ALU Result Register to output to the Y Bus.

The ALU Clock Buffer(2M) buffers clock signals used on the ALU Board. The clock signals are generated on the AUX Board.

6.0 AUX-R LOGICS

SHEET	CONTENTS
1	IC Locations
2	Backplane Connectors
3	Next Address Logic - Mode and Select Decode
4	Next Address Logic - Special Branches
5	Next Address logic - Branch logic - Literal Register
6	Next Address Logic - 2way and 4way Branches
7	Next Address Logic - Stack
8	Next Address Logic - Microstore Address Register - Control Store Page Register
9	Y Destination Decode - D Source Decode
10	Clock Buffer
11	Reset Logic - Programmable Interval Timer
12	UART - UART Clock - Diagnostic Port Connectors
13	Constant Three - Swap Register - Local Control Register
14	General Purpose Counters - 16 bit and 4 bit Binary Coded Decimal Adder
15	Front Panel Interface
16	IOU Control - IOB Data Buffers
17	Refresh Logic - Pullups

TABLE 6-1
AUX-R BOARD INDEX

SHEET 1

The block diagram presented shows the IC locations on the Auxilliary Board.

SHEET 2

Sheet 2 is a signal listing for the backplane connectors.

SHEET 3

The Next Address Mode and Select fields of the microword(MSDAT14-17 & MSDAT 20-22) are input to binary decoders which generate branch flags. The Next Address Mode Decoder(1G) generates one of three enables required for the Next Address Select Decoders(6H,6J,1F & 2F). The branch enables from the Mode Decoder(2WAY, 4WAY & SPEC) are thereby modified by the enables from the Select Decoders(GPNZC, BRRL, CODE1L, etc.).

The decoders are enabled at T3H; however, only the Next Address Mode Decoder is enabled solely by the clock signal(two low enables are tied down). MSDAT17 acts as a select signal for the Select Decoders. Table 6-2 is a truth table for the 74S138 used for the Next Address Mode and Select Decoders. The branch flags generated by the decoders are output to other next address branch logic.

INPUTS					OUTPUTS							
ENABLE		SELECT										
G1	G2	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

TABLE 6-2
Decoder(74LS138)
Truth Table

SHEET 4

Data is clocked into the Branch Register(8E) on the rising edge of the low-active Y Destination enable(BRR). Output from the register is always enabled(8E-1 is tied down) and the data propogates to two octal buffers(7E & 7D).

FLAG	BRANCH (buffer enabled)
BRRL	- 16 way branch on the lower nibble of the Branch Register (lower 7E)
BRRU	- 16 way branch on the high nibble of the Branch Register (upper 7D)
BRALL	- 256 way branch on the whole byte in the Branch Register (upper and lower 7E)
YBRRL	- 16 way branch on the low nibble of the low byte on the Y Bus (lower 2L)
YBRRU	- 16 way branch on the upper nibble of the low byte on the Y Bus (upper 2L)
YBRALL	- 256 way branch on the low byte on the Y Bus (lower 2L and upper 1M)
CODE1L	- 16 way branch on the low nibble in the CODE1 Register (upper 2P)
DECODE	- 256 way branch on the Fetch Vector (lower 1M and lower 2P)
GLINT	- 16 way branch on macro interrupts (lower 3B)
UINT	- 16 way branch on micro interrupts (upper 3B)

TABLE 6-3
Special Branch Flags

The Special Branch Buffers(7E,7D,2L,1M,3B &2P) are octal buffers that function as dual 4-bit buffers. Pin 1 is the gate enable for the lower half of the buffer(input pins 2,4,6 & 8 and output pins 12,14,16 & 18), and pin 19 is the gate enable for the upper half of the buffer(input pins 11,13,15 & 17 and output pins 3,5,7 & 9).

Special branch flags from the Next Address Select Decoders on Sheet 3 provide for selection of the least significant nibble of the next address on 16 way branches, or the least significant byte of the next address on 256 way branches. The special branch flags enable the selected value to be output from the Special Branch Buffers. Table 6-3 lists the special branch flags and the type of branch. The part of the buffer(s) enabled by the flag is shown in parentheses().

SHEET 5

When the Next Address Field of the Microword(MSDAT00-17) contains a literal, data is clocked into the Literal Register(6F,1E) on the rising edge of the low-active Literal Flag(LITT) output by the the Next Address Mode Decoder. Data is enabled out of the register by a low-active D Source Enable(LIT).

When the inputs to NAND Gate 5D are all high, the low nibble of the least significant byte of the microword(MSDAT00-04) is output unchanged from buffer 5B as the low nibble of the next address. On a branch, one of the inputs will be low and the lower half of the buffer will be disabled. The low nibble of the next address will be determined elsewhere.

On a two way branch, the low-active 2WAY branch flag enables the upper half of buffer 5B. 2WAY DAT replaces the least significant bit of the microword(MSDAT00) as the least significant bit of the next address(NA00). Determination of the status of 2WAY DAT is shown on Sheet 6.

When all of the inputs to NAND Gate 3C are high, the high nibble of the least significant byte of the microword is buffered in the lower half of 2E and becomes the high nibble of the least significant byte of the next address. On a special branch, the lower half of buffer 2E is not enabled and the high nibble of the least significant byte of the next address is determined by the logic on Sheet 4.

On a 4 way branch the lower half of the buffer is enabled and The high two bits of the low nibble of the microword(MSDAT02,03) are buffered and output as NA02 and NA03. The least significant two bits of the next address on 4 way branches are determined by the logic shown on Sheet 6.

SHEET 6

The 2Way Branch MUX(4E & 2K) selects the flag which is Exclusive ORed(XOR Gate 5C-3) with the least significant bit in the Next Address Field of the microword(MSDAT00) to determine the status of the least significant bit in the next address(2WAY DAT becomes NA00 on 2 way branches).

If either of the two least significant bits of the Next Address Field of the microword are low on a 4 way branch, the corresponding bit in the next address (NA00 &/or NA01) will be high. Inputs to buffers at 5L(shown at D4 and D2) are pulled up. 5L-1 and 5L-4 are high-active output enables which are generated by

NOR Gates at 4U(4U-1 & 4U-10) when MSDAT00 or MSDAT01 are low and the 4 Way Branch Flag(4WAY) is low. If either MSDAT00 or MSDAT01 is high, then the 4 Way Branch MUX(5F,3K,5K & 6G) will determine the status of the replacement bit in the next address.

The most significant bit of the Next Address Select Field of the microword (MSDAT17) determines which of the two 2 Way Branch MUXes will be enabled, while MSDAT17 and the least significant two bits from the microword determine which of the 4 Way Branch MUXes is enabled. OR Gates at 5J generate the strobe signals for the 2 Way Branch MUXes, and NAND Gates at 5D and 4B generate strobe signals for the 4 Way Branch MUXes. (Table 5-4 is a truth table for the 74S251 MUX). Note that on 4 way branches MSDAT00 and/or MSDAT01 must be high or it is replaced by the logic discussed above.

MSDAT14-16 are the select inputs to the MUX(es) and determine which input will be the Y output of the MUX(es).

The data inputs to the branch MUXes are miscellaneous condition flags from the ALU, General Purpose Counter and other logic. Note that FCH37 and FCH67 are the indirect address flags for OP2 and OP3.

XOR Gate 5C and OR Gates 5J shown at A5 and A6 provide logic to be used with P Code which is not yet implemented.

SHEET 7

Writes to the Stack(3J,3H,2H,2G & 6L) occur on a call to a subroutine, while a read is done on a pop or return. Stack controls are generated through dual flip-flops with preset and clear(4R,5E & 5P) and the Stack Pointer(6M).

The current Microstore Address(MSA00-13 & MSBS0-3) is clocked thru the Stack Address Register(1J & 1H) on the rising edge of T3H. The inverted outputs of the Stack Address Register then propogate to the Stack RAMs(3J,3H,2H,2G, & 6L) together with the WCS Flag(MSBS4) which is inverted through the upper half of 5E(also clocked at T3H).

The Stack Pointer is a 4-Bit up/down counter which loads 0 on IPL. The inputs to the counter are tied down and the IPL Reset(RST4) is the load enable to the counter(through AND Gate 6P which has one input pulled up). The outputs of the counter are triggered by the low-to-high transition of either count(clock) input. Therefore the first RAM address output by the Stack Pointer will be 0001, and the 16th RAM address output will be 0000, with carry output setting the Full Flags(FULL) through flip-flops at 5P.

On a call - assuming the Stack is not full - the low-active Call Flag(CALL) will enable the count up function of the Stack Pointer, and the incremented value will be output to the Stack RAM address lines(A-D).

The Stack Read/Write enable(R/W) is generated by dual flip-flop 4R and will be high on a read and low on a write. At T2H, both halves of 4R(at C6 & C7) are preset, setting R/W to read(high).

The low-active Call Flag(CALL) is generated at T3H by the Next Address Mode

Decoder on the ALU Board. The low will be clocked thru the upper half of 4R on the rising edge of T0H, then clocked thru the lower half of 4R on the rising edge of T1H. A write to the Stack occurs when R/W goes low(at T1H).

As noted previously, the 16th write to the Stack will be at address 0000. When this occurs, the Stack Pointer rolls over and outputs a low-active carry output(6M-12) that resets the lower half of flip-flop 5P(at A7), generating active(both high an low) Full Flags(FULL).

The high-active Full Flag(from 5P-6) is input to OR Gate 4A(4A-9), which disables the count up function of the Stack Pointer by maintaining 4A-8 high regardless of the status of the Call Flag. The high-active Full Flag is also input to the next address branch logic, and it disables the preset of the upper half of 5P. (Note as long as 5P-10 is low the clocking mechanism of the flip-flop is disabled).

The CALL which caused the rollover of the Stack Pointer also causes R/W to go low. At T2H, preset of 4R drives R/W high, enabling the clocking of the low-active Full Flag thru the upper half of 5P. The low output(5P-9) latches 4R in the preset mode(thru AND Gate 6P), disabling the clocking mechanism of the flip-flop and preventing R/W from going low until the Stack full condition is resolved through the microcode.

On a pop or return, the Stack Pointer is decremented, causing the address to the Stack RAMs to be decremented and a new Stack Address(ST00-20) to be output by the RAMs.

The low-active Return Flag(RET) is generated by the Next Address Mode Decoder on Sheet 3 at T3H. RET selects the Stack Address plus the offset specified in the microword as the Microstore Address(see Sheet 8), and it enables the count down function of the Stack Pointer if the Stack is not empty.

On a pop, the address on the Micro Stack prior to the count down(the last address written to the stack) is ignored. The low-active Pop Flag(POP) is generated by the Next Address Mode Decoder on Sheet 3.

A high-active Empty Flag(EMPTY) is generated by NOR Gate 6K when all of the inputs are low. This will occur when the Stack Pointer output is 0000 and the Full Flag is not set. EMPTY is output to the next address branch logic and input to NOR Gates 5N(5N-9 & 5N-3) where it prevents the generation of the low-to-high transition required for the Stack Pointer to count down.

SHEET 8

The least significant byte of the Next Address(NA00-07) is added to the last address written to the Stack(ST00-20) by the Offset Adders(4K & 5G). On a return from a subroutine, the least significant byte of the Next Address Field of the Microword(MSDAT00-07) will contain an offset value(between $00 and $0F). MSDAT00-07 is buffered and becomes NA00-07 prior to being input to the Offset Adders and the MUXes. The Stack Address plus the offset are the A inputs to the Next Address MUX(4L,7G,4G,3G & 6N) and will be selected on a return(RET will be low), otherwise the B inputs to the MUX will be selected as the outputs.

The B inputs to the MUX are the resolved low byte of the next address from the

next address branch logic(NA00-07), the high nibble of the next address from the Microword(MSDAT10-13), and the bank select and WCS Flag from the ALU(Y00-04). The address selected by the MUX is input to the Microstore Address Register(4J,7H&4H) and the Page Register(3F).

The Microstore Address Register consists of three cascaded binary counters which are loaded with the resolved next address from the branch logic, except when the Next Address Field of the Microword contains a literal constant. Count up is always enabled(tied up); however, when the counters are loaded(pin 9 is low), the count function is disabled.

The low-active Literal Flag(LITT) is inverted through NAND Gate 7K and causes the load to be disabled on a literal, and the counters increment. The Microstore Address Register is clocked on the rising edge of T0H.

The Microstore Bank Select bits(MSBS0-3) and the WCS FLag(MSBS4) are clocked thru the Page Register(3F) on the rising edge of a low-active signal from AND Gate 7N-11 on a return from a subroutine(RET will be low) or when the Page Register is specified as a Y Destination.

As noted previously, MSBS0-3 will all be 0s since there is only one bank(page) of WCS and Microstore.

SHEET 9

The Y Destination Decoder(3E,3N,4C & 4N) generates enables from the Y Destination Field of the Microword. For a complete listing of Y Destinations and the coding of YA00-04 see the Section 3.2.7. Table 6-2 is a truth table for the 74S138. The Y Destination Decode is enabled at T3H.

The D Source Decoder(4F&3M) generates enables from the D Source Field of the Microword(DA00-04). For a complete listing of D Sources and the coding of DA00-04, see Section 3.2.2.

SHEET 10

Discrete Clocks used throughout the system are generated by a 20MZ Hybrid Oscillator(2S). While the voltage shown indicates a 5V Battery power supply, the battery has not been implemented but the 5V power supply utilizes the same line.

The oscillator generates a pulse with a 50NS period which is used as the 1/4 Clock signal(1/4CLK and 1/4CLKA). The External Clock signal (XCLK) and the External Clock enable(XCLKEN) are used only in quality assurrance testing.

The 1/4 clock signal output from 2T-8 is used as the clock enable for flip-flops 2U, 3S, and 4S. The 50NS clock(1/2CLK), and the 100NS clocks(T0,T1,T2&T3) are generated from the 1/4 clock through 2U and select MUX 3U.

On the rising edge of the first 1/4 clock enable to 2U, all of the inputs will be low, and all of the outputs will be low. The low 1/2CLK output(2U-2) is looped through an inverter(1T) and a high is then presented as an input(2U-3). On the rising edge of the second 1/4 clock enable to 2U, the high input will be clocked to the 1/2CLK output. The net result of the looping is a toggling of the 1/2CLK output from low to high every 50NS.

The 1/2CLK output from 2U-2 is used as the select signal for 3U which generates the 100NS clocks. The 5,6,9, and 12 outputs from 2U are input to NOR Gate 1S to generate an initial high input to 3U(3U-11).

The select signal to 3U will initially be low and the high 3A input will be the initial 3Y output of the MUX. This output then generates a high T0 clock output from 2U-5 which is then looped back to the 3B and 2A inputs to MUX 3U. When the 1/4 clock goes high, the B inputs to the MUX will be selected as the outputs. 3Y will remain high, resulting in a 100NS T0 clock signal output from 2U.

When 1/2CLK goes low, the A inputs to the MUX will be selected, at which time 2A will be high, and the 2Y output will be high, generating the high T1 clock output at 2U-5. Parallel loops to that noted on the T0 clock output from 2U cause the high signal initially generated by NOR Gate 1S to be shifted through the MUX, generating the four 100NS clock signals. The 3A&B inputs(2U-11&10) generate the 3Y output which is the T0 clock signal; the 2A&B inputs(2U-5&6) generate the 2Y output(2U-7) which is the T1 clock; the 1A&B inputs generate the 1Y output which is the T2 clock; and the 4A&B inputs(2U-14&13) generate the 4Y output which is the T3 clock.

The Y outputs of 2U are input to the Hold Clock MUX(3T) which disables the hold clocks(identified by a trailing H) when a clock hold is asserted. The 1Y output of 2U is the T0H clock; the 2Y output is the T3H clock; the 3Y output is the T1H clock; and the 4Y output is the T2H clock.

When the Map Control logic asserts a clock hold, the Clock Hold Flag(HOLD) will be a high select signal to the MUX and the B inputs to the MUX will be selected as the outputs. The B inputs to the MUX are tied down, generating low outputs for all hold clock signals when the B inputs are selected.

The Clock Hold Signal(HOLD) is generated through NAND Gate 5T and is delayed until T2.5 through flip-flops 4S and 3S. Likewise, when the hold condition is removed, the change will be delayed until T2.5.

The Help Request Flags(HREQ1-2)input to NAND Gate 5T are for use with an expanded IOU(not yet developed), and the lines are pulled up. The Error Correction Flag(ECC) is for use with error correction(not yet implemented) and is also pulled up. Low-active Memory Refresh(RFSH) or Map Hold Flags(MAPHLD) from the Memory Data Board will drive the output of 5T(5T-8) high, generating the high-active Clock Hold Signal.

Dual flip-flop 6S generates two low-active inputs to NAND Gate 5T(HALT & CONT O/P). The high-active HALT signal from local control is effectively inverted by the upper half of 6S, which is clocked at T0. In addition to being input to NAND Gate 5T to generate the high-active HOLD signal, the low-active HALT signal is input to the decimal display on the front panel(See Sheet 15).

The clock enable and data input to the lower half of 6S are pulled down, disabling the clocking function of the flip-flop. The low-active Front Panel Halt Flag(CONT O/P) is generated by reset of the flip-flop when the halt switch on the front panel is released, generating the low-active Haltswitch Flag (HALTSW). Like the HOLD signal, the CONT O/P signal is input to NAND Gate 5T and to the decimal display on the front panel(see Sheet 15). The signals to 6S

are set inactive on IPL by a low-active reset(RST4) or by a continue signal from the front panel(CONT) through AND Gate 7N.

All clock signals sent to other boards are buffered at 1P & 1N. With the exception of T1H, T2H and T3A, all clock signals used on the AUX Board are buffered at 1R. T1H and T2H are clocked into flip-flops 4S and 3S on the rising edge of the 1/4 clock.

SHEET 11

The Q64 Reset logic provides for immediate resets on power-on and delays on IPLs without power-on. Note in reading the logics that the 5v battery backup is not implemented, and the lines showing to the 5v battery are connected to the system 5v power supply.

On power-on, buffer input 7R-15 will be low until resistor R31 is charged (approximately 2.2 seconds), when it will go high. The low output from the buffer(7R-5) resets the lower half of 7S, setting the Coldstart Flag(COLDS) high. The low output of the buffer is also input to NAND Gate 9S-4 and generates a high data input to the lower half of flip-flop 7T. Both halves of 7T are clocked at T2, resulting in a 400NS delay on signals clocked by both halves. The 7T-8 output generates 4 low-active inputs to buffer 7R, which outputs four low-active resets(RST1-4). RST1 is not connected beyond the backplane of the AUX Board. RST2 is used on the ALU and MDAT Boards. RST3 is used on the MAD Board, and RST4 is used on the AUX Board.

COLDS is output to the Next Address Branch Logic and is reset by the most significant bit of the UART Control Register. Note that the reset bit is input to 7S off the Y Bus(Y07), which is clocked by the UART Control Register enable (UCTL). When Y07 is high, 7S-6 will be low(COLDS will be inactive) and will preset 7S, latching the flip-flop and preventing it from being clocked. Since the flip-flop can only be reset on power-on, the Coldstart Flag cannot be set(high) after is has been reset, unless the power is shut off and turned on again.

If there is an IPL immediately after power-on, both 9S-12 and 9S-13 inputs to NAND Gate 9S will go high and the low output will preset the lower half of flip-flop 7T(7T-4). Preset will drive 7T-5 high and the resets will not be delayed for a machine cycle as they were in the previous instance.

A Voltage Comparator(7U) is set up to provide resets in a power fail situation. However, since the 5v battery backup is not presently implemented, the two inputs to the circuit are both connected to the 5v power supply and there will be no difference detected between the inputs to trigger the logic.

The Dual Monostable Multivibrator(9U) provides a delay from IPL to the generation of resets when it is not a power-on situation. The IPL Warning Flag (IPLWARN) invokes a microcode subroutine to reset controllers and IO devices in an orderly fashion before IPL resets.

The A inputs to the multivibrator are negative-edge triggered, and the setups of the external capacitor(to 9U-6 and 9U-7 for 9U-9) determine the duration of the output(9U-5). The Q outputs of the multivibrator are the inverse of the A inputs.

The low-active IPL Signal(IPL) is generated when the IPL switch on the Front Panel or the Diagnostic Panel is depressed and released. The negative transition at 7U-9 generates the high-active IPL Warning Flag(IPLWARN) output to the Next Address Branch Logic. IPLWARN has a duration of approximately 1.6-1.8 seconds. On a cold start, the resets will be generated as noted previously(COLDS and IPLWARN will cause a preset at 7T-4, etc.). Otherwise, the resets will be delayed.

IPLWARN is input to 9U-1, and when the signal fades, the negative transition at 9U-1 will generate a low output at 9U-4.

The low input to NAND Gate 9S(9S-5) drives the output high. The high is then clocked through both halves of dual flip-flop 7T at T2, further delaying the IPL resets one machine cycle, which means that resets are delayed approximately 2 seconds on warm start IPLs.

The Programmable Interval Timer(7A,8C,6D, & 7C) is made up of four cascaded 4-bit up/down counters which are always enabled for count down(pin 1 is tied down). Load to the counters is enabled by a low-active enable(YTIMER). The PIT Timer is clocked on the rising edge of T0.

The low-active ripple carry output from the most significant counter(7C-15) is clocked into the PIT Latch(7S) on the rising edge of T0. The 7S-9 output of the PIT Latch causes reset of the latch, freezing the status of the PIT Timeout Flag(TIMEOUT) until it is deactivated through local control. TIMEOUT is output to the next address branch logic.

The PIT is "reset" or "disabled" by presetting the PIT Latch through the Local Control Register. Presetting 7S will prevent clocking of the data input. Local control bit 2**4 is defined as the PIT disable bit and Local Control bit 2**5 is the reset bit.

The disable signal(PITSTOP) is routed through the Local Control Register(see Sheet 13); however the reset bit is used off the Y Bus(Y05). Y05 is inverted (8L-10) then NORed with the Local Control Register enable(LCL)(8L-13). 8L-1 will be low when either PITSTOP or 8L-13 are high.

The value in the PIT Timer is read back through the PIT Readback Register (7B&7F).

SHEET 12

The two serial port connectors(J2&J3) are diagnostic ports which allow direct access to the Q64 CPU. The JUART Connector(J3) is a serial connector that allows another system to be directly connected to the CPU via an RS-232 cable. The PUART Connector(J2) is a serial connector to a modem to allow communications to the CPU from a remote site.

Table 6-4 lists the signals on the connectors. The Transmit Data Register(8M) clocks 8 bits of data on the rising edge of the low-active enable(YUDATA). Data from the register is input to the UART Transmit Register (8H-26thru 33).

J2 PIN	DEFINITION
2	Xmit
3	Receive Data
4	Request to Send
5	Clear To Send
6	Data Set Ready
12	+12v Power Supply

J3 PIN	DEFINITION
2	Receive Data
3	Xmit
11	Clear To Send

TABLE 6-4
Serial Port Pinouts
(JUART & PUART)

The UART Control Register(9M) inputs eight bits of data from the Y Bus on the rising edge of a low-active enable(UCTL). The UART Control Register bit definitions are listed in Table 3-13.

A receive interrupt or a transmit interrupt will generate the UART Interrupt Flag(UART) output to the next address branch logic. A receive interrupt clocks the high data input of the upper half of dual flip- flop 9D when eight bits of data have been received by the UART(Data Received at 8H-19 is true), and the Enable Receive Interrupt is set through the UART Control Register(9M-6 is high). The receive interrupt is cleared by reset of the upper half of 9D.

The Transmit Data Strobe clocks the high data input to the lower half of dual flip-flop 9G. The output is then ANDed with the Transmitter Holding Register Full signal(8H-22) from the UART. The output of AND Gate 9E(9E-6) is a high-active transmitter interrupt. The transmit interrupt is cleared by reset of the lower half of flip-flop 9G.

The UART Interrupt Flag(UARTINT) is clocked through the lower and upper halves of dual flip-flop 8R on the rising edge of T0. This results in one machine cycle(400NS) delay from the generation of the interrupt to the flag being output to the next address branch logic. The delay allows for synchronization of the free running UART Clock with the system clock.

The UART Receive Register Clock(8H-17) and Transmit Register Clock(8H-40) are both controlled by a clock signal(UARTCLK) generated by the UART Clock(8U). The UART baud rate may range from 50 to 19200 bits per second depending upon the setting of two jumpers(JMP1 & JMP2).

When a jumper connects pin 1 to pin 2 on JMP2, the switch settings on JMP1 will determine the baud rate. A patch from pin 2 to pin 3 sets the baud rate at 300.

The switch settings for JMP1 are shown in Table 6-5.

BAUD RATE	CONNECTED SWITCHES	BAUD RATE	CONNECTED SWITCHES
19200	None	150	1-2
9600	1-2		3-4
7200	3-4		7-8
4800	1-2	134.50	5-6
	3-4		7-8
3600	5-6	110	1-2
2400	1-2		5-6
	5-6		7-8
2000	3-4	75	3-4
	5-6		5-6
1800	1-2		7-8
	3-4	50	1-2
	5-6		3-4
1200	7-8		5-6
600	1-2		7-8
	7-8		
300	3-4		
	7-8		
	7-8		

TABLE 6-5
Serial Port
Baud Rate Settings

The UART Receive Data Register(8G&9F) is enabled by a low-active enable(UDATA). The low byte of the register contains the character received by the UART and is valid only if Data Received(9F-14) is true(high).

The high byte of the UART Receive Data Register contains status bits from the UART and related logics. Table 6-6 is a bit definition of the high byte register(a.k.a. the UART Status Register).

BIT	DEFINITION
2**15	Data Set Ready
2**14	Clear to Send
2**13	Transmitter Holding Register Full
2**12	Transmitter Holding Register Overflow
2**11	Data Received
2**10	Receive Overrun
2**9	Receive Framing Error
2**8	Data Parity Error

TABLE 6-6
UART Status Register
Bit Definition

SHEET 13

The Swap Register(4D & 6B) inputs two bytes of data from the Y Bus and outputs the bytes in inverted significance to the D Bus. Data is clocked into the register on the rising edge of the low-active enable(LSWAP) and output when SWAP is low.

The two least significant inputs(8B-2 & 8B-4) to the Constant Three Register(8B & 8F) are pulled up, while the other inputs are tied down, which causes $0003 to be output when it is specified as a D Source(THREE is low-active).

The Local Control Register(6E&6C) is loaded from the Y Bus on the rising edge of the low-active enable(LCL). The Local Control Register bits are defined in Table 3-2. They are re-defined in Table 6-7 to allow referencing of other logics where some of the bits are used off the Y Bus rather than being routed through the Local Control Register.

BIT	DEFINITION(Location)
2**15	IO Mode(Not implemented)
2**14	Interrupt Daisy Chain(Not implemented)
2**13	Halt
2**12	Interrupt Acknowledge(Not implemented)
2**11	Set P-code mode(Not implemented)
2**10	Enable Macro interrupt
2**9	Set Stop light
2**8	Single step mode
2**7	Enable Fetcher decode(Memory Data Sheet 17)
2**6	Stop Fetcher(Memory Data Sheet 11)
2**5	Reset Programmable Interrupt Timer(AUX Sheet 11)
2**4	Disable Programmable Interrupt Timer(AUX Sheet 11)
2**3	Reset parity error latch(Memory Data Sheet 9)
2**2	Start WCS readback(ALU Sheet 6)
2**1	WCS Readback Register select bit 1
2**0	WCS Readback Register select bit 0

TABLE 6-7
Local Control Register
Bit Definition

Local Control bits 2**12, 2**14 and 2**15 will be used with an expanded 16-bit controller(not yet developed). Bit 2**13 will be used when P Code is developed for the system.

Outputs for Local Control bits 2**2, 2**3, 2**5 and 2**6 are not connected; however, the corresponding Y Bus bit is used in conjunction with the Local Control Register enable(LCL) to affect the logic described. The logic where the bits are used is shown in parentheses() on the preceding list.

SHEET 14

The 4-Bit General Purpose Counter(6T) and the 16-Bit General Purpose Counter (6U,5U,2M&3L) are loaded from the Y Bus. The low-active load enables(GPC2 for the 4-Bit GPC and GPC1 for the 16-Bit GPC) are generated from Auxilliary Board Y

Destination Decode on Sheet 9. The counters are set to count down(pin 1 is tied down).

The GPC zero flags(GPC24, GPC14 & GPC16) are tested in the next address branch logic(see Sheet 6), and the corresponding counter is then decremented. The flags are valid on the rising edge of T3H, and the decremented count is valid on the trailing edge of T3H(the rising edge of the inverted signal). The GPC branch flags(GPNZB4, GPNZC, GPNZ4C & GPCZ) provide the count enable(through AND Gate 7M-8).

The GPC flags are initialized in their inactive states at IPL by preset of the GPC Flag Registers(4T & 7L) by RST4.

The BCD Adder(8K) is a Lookup ROM which outputs a decimal value stored at an address specified by the two values to be added or subtracted, the BCD Function Flag(-/+), and carry from the previous operation. Since the high nibble of any decimal value is a 3, only the low nibble(Y00-03) of each operand is used to address the ROM.

The Operand 1 is clocked by a latch on the rising edge of the low- active BCD Operand 1 enable(BCD1). The BCD Operand 2 enable(BCD2) clocks the Saved Decimal Carry Flag(C10S) and the Accumulated Decimal Zero Flag(AZ10) while the second operand directly addresses the ROM.

The "Junk" Register is actually only two OR Gates(4A-3 & 4A-6). The Junk Register enable(JUNK) generated by the Auxilliary Board Y Destination Decode (see Sheet 9) and the least significant bit on the Y Bus generate signals to initialize binary and decimal addition and subtraction.

Addition is initialized by setting Junk to 0(Y00=0) which generates the low-active Addition Flag(ADD) at 4A-3. ADD sets the saved decimal carry flag(C10S) to false(0) through reset of the Decimal Carry Latch(5E) and sets the BCD Adder in the add mode(drives -/+ high) through preset of the BCD Function Latch(9D). The Addition Flag is output through the backplane to the ALU Board where other flags are set to initialize binary addition.

Subtraction is initialized by setting Junk to 1(Y00=1) which generates the low-active Subtraction Flag(SUB) at 4A-6. SUB sets the saved decimal carry to true(1) through preset of 5E and sets the BCD Adder to the subtraction mode through reset if 9D. The Subtraction Flag is also output through the backplane to the ALU Board where other flags are set to initialize binary subtraction. For decimal addition and subtraction, the Accumulated Decimal Zero Flag(AZ10) is set to false(1) by reset of 5M by the Junk Register enable(JUNK).

The BCD Result Register(8P) is clocked on the rising edge of the low-active enable(BCDRES) generated by the Auxilliary Board D Source Decode(see Sheet 9). Note that the high nibble of the BCD Result Register is preset to output 3 as the high nibble of the BCD Result.

SHEET 15

J1 is the connector to the Q64 Diagnostic Panel. Signals to the Diagnostic Panel are shown on the left side of Sheet 15 and signals from the Diagnostic Panel are shown on the right.

Data to the Diagnostic Panel Digital Display is buffered by a transparent latch(8N) and a buffer(9R). Eight data bits(Y00-07) (FPI0-7) and the Strobe enable(FPCNT) for the Digital Display are used to display hexidecimal values on the Diagnostic Panel. PARR, CONT O/P, HELP, MINT, and HALT are displayed as decimal points on the Digital Display. Two clock signals(T0H and T1) are used in the Diagnostic Panel logics.

RST4(generated at IPL) and an IPL light enable(NIPL) combine to briefly light the IPL light on the Front Panel during an IPL. The Start and Stop lights on the Front Panel are controlled by the STOPLITE signal. One of these lights will be on when the system is on. STOPLIT(Pin 38 on J1) is also used to light a decimal display on the Digital Display.

Data input from the hex keypad on the Diagnostic Panel is decoded on the Diagnostic Panel and input to the Diag Panel Readback Register.

The Switch Signal Latch(9N) is an Octal Buffer/Line Driver with inverted outputs which latches the status of signals from switches on the Front Panel or the Diagnostic Panel. Complementary signals from the switches are paired inputs to the buffer.

The Normally Open(N/O) switch signals are connected directly to the inverted outputs of the Normally Closed(N/C) switch signals. The inverted outputs of the N/O signals are connected to the inputs of the N/C signals. The inverted outputs of the N/C signals are then connected to the inputs of the N/O signals, forming a latch.

When a switch is depressed and released, the N/O signal will be pulled low. The direct line output will go low, while the N/O input to the buffer will be inverted, driving the corresponding N/C input high. The high N/C input is inverted, maintaining the the low status of the N/O signal and the corresponding flag. The latched signals are reset by logic on the Diagnostic Panel.

IPL, CONT, HALT and STOP are generated by switches on the Diagnostic Panel. IPL and STOP may also be generated by switches on the Front Panel.

SHEET 16

The IOU Control bits(IOE0-3) from the microword are buffered at 3A and output through the backplane to the ALU Board. They are also input to the IOU Control Prom(2J) with the Memory Refresh enable(RFSH) to generate the IOU Strobe signal (STROBE), the IOU Write enable(IOUWRT) and an enable signal for the Service Request Flag(SREQ). The low-active RFSH prevents the generation of the IOU Strobe during memory refresh.

The service request enable signal from the IOU Control PROM and the high active IOU Strobe are input to AND Gate 3R to generate the low-active clock enable for

the flip-flop at 5M that generates the Service Request Flag. On the rising edge of the low-active signal, the most significant bit of the least significant nibble of backplane data(DA03-P) is clocked to the 5M-5 output of the flip-flop. The high-active SREQ is input to the Next Address Branch logic, and looped to the IOU Control PROM.

IO TRICK, originating in the Local Control logic, is used with the IOU24 and causes the posting of the Service Request Flag, altering the address accessed in the IOU Control PROM during IO operations when an IOU24 is connected to the Q64 CPU. The Service Request Flag is set inactive on IPL by reset of the flip- flop by a low-active reset signal(RST4).

The IOU Strobe is latched into the lower half of 4T on the rising edge of the low active T1.5 clock signal generated by NAND Gate 3R. The flip-flop is reset, pulling STROBE low at T3 by a low-active signal generated by NOR Gate 4U. The IOMODE signal is for use with the expanded 16 bit controller and the 4U-12 input is presently tied down, effectively creating an inverter for the T3 clock signal.

Data output to IO devices is clocked into the IO Write Data Register(2B) on the rising edge of the low-active enable (YIODATA). Output from the register is always enabled. The IO Write Data Buffer(1B) is enabled by the low-active IO Write enable(IOUWRT) generated by the IOU Control PROM(2J).

Data input from IO devices is clocked into the IO Backplane Register on the rising edge of a low active signal generated by NAND Gate 5A. 5A-3 will go high, clocking backplane data only in the absence of STROBE and IOUWRT. Output from the register is always enabled(tied low) and the data propogates to the IO Read Data Register(2A). 2A is clocked on the rising edge of T0, and output is enabled when the register is specified as a D Source. (IODATA is low-active).

Buffer 1C and latches 1D, 2C and 1D are not presently used. Their function will parallel those discussed above when a sixteen bit IOU controller is developed.

SHEET 17

The Refresh Clock(5R&5S) is two cascaded 4-bit binary counters which are initialized at $DB by pulling up and tying down the inputs. The most significant output of the counters is the low-active Refresh enable(RFSH) which is also the load enable to the counters.

The counters increment from $DB to $FF+1, at which time the most significant bit will be 0, and the counters will again be loaded with $DB. The counters are clocked at T0 and increment each machine cycle unless a Refresh Hold is asserted. The low-active Refresh Hold Flag(RFSHLD) disables the count function. From the initial load of $DB to $FF+1 requires 38 machine cycles of 400 NS. In the absence of hold conditions, memory refresh is enabled every 15.2 microseconds.

RFSH is output to other logic through a buffer at 2N, and also provides the clock enable for the Refresh Address Register(4PN&3P).

The inputs to the Refresh Address Register are tied down, initializing the Refresh Address at $0000. The two cascaded 4-bit binary counters generate an

addressing range of $00-FF. The counters are clocked on the rising edge of the low-active Refresh enable(RFSH).

Since refresh requires a row address only, this range is sufficient to address all memory regardless of how many modules are installed. Approximately two(2) milliseconds are required to refresh all of memory.

7.0 MAD-R LOGICS

SHEET	CONTENTS
1	IC Locations
2	Backplane Connectors
3	Ribbon Cable Connectors to Memory Data-R
4	LARA
5	LARB
6	LARP
7	LARA, LARB, LARP + LARPO Readback - Base Register Readback
8	Logical Address MUX
9	Base Adders
10	Temp Register - Base Pointer - Base File
11	PARA
12	PARB
13	PARP
14	RAR
15	Physical Address MUX
16	Address A+1 Incrementer - Row and Column Address MUX
17	Map Control and Timing
18	Clock Buffer - D Source Decode - Y Destination Decode - Pullups
19	Offset MUX

TABLE 7-1
MAD-R Board Index

SHEET 1

The block diagram presented indicates the IC locations on the Memory Address Board.

SHEET 2

Signal listings for the backplane connectors are shown.

SHEET 3

Signal listings for the ribbon connector to the Memory Data Board are shown.

SHEET 4

The least significant three bits of Logical Address Register A are implemented using a buffer(6Y) and a latch(6W), while the more significant bits are implemented using four synchronous 4-bit up/down counters(7Y,8Y,9Y & 10Y). This implementation allows the LARA Increment PROM(6X) to specify the value of the least significant three bits and, utilizing only a single increment or decrement of the counters, determine a new logical address on increments and decrements between 1 and 8.

Buffer 6Y is enabled when LARA is loaded from the Logical Address Input MUX on the Memory Data Board(MD Sheet 13). The Logical Address Input MUX selects an address(YLARA00-17) to be loaded into LARA from the Y Bus, Operand 2 or the constant $000F.

On an increment or decrement, the LARA Increment PROM will be enabled, and the least significant three bits will be loaded into the latch(6W) by the PROM. The fourth output of the PROM(6X-9) provides one count enable to the least significant counter.

A low-active load enable for LARA is generated by AND-OR-INVERT Gate 10X(10X-8). When the Fetcher controls the loading of LARA, SEQLDA will be high and will be latched in the Sequence Load Latch(4N) on the trailing edge of T0.5. The high input to OR Gate 9X(9X-9) will drive LDABYP high. LDABYP is input to the map control logic for the physical address registers(see Sheet 17) and is gated with T1.5 through 10X to generate the load enable for LARA.

On an ALU Bypass, the high-active Bypass Flag(BYPASS) from the microword is inverted at 6V and input to NOR Gate 8X. It is gated with the low-active LARA Load enable(YLARA) to drive 8X-10 high. The high is then input to OR Gate 9X to drive LDABYP high.

ALU Bypass or Fetcher controlled loads of LARA meet the minimum setup requirements to be output on the trailing edge of T1(The rising edge of the inverted T1 signal).

When an ALU operation generates the logical address, the ALU output will not be valid until T3H, and a "late load" is required since the physical address register will not be mapped in the same cycle. BYPASS and YLARA will be low inputs to NOR Gate 8X, driving LDALATE high. LDALATE is output to the map control logic for physical address registers(see Sheet 17) and gated with T3.5H through 10X to generate the load enable for LARA. The address loaded at T3H will meet the minimum set up time required to be clocked on the trailing edge of T3.

In addition to enabling the loading of the upper 13 bits of the logical address into the LARA counters, the low-active load enable generated by 10X(10X-8) is input to OR Gate 7V to generate a clock enable for latch 6W at T1 or T3. The load enable also disables the LARA Increment PROM.

The Fetcher or the microword may perform a memory access using a specified address register(A,B,P or R) and cause the address in the register to be incremented or decremented. With the exception of the Real Address Register, register pairs(i.e. LARA & PARA) are incremented or decremented in parallel, allowing the logical address to track the physical address. The signal which enables the output of the address from the Physical Address MUX(e.i. EARA - see Sheet 15) is part of the signal which enables the increment or decrement of the logical and physical address register pair(see below).

Specific increment and decrements are discussed in the preceding sections on the Fetcher branch instructions(Section 2.6.2.2) and microstates(Section 2.6.3), and Memory Control and Memory Increment (Section 3.4.3).

When the logical address is incremented or decremented, the Increment Select

Field(INC0-2) and the Increment Flag(INC) are input to the LARA Increment PROM(6X) with the least significant 3 bits of the logical address(LARA00-02) currently in LARA(from latch 7W). The polarity of INC determines whether the address will be incremented or decremented and the direction of count for the counters that generate the high 13 bits of the logical address. When INC is high, the address will be incremented and the count up function will be enabled for the counters. When INC is low, the address will be decremented and the count down function will be enabled.

While INC enables the direction of the count for the upper 13 bits of the Logical Address Register counters, two count enables(ENP at pin 7 & ENT at pin 10) require low inputs before count can occur. ENP(PLARA) is generated by NAND Gate 5V from the A Address Register enable(EARA), the Increment/Decrement enable(EI/D), the absence of refresh(RFSH will be high) and the T1.5-2 clock signal. EARA and EI/D are generated either by the Fetcher or the Microword decode on the Memory Data Board. The T1.5-2 clock signal is derived from the ANDing of the T1 and T1.5 clock signals(See Sheet 6, AND Gate 2S). EARA is inverted at 2X. When all of the inputs to NAND Gate 5V are high, PLARA will be low. In addition to providing one count enable for the LARA counters, PLARA is also output to the map control and timing logic.

The second enable for the least significant LARA counter(7Y) is generated by the LARA Increment/Decrement PROM(6X). The second enable for each of the more significant counters is the ripple carry output(RCO) of the less significant counter.

The logical address generated by the increment or decrement of the previous logical address will be clocked to the outputs of LARA on the trailing edge of T3.

When increment or decrement of the logical address causes a change in the base number, the ripple carry output from counter 4Y(4Y-15) will be low, setting the Remap Flag(RMAPA). The Remap Flag is input to the map control logic and generates a map enable for PARA at T1.5 which will cause the incremented address in PARA to be overwritten by an address that reflects the new base number.

The upper seven bits of the Base Address Offset(LARA04-12) are input to NAND Gate 7X to determine if the offset is within sixteen(16) bytes of a 2K Base Address boundary. If all of the inputs are high, the LARA Base Hazard Flag (BHAZLA) is set(low).

The PARA Bank Hazard Flag(BNKHAZLA) is generated when the address from Physical Address Register A(PARA00-17) is within 16 bytes of a 64K memory bank(See Sheet 11). If the base or bank hazard flag are set(low), the A Register Hazard Flag (HAZAL) will be set(high) through NAND Gate 1Y. HAZAL is input to the Branch Logic on the Auxilliary Board.

SHEET 5

The logic for LARB parallels the logic for LARA. With the exception of the Sequence Load Latch and the inverter for the ALU Bypass enable, the logics are identical.

Note that LARB and LARP share a common input bus as shown by the signal names (YLARBP) input from the Logical Address Input MUX.

SHEET 6

The logic for LARP parallels the logic for LARA and LARB with the exceptions noted on Sheet 5 and AND Gate 2S which generates the T1.5-2 clock signal.

SHEET 7

Output from the readback registers for LARA(6S&5T), LARB(7S&5S), LARP(6R&6R), and the Base Register(7P,7R&7T) is enabled by low-active signals when the register is specified as a D Source. Two sequential reads are required to gate the 24 bit contents of a Base Register (Base00-27) onto the 16 bit D Bus. The Base Register Select bits(BRSEL0-3) are read back with the low byte of the Base Register value(BASE00-07). The Base Register Select Readback Register is shown on Sheet 19.

LARP is the Program Counter used by the Fetcher in memory operations. The Indirect Counter is reset on each Fetch. The high-active reset(RST INDCNT) is gated with the T1H clock at NAND Gate 3J to generate the clock enable for LARP0 (6P&5P). The "old" contents of LARP(LARP00-17) are loaded into LARP0 on the rising edge of the low-active signal from NAND Gate 3J(3J-11). Output is enabled by a low-active signal(LARP0).

SHEET 8

The address from each of the logical address registers is buffered and output via the LAR Bus to the Base Adders and Base Register. Low-active Map enables (MAPA,MAPB&MAPP) for the physical address registers enable the output for the logical address register paired with the physical address register to be mapped.

SHEET 9

The Base Adders(7K,7L,7M,6M,6L&6K) are six ALU Function Generators that are set to add the A inputs to the B inputs. The 11 bit offset from the logical address (LAR00-12) is added to the 24 bit value from the Base Register(BASE00-27). Full fast simultaneous carry generation is provided through the use of two Lookahead Carry Generators(7N&6N).

Note that the most significant input to device 7M(7M-16) is tied down. This limits the displacement added to the value in Base Register to $7FF(2K).

SHEET 10

The Base File(9K,9L,9M,9N,9P&10R) consists of six(6) 256X4 RAMs. On an initial write to a Base Register, an eight bit address is loaded into the Base Pointer (10L&10M) from the Y Bus. The low nibble of the base register address(BRSEL0-3) determines which set of 16 Base Registers will be selected, while the high nibble(LAR13-16) determines which of 16 registers within that set will be used. Note that only one set of Base Registers is presently supported by the software; therefore, the BRSEL0-3 will be 0s.

The low nibble of the Base Pointer is a counter(10L). It is usually loaded with zero and increments on each successive write to a Base Register. The load enable

for the Base Pointer(BPTR) is low-active. The high nibble of the Base Register address is clocked to the outputs of the Base Pointer on the rising edge of the low-active signal; however, the low nibble will be clocked to the outputs on the rising edge of T0H.

In the Y Destination Decode shown on Sheet 18, two low-active enables(YBASE & YBASEA) are generated for a write to the Base Register during one decode. One of the two decoders(1V) requires an inverted T3H clock signal to generate the Base Register Write enable(YBASE), while the other decoder(1U) is not gated with a clock enable and will generate the Base Address enable(YBASEA) prior to T3H. YBASEA is one of three inputs to AND Gate 10N which provides the select signal to the Base Register Select MUX(10P&10R). When YBASEA is low, the select signal will be low and the A inputs to the MUX will be the selected outputs. The A inputs are the outputs from the Base Pointer. Subsequently, the write enable for the Base Register(YBASE) will be generated, at which time the address selected on the PROMs will be the address from the Base Pointer. YBASE is inverted at 6V and is the increment enable to the Base Pointer. The incremented Base Register address is valid at T0H following the write to the Base Register.

Two machine cycles are required to write a 24 bit value to a Base Register from the 16 bit Y Bus. On the first cycle, an eight bit value(Y00-07) is loaded into Temp Register(10K) on the rising edge of the low-active Temp Register enable (YTEMP). Output from the Temp Register is input to the two least significant Base File RAMs. On the second cycle, the Base File Write enable(YBASE) is generated and a 16 bit value from the Y bus(Y00-17) is written directly to the more significant RAMs at the same time as the value from the Temp Register(TEMP00-07) is written.

On a readback of a Base Register, the Base Register Address is first loaded into the Base Pointer. The enables for the Base Register Readback(BASE&BASEL) pull the select signal to the Base Address Select Mux(10R-1) low, which selects the A inputs (the output of the Base Pointer) as the high nibble of the Base Register Address.

SHEET 11

The logic of PARA parallels the logic of LARA discussed previously(See Sheet 4). The 24 bit Physical Address(PARA00-24) is incremented or decremented in parallel with the Logical Address. The same signals(EARA,E I/D & RFSH) generate the count enable, and the PARA Increment PROM(8B) functions the same as the LARA Increment PROM(disabled by load enable to the counters, etc.). Note that the increment or decrement of the physical address is not gated with a clock signal.

PARA is loaded at T3.5 when a PARA Map enable(MAPA) is generated in the map control and timing logic. The physical address is valid on the trailing edge of T3(the rising edge of the inverted signal).

The Physical Address(PARA00-24) can be broken down into specific memory address components. PARA00-02 corresponds to the Byte Offset(MOFF0-3) which determines where the address starts within a memory word; PARA03-12 to the row address; PARA13-22 to the collumn address; and PARA25-27 to board and module selects. PARA23&24 are to be used with the 265K memory chips(not yet implemented).

PARA03-17 are input to NAND Gate 5E to generate a low-active PARA Bank Hazard

Flag(BNKHAZLP). All of the inputs will be high when the address generated is within 16 bytes of a 64K block of memory.

SHEET 12

The logic for PARB is the same as that for PARA.

SHEET 13

The logic for PARP is the same as that for PARA and PARB.

SHEET 14

The Real Address Register(RAR) provides direct memory access via the loading of an address from the Y Bus. Two machine cycles are required to load the 24 bit address from the 16 bit Y Bus. On the first cycle, the upper eight bits are loaded into the Temp Register(6J) on the rising edge of the low-active enable. On the second cycle the lower 16 bits are loaded from the Y Bus directly to the RAR, and the outputs of the Temp Register are loaded into RAR.

The RAR address may be incremented or decremented and the logic parallels the increment/decrement logic of the other address registers.

The "Real" Address(RAR00-27) can be broken down into specific memory address components with the bit designation paralleling that of the Physical Address(see discussion of Sheet 11).

SHEET 15

Address selection from the Physical Address MUX is determined by low-active address register enables(EARA,EARB,EARP,or EARR) generated by the Fetcher Memory Control PROM or the Microcode Memory Control PROM on the Memory Data Board. The address from the MUX is input to the Row & Column Address MUX and the Address A+1 Incrementer(see Sheet 16).

The upper 5 bits of the Real Memory Address(RMA23-27) are output thru the backplane to the Memory Data Board. The Byte Offset(MOFF0A-3A) is output through the backplane to the Memory Mother Board(MEM64A). An Offset Mux is shown on Sheet 19. The least significant three bits of the Physical Address are input to a buffer and enabled out by the corresponding address register enable(EARA,etc.).

SHEET 16

The Row and Column Addresses(A0-7) are generated from 16 bits of the Real Memory Address(RMA03-22) in the Row & Column Address Mux(1B&1A) and are output to the Memory Mother Board and the Parity Error Address latch on the Memory Data Board. The same address lines are used for both Row and Column Address, but conflict is avoided since the enables for the buffers(ENCOLADD & ENROWADD) are generated at two different timing states. ENROWADD is generated at T0 while ENCOLADD begins at T1+60NS, when ENROWADD becomes inactive(see Memory Data Board Sheet 13).

The Address A+1 Incrementer(2B,3B,3F&2F) is made up of four ALU Function Generators which are set to add the A inputs to the B inputs. The B inputs to all four devices are tied down; however the pullup on the carry input to the

least significant device(2B-15) causes 1 to be added to the Real Memory Address. Full fast simultaneous carry is provided through a Lookahead Carry Generator (1F).

Row and Column A+1 Addresses(A+10-17) are buffered and output concurrently with the Row and Column Addresses(A0-A7).

The ninth row and column address bits(A8 & A18) are unaffected by the A+1 Address logic. A8 and A+18 will be the same until 256K memory chips are implemented. CASCLK is not connected beyond the backplane.

SHEET 17

The Map Control Clock Generator(1P) is a 4-bit Parallel Access Shift Register which is set to synchronous shift by pulling up the shift/load control input (1P-9). The T0 clock signal is serial input to the shifter. The shift register contains a series of positive edge triggered flip-flops and an AND-OR-Inverter which shift the input to each of the four outputs on the second positive transition of the clock enable(T1/4). The transition does not occur on the first clock because the transition of T0 parallels the transition of the clock enable and the data input will not be logically high in time to meet the minimum setup time requirements to be transferred to the data output. On the rising edge of the next 1/4 clock, (50NS later), the input will a stable high and will be transferred to the output. Parallel setup and transfer occurs as the signal is shifted to other flip-flops within the Shift Register. The four 1/2 clocks output are input to a series of NAND Gates(1L &1J) which gate the Map enables with specific 1/2 clocks.

Map controls for each of the three physical address registers are the same. Under specified conditions, low-active and high-active map enables will be generated by flip-flops at 3N or 3L. Register priority is established through NAND Gates 1L and 2M. The low-active map enables for one register prevent the generation of map enables for other registers through reset of the map enable flip-flops(3N & 3L). When this occurs, map pending flags are set for the register requiring mapping when the circuit is busy.

Generation of a map enable for a register clears the pending flag for that register through reset of the corresponding pending flip-flop(3L or 2N).

On a Fetcher or ALU Bypass load of the logical address register, the logical address is valid on the trailing edge of T1. The LDABYP signal is input to NOR Gate 1J and at T2 the STBMAPA signal will be generated though NAND Gate 1K. STBMAPA will clock flip-flop 3N to generate the active Map enables for Physical Address Register A(MAPA), unless the mapping circuits are busy.

If the mapping circuitry is busy, then the low-active map enable from PARB or PARP will be low inputs to AND Gate 3P and will cause reset of 3N, disabling clocking of the flip-flop. MPBUSY will be a high input to NOR Gate 2K, generating STBPENDA which will clock flip-flop 3L and produce active pending flags for PARA(APEND).

When the Logical Address Register is loaded by an ALU function, the logical address will be valid on the trailing edge of T3. LDALATE will be input to NAND Gate 2L and at T3H will generate STPENDA through NAND Gate 2J to clock flip-flop

3L to produce the PARA Pending Flags(PENDA).

When a base fault occurs during the increment or decrement of a Logical Address Register, a remap of the corresponding Physical Address Register will occur. RMAPA and PLARA will be input to NAND Gate 1M and at T1.5 will generate STBMAPA through NAND Gate 1K to clock flip-flop 3N, producing the map enables for PARA(MAPA), unless the mapping circuits are busy.

Pending Flags are resolved at T0.5 or T1.0. Priority is given to mapping physical registers which will be used during the current microcycle. In this case the address register enable (EARA) and the pending flag(PENDA) will be high inputs to NAND Gate 1L and at T0.5 STBMAPA will be generated through NOR Gate 1K to clock 3N, producing the active map enables(MAPA).

Resolution of pending flags for registers to be used in the current cycle cause the hold of the microcycle. The low-active output of NAND Gate 1L(or 2P) drives the output of NAND Gate 2M high, clocking the lower half of dual flip-flop 1N. 1N-6 then presets the upper half of the flip-flop, generating the low-active Map Hold Flag(MAPHLD) from 1N-8. MAPHLD is output through the backplane to the clock hold logic on the AUX Board. This results in the hold clocks(T0H, T1H, etc.) being held from T2.5 to the following T2.5.

The lower half of 1N is reset at T3(inverted at 1M) pulling the Q output(1N-5) low. The low is clocked into the upper half on the trailing edge of T3, driving MAPHLD high. The high MAPHLD signal enables the restart of the hold clocks on the AUX Board.

All Map enables are clocked thru a latch(3K) at T3 and input to NAND Gates 3J. At T0, and active(high) flags will generate a low output from 3J which will deactivate the map enable through reset of 3N and 3L. Note that pending flags are reset only by the generation of corresponding map enables.

Map Control and Timing are discussed in detail and timing diagrams are given in Section 2.9.

SHEET 18

The Memory Address Board Clock Buffer(1R) buffers all clock signals used on the Memory Address Board. T3.5 and T3.5H clock signals are generated through a flip-flop at 1S. On the rising edge of the 1/4 clock signal that parallels the rising edge of T3, the data input will not meet the setup time requirement of the flip-flop. On the next 1/4 clock(50NS later) the setup time requirement will be met and the T3.5 and T3.5H clocks will result.

Two decoders are used for the Memory Address Board Y Destination Decoder(1U&1V). As noted previously, the decoder at 1V is enabled at T3H(an inverted T3H is one of three enables required for the device). YA3 & YA4 provide the other enables for the devices.(The third enable on 1U is tied down).

One decoder is used for the Memory Address Board D Source Decode(1W), which is enabled by DA3 & DA4.

SHEET 19

Readback of the low nibble of the Branch Register Address is discussed on Sheet 7.

The Byte Offest Mux(10J & 10S) is discussed on Sheet 15.

8.0 MDAT-R LOGICS

SHEET	CONTENTS
1	IC Locations
2	Backplane Connectors
3	Ribbon Cable Connectors to Memory Address-R
4	Y to MRD 0 and 1 - Fetch Register WD 0 and 1
5	Y to MRD 2 and 3 - Fetch Register WD 2 and 3
6	Memory Data Buffers
7	Read Bit Shift
8	RWORD, RBYTE, RCAS, LCAS
9	Parity Generator/Det - Parity Error Address Trap
10	Fetch Mode Decode - Movelen D SCR - Fetcher Sequence Loader - P Counter Loader
11	Fetch Sequence Control - Indirect Counter
12	Fetch Vector
13	Main Memory Timing - Logical Address Input MUX
14	Logical Address Input MUX - Indirect Decrementer
15	Write MUX - Write Bit Shifter
16	Memory Control Decode - Main Memory Timing
17	D Source Decode - Y Destination Decode - Local Control Register - Pullups

TABLE 8-1
MDAT-R Board Index

SHEET 1

The block diagram presented shows the IC locations on the Memory Data Board.

SHEET 2

Signal listings for the backplane connectors are shown.

SHEET 3

Signal listings for the ribbon connectors to the Memory Address Board are shown.

SHEETS 4&5

The 64-bit Fetch Register(7A,8C,5P,7M,7L,6M,7K&8J) is the read and write data path from and to main memory. Data to be written from the 16-bit Y Bus to memory is clocked into the Fetch Register from the Write Data Buffers(8H,8K,8G,7N,9L,8M,8L&8E) on the rising edge of the low-active enables (WWORD1-4). Data to be written to memory is output from the Fetch Register to the Write Bit Shifter via the 64-bit Shifter Bus(MRD Bus).

On a Fetch from memory, data on the Shifter Bus from the Read Bit Shifter is clocked into the Fetch Register on the rising edge of the low-active enable (FCHSTB) generated by the memory controls.

In addition to providing a memory path to and from memory, the Fetcher is capable of decoding some macroinstructions and resolvirg indirect addresses on

Operand 1. The Fetcher also provides a "Decode Vector" input to the branch logic for the microcode. The Fetch Register can be conceived as a multipurpose resource partitioned as shown in Table 8-2.

PARTITION	Y DESTINATION	D SOURCE	D SOURCE
BYTE 0	WRITE WORD 1	OPERAND 1	
BYTE 1	WRITE WORD 1	OPERAND 1	
BYTE 2	WRITE WORD 2	OP CODE 1	READ WORD 2
BYTE 3	WRITE WORD 2	OPERAND 2	READ WORD 2
BYTE 4	WRITE WORD 3	OPERAND 2	READ WORD 3
BYTE 5	WRITE WORD 3	OP CODE 2	READ WORD 3
BYTE 6	WRITE WORD 4	OPERAND 3	READ WORD 4
BYTE 7	WRITE WORD 4	OPERAND 3	READ WORD 4

TABLE 8-2
Fetch Register
Partitions

The Read Word and Read Byte Registers are dual purpose registers and are shown on Sheet 8. Fetch Register Bytes 1 & 2 may contain the same data as the Read Word and Read Byte Registers; however since they can be loaded seperately from the Fetch Register, and since Operand 1 may also be loaded independently(on an indirect), the data may vary. For further discussion on the Fetch Register see Section 2.6.

SHEET 6

The bidirectional Memory Data Buffer(1J,1L,1F,1K,1H,1B,2H&1A) gates data to main memory from the Write Bit Shifters(see Sheet 7) and the Parity Error Generator (see Sheet 9) via the 64-bit bidirectional Memory Data Bus(MDAT Bus). It also gates data from main memory to the Read Bit Shifter and the Parity Error Detector(see Sheet 9) via the same bus. The Data Bus between the backplane and the Memory Data Buffers in also a 64-bit bidirectional bus. The direction control to the buffers(MWRITE) will be high on a write to memory and low on a read from memory.

SHEET 7

The Read Bit Shifter(4A,4C,4K,5G,4D,4E,4G&4H) is disabled on a write to main memory. Any low Write Word enable input to NAND Gate 5X will drive WDSAB high, disabling the PROMS.

The Byte Offset(NOFF0-2) from the Memory Address Board is buffered at 9K. The Byte Offset determines the degree of shift required to allign the memory words.

SHEET 8

The Read Word Register(7B&7F) and the Read Byte Register(7G) were mentioned briefly in the discussion of the Fetch Register on Sheets 4 & 5. Whenever the Fetch Register is loaded, Read Word 1 is loaded; however Read Word 1 and Read Byte may be loaded from memory independently of the Fetch Register. Output from the Read Byte Register is the low byte on the D Bus(DB00-07). On a single byte

read, the high byte on the D Bus will be $00 and is output by a register shown on Sheet 10.

The Read Byte Register also functions as part of the Cascade Register(7G,7F,7D &7E). On a cascade, two sequential bytes are clocked into 7J on the rising edge of the first low-active Memory Read Strobe(MRDSTB). The data will prapogate to buffers 7G and 7F and the the inputs to the Cascade Register(7H). Propagation delay through 7J guarantees a minimum hold time for previous data in 7H. Data on the inputs on the first read strobe will be clocked to the outputs and propagate to buffers 7D and 7E on the second read strobe. Simultaneously the second byte will be clocked through 7J and propagate to buffers 7G and 7F.

On a Left Cascade, buffers 7G and 7E will be enabled. The second byte read will be output as the low byte on the D Bus(DB00-07), and the first byte read will be output as the high byte(DB10-17). On a Right Cascade, buffers 7F and 7D will be enabled and the first byte read will be output as the low byte on the D Bus and the second byte read will be the high byte.

SHEET 9

The Parity Generator/Detector(2J,2L,2E,2K,2F,2B,2G&2C) generates a parity bit for each byte of data written to memory and tests the parity of data read from memory using the parity bit generated on the write.

On a write to memory, data(MDAT00-77) is input to the Parity Generator from the Write Bit Shifters. A low-active Memory Write enable(MWRITE) will generate a low I input to the Parity Generator(through AND Gate 2A). The number of high inputs to the Parity Generator are summed to determine the status of the odd(pin 6) and even(pin 5) outputs. The even ouput will be low if the number of high inputs is odd, and it will be high if the number of high inputs is even. The odd output will be the complement of the even output. The even(pin 5) outputs are buffered at 2D and become the parity bits(DATA0P-7P) output to memory with the byte of data which is used to generate the bit. Both odd and even outputs are used in parity error detection.

On a read from memory, data from the Memory Data Buffer is input to the Parity Detector with the parity bit stored with each byte. The number of high inputs is summed to determined the status of the odd and even outputs. An odd number of high inputs will pull the even output low, while an even number of high inputs will drive the even output high. The odd output is expected to be high unless a parity error is detected. The even output is expected to be low unless a parity error is detected.

The Parity Error MUX(1M,1N&2U) allows for testing parity on reads of one, two or eight bytes. Eight bytes are tested on reads of three or more bytes. The Byte Offset(MOFF0-2) is the select field for the one and two byte Parity Error MUXes (1M&1N respectively). The Byte Offset identifies where in the eight byte memory word the first byte is located. The Write Offset(WOFFSET0-2) provide the select signals for the eight byte Parity Error MUX. The Write Offset identifies the number of bytes(range) read or written.

With a zero offset, MOFF0-2 will be 000, and the D0(pin 4) inputs to 1M & 1N will be selected as the outputs. Parity bits from bytes 0 and 1(PERR0 & PERR1) will be selected as the outputs from MUXes 1M and 1N. On a parity error, the Y

output from 1M(pin 5) will be high, giving a high input to MUX 2U.

The W outputs from 1M and 1N(pin 6) are input to NAND Gate 3P. On a parity error one of these inputs will be low, giving a high D1 input to MUX 2U.

On a single byte read, WOFFSET0-2 will be 000 and the D0 input to MUX 2U will be selected as the W output. On a two byte (one "word") read, WOFFSET0-2 will be 001 and the D1 input will be selected as the W output. On a parity error the W output will be low.

On a parity error on reads of more than two bytes, at least one of the inputs to NAND Gate 2M will be low, driving the output(2M-8) high. The D2 through D7 inputs to MUX 2U are all tied to the output of 2M. On reads of greater than three bytes, MOFFSET0-2 will be greater than 001 and will select one of the inputs D2 through D7 as the W output. A parity error in any one of the bytes in reads of three or more bytes will generate a low W output from 2U(2U-6).

The Parity Error Address Register and the Parity Error Latch make up the Parity Error Trap. On all reads from memory, the address is clocked into the Parity Error Address Register(1D,1E &1G). In the absence of parity error that has not been reset, a memory write, or memory refresh, the Row Address Strobe(RAS) will cause the clocking of the memory row address(A0-7) into transparent latch 1G. The Memory Read Strobe(MRDSTB) will clock the memory column address(A0-7), the Byte Offset(MOFF0-2) and the Board and Module Select bits(RMA23-27) into transparent latches 1D & 1E.

When a parity error is detected, the low-active W output from the Parity Error MUX(2U-6) is clocked thru the Parity Error Latch(7U) on the rising edge of the low-active Memory Read Strobe(MRDSTB). The complementary outputs of 7U are high and low-active Parity Error Flags(PARR) which prevent the clocking of any further addresses until the latch is cleared. The low-active Parity Error Flag causes the reset of 7U, disabling the clocking mechanism until the flip-flop is preset. The low is also input to NAND Gate 5X(5X-5) which maintains 5X-6 high and prevents the clocking of 1G.

The high-active Parity Error Flag is input to NOR Gate 8V, pulling 8V-13 high and prevents the clocking of 1D and 1E. The high-active Parity Error Flag is also output through the backplane to the Next Address Logic on the AUX Board and is input to the Fetcher logic.

The Parity Error Address is read back through local control as a D Source. Two reads are required to gate the 24 bit address to the 16 bit D Bus.

WCS Readback Register Select bit 0(WCSDSEL0) from the Local Control Register and the Parity Error Register enable(PERR) must both be low inputs to AND Gate 2W(at C2 and B2) to enable the output.

The Local Control Register(Sheet 17) generates complementary WCSDSEL0 outputs. The low-active WCSDSEL0 is generated when LCL bit 2**0 is high, and output from 1D and 1E will be enabled. The high-active WCSDSEL0 will be low when LCL bit 2**0 is low, and output from 1G will be enabled.

The Parity Error Trap is reset at IPL through AND Gate 7S by RST3 which presets the Parity Error Latch(7U). The Parity Error Latch may also be preset through

the Local Control Register. Bit 2**3 is used directly off the Y Bus(Y03) in conjunction with the Local Control Register enable(LCL).

SHEET 10

The Flag Register(7P) is clocked on the rising edge of the low-active enable (YFLAGS). The Qantel Macro Flags(Zero, Minus and Overflow) are set by a Macro-code routine and tested for macro branches by the Fetcher through the Fetch Mode Decoder(4L). The status of the Qantel Macro Flags can be read from the Qantel Macro Flag Readback Register(5R).

The Fetch Mode Decoders(4L & 6P) decode Opcode 1 and Opcode 2 and output miscellaneous signals used in other Fetch logic. They are hereafter referred to as the Opcode 1 and Opcode 2 Decoders for simplicity.

The Qantel Macro Flags and Opcode 1(FCH20-27) are input to the Opcode 1 Decoder (4L) to determine the type of instruction and output enables for other logic if the instruction can be decoded by the Fetcher. A discussion of the Opcode Decode is contained in the Section 2.6.2.

Opcode 2(FCH50-57) is input to the Opcode 2 Decoder(6P) which outputs increment controls for multiple operand instructions(double or triple).

The Fetcher Increment MUX(6R) selects the Increment Select Field(EINC0-2) for LARP from the outputs of the Opcode 1 or Opcode 2 Decoder, depending upon whether or not the macroinstruction is a multioperand instruction. A multi-operand instruction is defined as having Opcode 1(FCH20-27) equal to $Fx.

The Multioperand Flag(MULTIOP) is the select signal to the MUX, and the B inputs(from the Opcode 2 Decoder) will be selected as the outputs when MULTIOP is high. The A inputs(from the Opcode 1 Decoder) will be selected when MULTIOP is low. MULTIOP is generated from the high nibble of Opcode 1 input to AND Gate 5T(see Sheet 12).

Note that the most significant bit of the Increment Select Field(EINC2) will be high regardless of the selected output from the MUX because the corresponding inputs (3A & 3B) are pulled up. This identifies the Increment Select Field as being a Fetcher increment to the Address Register Increment PROMS on the Memory Address Board.

The output enable for the Increment Select MUX is the inverted Fetcher Continue Signal(FETCH), from the Fetcher Sequence Latch(see Sheet 11). Definition of the Increment Select Field is listed in Table 2-7.

The Fetcher LAR Load Decoder(6V) generates sequence load enables for the logical address registers from the Operand 1 Indirect Address Flag(FCH07), the Fetcher Microstate(FCHSEQ0-4) output by the Fetcher Sequence Latch(see Sheet 11) and the Branch Taken(BRANCH TAKEN) and Double Triple Operand(DOUBLE TRIPLE) flags from the Opcode 1 Decoder. The effects of the Fetcher Microstate on the address registers is shown in the Fetcher Microstate Block Diagram(Figure 2-10) and discussed in Section 2.6.3.

On a six byte move instruction, the low nibbles of the Opcode 1 and Opcode 2(FCH20-23 & FCH50-53) contain the move length. A length of $00 implies a

length of 256($0100). When the move length is not specified, FCH20-23 & FCH50-53 will all be low inputs to NOR Gates 6S. This will generate two high inputs to AND Gate 8R. Since RBYTE will be high(not active) on a move instruction, the least significant input to the high byte of the Move Length Register (6N-2) will be high. Since the remaining inputs to the high byte register are tied down, and the low byte will be 0, $0100 will be output to the D Bus as the move length. The move length is specified otherwise in eight byte instructions and this register is not used.

Register 6N outputs $00 as the high byte to the D Bus on a single byte read from memory. On a Read Byte instruction, the low-active Read Byte enable(RBYTE) will drive the output of AND Gate 8R(8R-12) low, making all of the inputs to 6N low, causing $00 to be output.

SHEET 11

The Fetcher Sequence Latch(3R) is clocked at T0H and T2H. When the Fetcher is idle, the output of the Sequence Latch will be $00 and the Decode Vector output by the Fetch Vector Latch(see Sheet 12) will also be $00. The content of the Fetcher Sequence Latch(FCHSEQ0-4) is referred to as the Fetcher Microstate, which can be directly decoded(see Section 2.6.3).

The Fetcher Microstate(FCHSEQ0-4) is input to the Fetcher Control Decoder(3T) which determines which of the other Fetcher mechanisms are enabled. When the Fetcher is idle, the Fetcher Anticipate Decoder(4N) will be enabled, and output data(FESEQANT0-4) to the Anticipate Bus to be interpreted by other decoders. In the absence of a Start(START) or Decode(FDECODE) signal input to decoder 4N, the Fetcher will continue to loop in the idle state.

The Fetcher Control Decoder also increments and resets the Indirect Counter(4W). The Indirect Counter is a 4-bit binary counter. Load to the counter is disabled, and it is initialized at 0 when it is reset.

On an indirect, the Fetcher Microstate(FCHSEQ0-4) will generate the increment enable(INCINDCNT) from the Fetcher Control Decoder at T2H. The incremented count will be clocked on the trailing edge of T3H(the rising edge of the inverted signal). The Indirect Overflow Flag(INDOFL) will be output from the ripple carry output of the counter on the 16th increment, indicating an excessive indirect chain. INDOFL is input to Decoder 4N and causes the Fetcher to halt. INDOFL is also input to the Fetch Vector logic(see Sheet 12) which generates a Decode Vector of excessive indirect chain before handing off control to the microprocessor.

The Fetcher Memory Control Decoder(4U) generates memory controls from the Fetcher Anticipate Data(FSEQANT0-4). The Fetcher Memory Control Latch(4V) is clocked at T0H and output is enabled by the high-active Fetcher Continue Signal(FETCH). Output from the latch will be disabled by a clock hold for memory mapping(HOLD will be low), or if the Fetcher is in the single step mode or halted due to an excessive indirect chain. Any of the preceding conditions will generate a low input to NAND Gate 3P, driving the output(3P-8) high to disable output.

The Fetch Sequence Latch is reset at IPL by RST3. It may also be halted(reset) at T0 through Local Control Register bit 2**6(Y06). Note that bit 2**6 is used

directly off the Y Bus in conjunction with the Local Control Register enable (LCL) to stop the Fetcher.

Three NAND Gates at 5V are used to generate the halt signal for the Fetcher at T0, then reset the signal at T2H. LCL and the inverted Y06 will generate a low input to 5V-1(through OR Gate 1X. The low input will drive 5V-3 high, and at T0 the Fetcher will be halted through reset of the Fetcher Sequence Latch.

At T2H, the inverted clock signal input to 5V-5 will drive 5V-6 high. LCL will by this time be a high input to OR Gate 1X, driving 1X-3 high. Both highs will be input to 5V-2 and 5V-1 respectively, pulling 5V-3 low. The low input to 5V-13 will drive 5V-11 low, disabling the reset of the Fetcher Sequence Latch and allowing the Fetcher to be restarted on the following T0H clock cycle.

SHEET 12

The Fetcher Microstate(FCHSEQ0-4) and flags generated by the Fetcher Decoders (see Sheets 10&11) are input to the Decode Vector Control PROM(6T) which selects which of the Decode Vector PROMs(2P,1R,2T,2R&1U) will be enabled. The Decode Vector Control PROM also generates the enable for the Decode Vector Latch(ENBVECT).

ENBVECT0 will enable Decode Vector PROMs 2P & 1R on a single operand instruction when no hazards or indirects are detected. Opcode 1(FCH20-27) will determine the Decode Vector.

ENBVECT1 will enable Decode Vector PROM 2T when a hazard or indirect is detected. The Fetcher Microstate and flags from the Fetcher Decoders will determine the Decode Vector.

ENBVECT0 and the Multioperand Flag(MULTIOP) will enable Decode Vector PROM 2R. Opcode 2(FCH50-57) and the indirect address flags from Operand 2(FCH37) and Operand 3(FCH67) will determine the Decode Vector.

ENBVECT2 enables the Global Decode Vector buffer(1U) which determines the Decode Vector from the global interrupt flags.

When the Decode Vector Latch is enabled, the Decode Vector(VECT0-7) is valid at T1H or T3H. Table 2-6 lists the Decode Vectors output to the microcode branch logic on the Auxilliary Board.

SHEET 13

Clock signals used on the Memory Data Board are buffered at 1S. Timing Delay Circuit 4S generates T1.2, T1.6 and T1.8 clocks used in main memory timing. Timing Delay Circuit 5S generates a T0.6 clock also used in main memory timing.

With the exception of the Row Address Strobe(RAS), the low-active Memory Refresh enable(RFSH) prevents the generation of most memory enables and strobes. Clock holds due to memory mapping and microhalts which suspend microcycles will also delay the generation of memory enables and strobes(also with the exception RAS). Note that RFSH is generated at T0 every 15.2 microseconds(see Auxilliary Board Sheet 17).

Assuming the absence of hold or halt conditions, a memory request, or a refresh, a high data input will be clocked thru the RAS Register (the upper half of dual flip-flop 4T) at T1.2, generating high and low-active Row Address Strobes(RAS). The high-active RAS provides a partial enable for the gating of the Parity Error Address(see Sheet 9) and, in the absence of refresh(RAS and RFSH will both be high inputs to AND Gate 5W) will provide a high data input to the CAS Register (the lower half of dual flip-flop 7V). The high input will be clocked at T1.8 to generate the low-active Column Address Strobe(CAS). The low-active RAS and CAS are buffered at 1P and output through the backplane to main memory. RAS and CAS are deactivated at T0 by reset ot the upper half of 4T and the lower half of 7V.

The reset of the RAS Register pulls the high-active RAS low, which then resets the Row and Column Enable Register(the upper half of dual flip-flop 7V). Reset drives 7V-8 high, generating the low-active Row Address enable(ENROWADD) through NAND Gate 7T in the absence of refresh. Note that ENROWADD will be generated at T0.

The low signal to reset of the Row and Column Enable Register will go high when RAS is generated(at T1.2), and the high(pulled up) data input will be clocked at T1.6, driving 7V-9 high, generating the low-active Column Address enable (ENCOLADD). ENCOLADD and ENROWADD are output through the ribbon connector to the Memory Address Board.

The Memory Write Enable Register(the lower half of dual flip-flop 4T) generates active enables through preset of the flip-flop. When all of the inputs to NAND Gate 3V are high, a low will preset the flip-flop, generating high and low-active Memory Write enables(MWRITE & WR). NOTE: WR is not connected beyond the backplane of the Memory Address Board.

The low-active MWRITE disables the clocking of the Parity Error Address Register (Sheet 9), while the high-active MWRITE enables the Write Bit Shifter(Sheet 15) and provides the direction control for the bi-directional Memory Data Buffers (Sheet 6).

On a write to main memory, one of the memory write controls from the Microcode (WRTH or WRTL) will be a low input to NAND Gate 7T, generating two high inputs to NAND Gate 3V. In the absence of refresh, clock hold or microhalt conditions, all of the inputs to 3V will be high at T0.6. The Memory Write Enable Register will be preset and the enables generated. The low to preset will go high at the end of T0, and on the rising edge of the next T0, the low(tied down) data input to the flip-flop will be clocked, deactivating the enables.

Table 8-3 is a list of the time dependent main memory enables, listing the times they are generated. Note that all of the enables are reset at T0, except ENROWADD which is reset at T1.6 when ENCOLADD is generated.

TIME GENERATED	SIGNAL
T0	ENROWADD
T0.6	MWRITE
T1.2	RAS
T1.6	ENCOLLADD
T1.8	CAS

TABLE 8-3
Memory Enable Timing

As noted previously, the Fetcher cannot perform memory write operations. The microcode memory controls(WRTH or WRTL) disable the Fetcher during write operations by generating a high input to OR Gate 1X, preventing the generation of the Fetcher Strobe(FCHSTB).

WRTH or WTRL also prevent the generation of the low-active Memory Read Strobe (MRDSTB) during write operations with a high input to OR Gate 5U(5U-9). This prevents data contention on the bi-directional Memory Data Bus.

Because they are all dependent upon CAS, signals to load part or all of the Fetcher from main memory(FCHSTB, INDSTB, and MRDSTB) will be generated at T1.5.

The low-active Fetch Request(FETCHRQ) from the Microcode or Fetcher memory controls will generate FCHSTB(5U-11) which clocks the Fetcher Register and generates a memory request.

The generation of CAS in the absence of write controls from the Microcode will generate MRDSTB(5U-8) which clocks the Read Byte and Read Word 1 Registers, the Parity Error Latch, and part of the Parity Error Address Register.

When the Fetcher detects an indirect address in the decode of a macroinstruction, the Fetcher Memory Control Latch will output the low-active Fetcher Indirect Address Flag(FCHIND), which will generate the Indirect Strobe(INDSTB) which enables the loading of Operand 1 with the resolved indirect address(see Sheet 4).

The Logical Address Input MUX is shown on Sheets 13 and 14. The MUXes on Sheet 13 select the value to be loaded into LARA, while the MUXes on Sheet 14 select the value to be loaded into LARB or LARP. On all of the MUXes the A inputs are selected when the select input(pin 1) is low, and the B inputs are selected when the select input is high. The logical addresses from the MUX are output through the ribbon connector to the Memory Address Board.

The B inputs to the Fetcher LARA Input MUX(9S,9N,9R&9P) are preset to $000F. The B inputs to 9S are pulled up and the B inputs to 9N,9R,&9P are tied down. The A inputs are Operand 2(FCH30-47).

On a single operand instruction A SRC SEL from the Fetcher Memory Control Latch will be high and $000F will be loaded into LARA. On a double or triple operand instruction, Operand 2 will be loaded into LARA.

The LARA Input MUX(8S,9M,8N&8P) selects the address to be loaded into LARA from the Fetcher or the ALU. On a Fetcher load, SEQLDA from the Fetcher LAR Load Decoder(see Sheet 10) will be high and the address from the Fetcher LARA Input MUX will be selected. On an ALU load data from the Y Bus(Y00-17) will be selected.

SHEET 14

The Indirect Decrementer(9X,9H,8D&9A) is four cascaded 4-bit binary adders which add the A inputs to the B inputs. The B inputs are pulled up, causing $FFFF to be added to Operand 1(FCH00-07), decrementing the address by 1. Note that the most significant bit of Operand 1(FCH07) is not input to the Indirect Decrement-

er, but the corresponding input(9A-12) is pulled up. If Operand 1 is an indirect address, then FCH07(the indirect address flag) will be high and the result of the calculation would be the same. Note that Operand 1 will always be decremented; however, the result may not be used.

The A inputs to the Indirect/Direct(I/D) Address Input MUX(9W,9G,9D&8A) are Operand 1, while the B inputs are the decremented address from Operand 1. Note that the most significant input of the decremented address(8A-3) is tied down, effectively discarding the most significant bit of the logical address which is used only as the Indirect Address Flag.

The Indirect Address Flag(FCH07) is the select input to the MUX. The decremented address will be selected when FCH07 is high and Operand 1 will be selected when FCH07 is low. (Note since P Codes are not implemented, 8T-4 is pulled up and the status of 8T-6 will be the same as FCH07).

The LARBP Input MUX selects the address to be loaded into LARB or LARP from the Fetcher or the ALU. On a Fetcher Load, SEQLDP or SEQLDP from the Fetcher LAR Load Decoder(see Sheet 10) will be high and the resolved I/D Address will be selected. On an ALU load, data from the Y Bus will be selected.

Note that LARB and LARP share a common bus from the LARBP Input MUX through the Ribbon Connector to the Memory Address Board and the logical address registers. The Sequence Load enables(SEQLDB & SEQLDP) select the loading of the appropriate logical address register in addition to providing the select input to the MUX.

SHEET 15

As noted previously, the Fetch Register is also the Write Data Register. On a write to memory, Write Word 1(FCH00-17) is input to the Write MUX(9C&9J). The low byte(FCH00-07) is input to the A inputs of the MUX and the high byte(FCH10-17) is input to the B inputs to the MUX. During "normal" writes to memory, the low byte is written to the memory location specified by the address. The Write MUX allows the high byte to be written to that location.

High byte writes are controlled through the Microword, and are always single byte writes. When the Memory Control Field of the Microword specifies a high byte write, the high-active WRTLSEL will be generated by the Write Offset Decoder(see Sheet 16) and the high byte(FCH10-17) will be selected as the output from the Write MUX(WRMUX0-7). Otherwise the low byte will be selected as the outputs. The outputs from the Write MUX are input to the 0 address lines(pin 8) of the Write Bit Shifter.

Data to be written to main memory is aligned to the address of the starting byte within a memory word by the Write Bit Shifter(5A,5C,5D,5E,5H,5K,5L & 5N). The Byte Offset(MOFF0-2) defines the addressed byte within the 8-byte memory word and is defines degree of shift of the data from the Write Data Register. The Write Bit Shifters are enabled by the high-active Memory Write enable(MWRITE) generated by the Memory Write Enable Register.

SHEET 16

The Memory Control Field of the Microword(MSDAT32-37) is input to the Microcode Memory Control Decoder(7W,7X&6X) which generates enables for the address reg-

isters, the Fetcher, the Write MUX and other related logics. The most significant memory control bit(MSDAT37) is a select bit which enables 7W and 6X when low and 7X when high.

The Microcode Memory Control Latch(6W&6Y) is clocked at T0H. Output is disabled by a high signal from NAND Gate 3X(3X-8) if the Fetcher is running, a clock hold has been asserted, or a microhalt is asserted through local control. 5Y-12 is pulled up(effectively making 5Y an inverter) to generate the low-active enable for 7X when MSDAT is high.

On a write to main memory, a low-active write enable from decoder 7W or 7X(DWRH or DWRL) will generate the low-active Refresh Hold(RFHLD) through AND Gate 5U.

The output of Decoder 6X-1 will be low when a Fetch is to be started in the following microcycle. The low will be clocked thru latch 1Y at T0H and input to NOR Gate 7Y. The data input to the upper half of dual flip-flop 7U is tied down, and at T0H the inverted output(7U-8) will be driven high. The high input to NOR Gate 7Y prevents the high-active Fetcher Start Signal(START) from being generated. At T2H the upper half of flip-flop 7U is preset, pulling the inverted output low, at which time START will go high.

START is input to the Fetcher Anticipate Decoder(Sheet 11) and will be clocked through the Fetcher Sequence Latch(also on Sheet 11) at T0H to start the Fetcher.

The Memory Increment Field of the Microword(MSDAT30 & 31) is input to Microcode Memory Control Latch(1Y) and is output with an additional bit as the Increment Select Field(EINC0-2). The added bit(EINC2) will be low because the corresponding input is tied down. This bit identifies the increment as a microcode increment to the address register increment decoders on the Memory Address Board and the Write Offset Decoder(4R).

The Increment Select Field from the Microcode Memory Control Latch or from the Fetcher Memory Control Latch(see Sheet 10) is buffered at 5R and output through the ribbon connector to the Memory Address Board. It is also input to the Write Offset Decoder to determine Write Offset(WOFFSET0-2) which is buffered and output to the Parity Error MUX(see Sheet 9) and through the backplane to the Memory Mother Board(MEM64A).

SHEET 17

The Memory Data Board D Source Decoder(8W,8X&6Y) generates the enables for the data sources on the Memory Data Board. DA3 and DA4 select which of the decoders will be enabled.

The Memory Data Board Y Destination Decoder(4X) generates the enables for the data destinations on the Memory Data Board. YA3 and YA4 provide enables for the decoder which wil be enabled at T3H when selected.

Some local control enables are generated by the Local Control Register(4Y) and NOR Gate 7Y. The Local Control enable(LCL) is generated by the Auxilliary Board Y Destination Decode. Bit definitions are given in Section 2.2.8.5.

9.0 DIAG-R LOGICS

While it is not specifically part of the Q64 CPU, the Diagnostic Panel is a permanent part of the Q64 CPU assembly and is interactive with the CPU when diagnostic ROMs(Board Bug) are substituted for the Microstore ROMs on the ALU Board. To avoid confusion a distinction is made between the Diagnostic Panel and the Front Panel on the Q64. The term Front Panel refers to the switches and lights which are located on the face of the System 64 assembly. The Diagnostic Panel is located below the Front Panel, under a CE access panel.

SHEET 1

The IC locations on the board are shown in the block diagram in columns 7 & 8.

J1 is the ribbon connector to the AUX Board. The signal mnemonics, pin assignments and signal source or destination are shown.

J2 is the ribbon connector to the Front Panel switches. The signal mnemonics, pin assignments and signal source or destination are shown.

J5 and J6 are the connectors to the keyboard on the Diag-R Panel.

SHEET 2

Signals from the keyboard and switches are input to the 20 Key Encoder(2D). Note that START/STOP and IPL signals(shown at A8) from the Front Panel as well as the Diag Panel are input to 2D. The Key Encoder decodes the row and column inputs and outputs hexidecimal data to be displayed by the Input Hex Displays(4F&4E).

The Data Available(DAV) output(2D-13) goes to a high level when a valid keyboard entry is made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. DAV will return high to indicate acceptance of the new key after a normal debounce period. An internal register latches the last key input even after the key is released.

Table 9-1 is a truth table for the 20 Key Encoder.

SWITCH POSITION		0 Y1,X1	1 Y1,X2	2 Y1,X3	3 Y1,X4	4 Y2,X1	5 Y2,X2	6 Y2,X3	7 Y2,X4	8 Y3,X1	9 Y3,X2	10 Y3,X3	11 Y3,X4	12 Y4,X1	13 Y4,X2	14 Y4,X3	15 Y4,X4	16 Y5,X1	17 Y5,X2	18 Y5,X3	19 Y5,X4
DATA OUT	A	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	B	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	C	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
	D	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
	E	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

TABLE 9-1
20 Key Encoder
Truth Table

DAV is inverted(3D) and clocks flip-flops 2A(both halves) and 3B. If the data input is from the keypad, 2D-15(the E output) will be low, and the data input to the upper half of 2A will be low when it is clocked on the rising edge of the inverted DAV signal. The high 2A-8 output clocks the Hex Display Shifter(3C) which outputs the decoded data to the Input Hex Displays. The first data input is displayed by the Least Significant Input Display(4F) and looped back to the most significant inputs of the Hex Display Shifter. When a second key is depressed and released, the shifter will be clocked a second time and the first data input will be displayed in the Most Significant Input Display(4E). Input keypad data(FP00-07) is buffered at 2C and output to the Auxilliary Board through the ribbon connector.

When one of the switches on the Diagnostic Panel or the Front Panel is depressed and released, the Y5 input to the 20 Key Encoder will go high. The data input to the upper half of 2A will be high, generating a low 2A-8 output which will not clock 3C.

The low to high transition of the inverted DAV signal clocks flip-flop 3B. The inputs to 3B are the A, B & E outputs of the 20 Key Encoder, which encodes the 4 switch inputs in binary format. The encoded switch data is input to the Switch Decoder(2B).

The lower half of dual flip-flop 2A and both halves of dual flip-flop 1A generate the strobe enable for the Switch Signal Decoder, delaying the enable until the second T1 after DAV is true(high). The delay synchronizes the switch signals with the system clocks and provides for an orderly interrupt.

The low(tied down) data input of the lower half of flip-flop 2A(2A-2) is clocked on the rising edge of the inverted DAV. The low is then clocked through both halves of dual flip-flop 1A on the rising edge of the inverted T1 clock signal. Propogation delay through the lower half of 1A prevents the low from being clocked by the upper half until one full cycle after the lower half is clocked.

The low 1A-9 output presets the lower half of 2A, driving 2A-5 high, effectively resetting the low-active strobe enable two machine cycles(T1s) later. When 1A-9 goes high, 2A-4 will go high, and the lower half of 2A can be clocked.

The low 1A-9 also presets the upper half of 2A(2A-10), which holds the 2A-8 output high, effectively preventing the clocking of data input from the hex keypad through 3C until the front panel switch flags have been processed.

The low 1A-9 output is the strobe enable for the Switch Decoder(2B) which is set up as a 2 to 4 decoder. Table 9-2 is a function table for decoder 2B.

INPUTS				OUTPUTS			
SELECT		STROBE	DATA				
A	B	1G	1C	1Y0	1Y1	1Y2	1Y3
X	X	H	X	H	H	H	H
L	L	L	H	L	H	H	H
L	H	L	H	H	L	H	H
H	L	L	H	H	H	L	H
H	H	L	H	H	H	H	L
X	X	X	L	H	H	H	H

TABLE 9-2
2-line-to-4-line Decoder
Truth Table

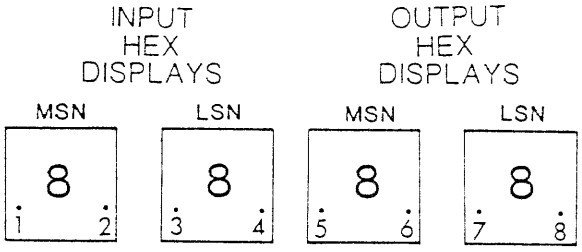
The Input Hex Displays and and Switch Signal Decoder are initialized by a power on reset sequence to ensure the circuits are in the proper state to accept keypad input. The Hex Display Shifter(3C) and the Switch Signal Latch(3B) are cleared by a power on reset through a buffer at 3A.

Input 3A-8 will be low until resistor R2 is charged(approximately 2.2 seconds). The low clears 3C and 3B and presets the upper half of flip-flop 1A, disabling the strobe of the Switch Signal Decoder(2B). When R2 is fully charged, 3A-8 will go high, and the affected devices(3C,3B&1A) may be clocked.

At T1 after the power on reset, output 1A-5 will be low(having been reset), and the low will be clocked by the upper half of 1A. The low 1A-9 output strobes the Switch Signal Decoder(2B) which decodes 0s since the Switch Signal Latch(3B) has been previously cleared. The low 1A-9 output also presets the upper half of 2A, pulling 2A-8 low. 1A-9 will return high(see above) to allow the clocking of the upper half of 2A when keypad data is input. The power on sequence ensures that 2A-8 will be low when keypad data is input, and that it will go high to clock the Hex Display Shifter when data is input.

Inverting buffer 3D and part of buffer 1C(shown at B2) input the switch signals and output complementary signals to the AUX Board(see AUX Board, Sheet 11). The STOP signal from the START/STOP Switch on the Front Panel or Diag Panel is input to a Dual Monostable Multivibrator(1B) which will output a low-active signal with a duration of approximately 1.6-1.8 seconds on the high to low transition of the 1B-1 input. A jumper should be in place between pins 2 and 4 on JMP1.

The other part of buffer 1C is used to buffer the TOH Clock and signals from the CPU that are displayed as decimal points on the Input Hex Displays. Figure 9-1 shows the four hex display units and the interpretation of the decimal points on the displays. The TOH Clock input to the Input Hex Displays causes the units to blank every 400NS, and the display is updated by the outputs of the Hex Display Shifter after each time they blank.



DECIMAL POINT	MEANING	NORMAL STATUS
1	CONT - Clocks are running	OFF
2	HALT - Clocks halted	OFF
3	MINT - Micro interrupt	ON
4	HELP - Macro interrupt	ON
5	Not used	OFF
6	Not used	OFF
7	PARR - Parity Error	ON
8	STOP - Executions stopped	OFF

If decimal point status is opposite of that shown, an error condition exists.

FIGURE 9-1
Diag Panel
Hex Displays

SHEET 3

Data output by the CPU to the Output Hex Displays(4H&4G) are buffered and input to the display units. The strobe enable for the display units(FPCNT) and the Front Panel Parity Error Flag(FPPARR) input to the left decimal point on the least significant display unit are buffered at 3A. The Output Hex Displays are not blanked(pin 8 in tied down) and the decimal points on the most significant device are not used.

MNEMONIC	DESCRIPTION
(A,B,P)PEND	MAP PENDING FLAG FOR ADDRESS REGISTER A, B, OR P
1/2CLK	50 NS CLOCK WITH PERIOD OF 100 NS
1/4CLK	25 NS CLOCK WITH PERIOD OF 50 NS
2 WAY DAT	REPLACEMENT LSB OF NEXT ADDRESS ON 2 WAY BRANCH
2WAY	2 WAY BRANCH FLAG
4WAY	4 WAY BRANCH FLAG
A SRC SEL	FETCH OR Y BUS SELECT SIGNAL
A+10-18	MAIN MEMORY ADDRESS PLUS ONE
A0-3	2901B INTERNAL REGISTER DATA
A0-8	MAIN MEMORY ADDRESS
ADD	ENABLE FOR BCD ADDITION
ALU	ENABLE TO SAVE FLAGS
AZ10	ACCUMULATED DECIMAL ZERO FLAG FROM BCD ADDER
AZ4	ACCUMULATED 4-BIT ZERO FLAG FROM ALU
AZ8	ACCUMULATED 8-BIT ZERO FLAG FROM ALU
B0-3	2901B INTERNAL REGISTER DATA
BASE	READBACK ENABLE FOR UPPER 16-BIT BASE REGISTER
BASE00-27	BASE REGISTER DATA
BASEL	READBACK ENABLE FOR LOWER 8-BIT BASE REGISTER
BCD1	LOAD ENABLE FOR BCD ADDER OPERAND 1
BCD2	LOAD ENABLE FOR BCD OPERAND 2(ENABLES FUNCTION)
BCDRES	READBACK ENABLE FOR BCD ADDER RESULT
BHAZL(A,B,P)	BASE HAZARD LOGICAL ADDRESS REGISTER A,B OR P
BNKHAZL(A,B,P)	BANK HAZARD FOR PHYSICAL ADDRESS REGISTER A,B OR P
BPTR	LOAD ENABLE FOR BASE POINTER
BRALL	256 WAY BRANCH ON CONTENTS OF BRANCH REGISTER
BRANCH NOT TAKEN	FETCHER BRANCH NOT TAKEN FLAG
BRANCH TAKEN	FETCHER BRANCH TAKEN FLAG
BRR	LOAD ENABLE FOR BRANCH REGISTER
BRRL	16 WAY BRANCH ON LOWER NIBBLE OF BRANCH REGISTER
BRRU	16 WAY BRANCH ON UPPER NIBBLE OF BRANCH REGISTER
BRSEL0-3	BASE REGISTER SELECT BITS(LOW NIBBLE BR ADDRESS)
BYPASS	ALU BYPASS ENABLE
C10	BCD CARRY FLAG
C10S	SAVED BCD CARRY FLAG
C12	ALU 12-BIT CARRY FLAG
C16	ALU 16-BIT CARRY FLAG
C16S	SAVED ALU 16-BIT CARRY FLAG
C4	ALU 4-BIT CARRY FLAG
C8	ALU 8-BIT CARRY FLAG
C8S	SAVED ALU 8-BIT CARRY FLAG
CALL	SUBROUTINE CALL FLAG-ENABLES WRITE TO STACK
CAS	COLUMN ADDRESS STROBE
CASCLK	COLUMN ADDRESS STROBE ENABLE
CC0,1	CARRY CONTROL FOR 2901B
CE	CHIP ENABLE
CODE1L	BRANCH ON LOWER NIBBLE CODE1 REGISTER
COLDS	COLDSTART FLAG
CONT	CONTINUE SIGNAL FROM FRONT PANEL(RESTARTS CLOCKS)
DA00-05	D SOURCE SELECT BITS
DATA SET READY	SERIAL PORT HANDSHAKE SIGNAL
DATA00-77	DATA FROM MAIN MEMORY
DATA0P-7P	PARITY BITS FROM MAIN MEMORY
DB00-17	D BUS DATA
DECODE	FETCH VECTOR BRANCH FLAG

MNEMONIC	DESCRIPTION
DFCH	MICROWORD DECODED FETCH REQUEST
DFRPNL	ENABLE FRONT PANEL DATA TO D BUS
DLCL	READBACK ENABLE FOR LOCAL CONTROL REGISTER
DOUBLE TRIPLE	MULTIPLE OPERAND FLAG
DWRH	MICROWORD WRITE HIGH BYTE ENABLE
DWRL	MICROWORD WRITE LOW BYTE ENABLE
EAR(A,B,P,R)	ENABLE ADDRESS REGISTER A,B,P OR RAR
EI/D	ENABLE INCREMENT/DECREMENT
EINC	ENABLE INCREMENT
EINC0-2	INCREMENT SELECT FIELD
EMPTY	EMPTY FLAG FROM MICRO STACK
ENBVECT	ONE OF 4 FETCH VECTOR ENABLES
ENBVECT0-2	ENABLES FOR FETCH VECTOR
ENCOLADD	ENABLES COLUMN ADDRESS ONTO MEMORY ADDRESS LINES
ENP	ONE OF TWO COUNT ENABLES FOR SEVERAL COUNTERS
ENROWADD	ENABLE ROW ADDRESS TO MEMORY ADDRESS LINES
F16	MOST SIGNIFICANT BIT OF ALU FUNCTION(SIGN BIT)
FCH00-77	DATA FROM FETCHER
FCHIND	INDIRECT ADDRESS COUNTER ENABLE
FCHSEQ0-4	FETCHER SEQUENCE LATCH DATA
FCHSTB	FETCHER STROBE
FDECODE	FETCH VECTOR DECODE ENABLE(FROM LOCAL CONTROL)
FETCH	FETCHER CONTINUE SIGNAL
FETCHRQ	FETCHER REQUEST FLAG
FLAGS	READBACK ENABLE FOR QANTEL MACRO FLAG REGISTER
FP00-07	FRONT PANEL OUTPUT DATA
FPCNT	ENABLES Y BUS DATA TO FRONT PANEL
FPCONT	FRONT PANEL CONTINUE SWITCH SIGNAL
FPCONT0/P	SIGNAL TO FRONT PANEL FROM CLOCK CIRCUIT ON CONT
FPHALT0/P	SIGNAL TO FRONT PANEL THAT HALT HAS BEEN CLOCKED
FPHALTSW	FRONT PANEL HALT SWITCH SIGNAL
FPHELP	FRONT PANEL MACRO INTERRUPT SIGNAL
FPI0-7	FRONT PANEL INPUT DATA
FPIPLN/O	FRONT PANEL IPL SIGNAL(NORMALLY OPEN)
FPIPLN/C	FRONT PANEL IPL SIGNAL(NORMALLY CLOSED)
FPMINT	FRONT PANEL MICROINTERRUPT INDICATOR SIGNAL
FPPARR	FRONT PANEL PARITY INDICATOR SIGNAL
FPSTOPSW	FRONT PANEL STOP SWITCH SIGNAL
FPT0H	FRONT PANEL T0H SIGNAL
FSEQANT0-4	ANTICIPATED FETCHER SEQUENCE DATA
FULL	FULL FLAG FROM MICRO STACK
G0-3	CARRY GENERATE SIGNALS BETWEEN 2901BS & CLA
GLINT	GLOBAL INTERRUPT BRANCH FLAG
GPC1	LOAD ENABLE FOR 16-BIT GENERAL PURPOSE COUNTER
GPC14	4-BIT ZERO FLAG FROM 16-BIT GPC
GPC2	LOAD ENABLE FOR 4-BIT GENERAL PURPOSE COUNTER
GPC24	4-BIT ZERO FLAG FROM 4-BIT GPC
GPCZ	16-BIT GPC ZERO FLAG
HAZ(A,B,P)L	BASE FAULT HAZARD FLAGS LOGICAL ADDRESS REGISTERS
HELP	AND OF MACRO-INTERRUPT ENABLE AND MACRO-INTERRUPT
HLT	MICRO HALT FLAG
HLTSW	HALT SWITCH SIGNAL
HOLD	CLOCK HOLD SIGNAL FOR MAPPING(STOPS H CLOCKS)
I/D00-17	RESOLVED INDIRECT OR DIRECT ADDRESS FROM FETCHER
I0-8	2801B CONTROLS

MNEMONIC	DESCRIPTION
INC	INCREMENT ENABLE
INC0-3	INCREMENT SELECT FIELD
INCINDCNT	INCREMENT INDIRECT COUNTER
INDOFL	INDIRECT COUNTER OVERFLOW FLAG
INDSTB	INDIRECT STROBE
INT	CONTROLLER INTERRUPT
IOE00-03	COMMAND LINES TO IOU CONTROLLERS
IOUWRT	WRITE ENABLE SIGNAL FOR IO DEVICES
IPLWARN	IPL WARNING SIGNAL
JUNK	ENABLE FOR JUNK REGISTER
LAR00-17	LOGICAL ADDRESS
LCAS	LEFT CASCADE ENABLE
LCL	LOCAL CONTROL ENABLE
LD(A,B,P)LATE	ENABLE LATE LOAD LAR(A,B OR P)
LD(B,P)BYP	LOAD ADDRESS B OR P THROUGH BYPASS
LIT	READBACK ENABLE LITERAL REGISTER
LITT	LOAD ENABLE LITERAL REGISTER
MACROINT	MACROINTERRUPT FLAG FROM IOU
MACROINTEN	ENABLE MACROINTERRUPT
MAP(A,B,P)	MAP ENABLE PHYSICAL ADDRESS REGISTER A,B OR P
MAPHLD	MAP HOLD SIGNAL
MDAT00-77	UNSHIFTED DATA FROM MEMORY/SHIFTED DATA TO MEMORY
MEMRQ	MEMORY REQUEST FLAG
MINT	MACRO INTERRUPT GENERATED BY ANY MICRO INTERRUPT
MOFF0-2	BYTE OFFSET
MOVLEN	READBACK ENABLE FOR MOVLEN REGISTER
MPBUSY	MAPPING BUSY FLAG
MRD00-77	SHIFTED READ DATA FROM MEMORY
MRDSTB	MEMORY READ STROBE
MSA00-07,10-13	MICROSTORE ADDRESS
MSBS0-3	MICROSTORE BANK SELECT BITS
MSBS4	WCS FLAG
MSDAT00-77	PIPELINE DATA
MULTIOP	MULTIPLE OPERAND FLAG
MUXSEL	ROM OR WCS ADDRESS SELECT FOR MICROSTORE ADDRESS
MWDIS	SIGNAL FROM TEST PANEL TO DISABLE MICROSTORE
MWRITE	MEMORY WRITE ENABLE
NA00-07	NEXT ADDRESS
NOFF0-2	UNBUFFERED BYTE OFFSET(MOFF)
OE	OUTPUT ENABLE
OPCD1,2	ENABLE CODE1 AND CODE2 REGISTERS
OVR	ALU OVERFLOW FLAG
P0-3	CARRY PROPOGATE SIGNALS BETWEEN 2901BS & CLA
PAR(A,B,P)00-27	PHYSICAL ADDRESS
PARR	PARITY ERROR FLAG
PERR	PARITY ERROR ADDRESS READBACK ENABLE
PERR0-7	PARITY BITS
PFL	POWER FAILURE SIGNAL
PLAR(A,B,P)	COUNT ENABLE FOR LAR(A,B,P) AND PAR(A,B,P)
PMODE	PMODE ENABLE(NOT IMPLEMENTED)
POP	ENABLE DECREMENT STACK POINTER
PWRFAIL	WARNING SIGNAL TO CPU OF UPCOMING POWER FAILURE
RA+1-03 -RA+1-22	"REAL" A+1 ADDRESS
RA00-17	"REAL" ADDRESS OUTPUT BY BASE ADDERS
RAS	ROW ADDRESS STROBE

MNEMONIC	DESCRIPTION
RBYTE	READ BYTE REGISTER ENABLE
RCAS	RIGHT CASCADE ENABLE
RDWCS0-3	WCS READ DATA WORD0-3
RECIEVE DATA	UART INPUT SIGNAL
REF	REFRESH ENABLE
RET	ENABLE ADDRESS OFF MICRO STACK ON RETURN
RFA0-7	REFRESH ADDRESS FOR MAIN MEMORY(ROW ONLY)
RFHLD	REFRESH HOLD FLAG
RFSH	REFRESH ENABLE
RFSHLD	REFRESH HOLD FLAG
RMA00-27	"REAL" MEMORY ADDRESS FROM PAR(x)
RMAP(A,B,P)	REMAP SIGNAL FOR ADDRESS REGISTER A,B OR P
ROMEN	MICRO STORE ROM BUFFER ENABLE
RST1-4	IPL RESET SIGNALS
RSTINDCNT	RESET INDIRECT COUNTER
RWORD1-4	READ WORD ENABLES
SC0,1	MICROWORD SHIFT CONTROL BITS TO ALU
SEQLD(A,B,P)	SEQUENCE LOAD FOR ADDRESS REGISTER A,B OR P
SINGLE OP	SINGLE OPERAND FLAG
SPEC	SPECIAL(16-256 WAY) BRANCH FLAG
SR	SAVE REGISTER DATA(FROM ALU SHIFT MUX)
SREQ	SERVICE REQUEST FLAG FOR IO DEVICES
SS	FETCHER SINGLE STEP FLAG
ST00-17 & 20	MICROSTORE ADDRESS READ FROM STACK
START	START FETCHER SIGNAL
STARTLIT	ENABLE FOR FRONT PANEL START LIGHT
STBPEND(A,B,P)	STROBE FOR LOAD PENDING FOR ADD REG A,B OR P
STOPLITE	ENABLE FOR FRONT PANEL STOP LIGHT
STOPSW	SOFTWARE STOP SIGNAL FROM FRONT PANEL
STROBE	I/O DEVICE STROBE
SUB	BCD SUBTRACTION ENABLE
SWEXT	ROM SIMULATOR ENABLE(DISABLES MICROSTORE)
T0,T1,T2,T3	100NS CLOCK PULSES WITH CYCLE OF 400NS
T0H,T1H,T2H,T3H	SAME AS T0,T1,ETC. CAN BE HELD FOR REFRESH, ETC
TEMP0-7	TEMP REGISTER DATA
THREE	ENABLE CONSTANT THREE TO Y BUS
TIME OUT	INTERRUPT SIGNAL GENERATED BY PIT
TPT0H	TEST PANEL T0H CLOCK SIGNAL
UARTCLK	ENABLE FOR CLOCKING UART OUTPUT
UARTINT	UART INTERRUPT FLAG
UCTL	UART CONTROL REGISTER ENABLE
UDATA	UART DATA ENABLE
VCC/CNTL	VOLTAGE CONTROL
VECT0-7	FETCH VECTOR
WCS11/WE	WCS WRITE ENABLE
WCSA	WCS ADDRESS REGISTER ENABLE
WCSBUFEN	WCS BUFFER ENABLE
WCSCE	WCS CHIP ENABLE
WCSD	WCS WRITE DATA REGISTER ENABLE
WCSD1-4	WCS WORD ENABLE
WCSDSEL0-4	BANK SELECT FOR WCS
WCSERR	WCS PARITY ERROR FLAG(NOT IMPLEMENTED)
WCSRDSTR	WCS READ STROBE
WDSAB	ENABLE FOR READ BIT SHIFTERS(WRITE DISABLE)
WE0-17	WRITE ENABLE SIGNAL FOR ROW DESIGNATED BY DIGIT

MNEMONIC	DESCRIPTION
WOFF0-2	WORD OFFSET(RANGE) FOR MEMORY WRITE(NO. OF BYTES)
WR	WRITE ENABLE
WR/W00-07	WCS WRITE ADDRESS
WRMUX0-7	ENABLES FOR WRITE BIT SHIFTERS
WRTH	HIGH BYTE WRITE ENABLE
WRTL	LOW BYTE WRITE ENABLE
WRTSEL	WRITE SELECT FOR HIGH OR LOW BYTE
WWORD1-4	WRITE WORD ENABLE
XCLK	EXTERNAL CLOCK ENABLE FOR QA
XMIT	UART TRANSMIT DATA SIGNAL
Y00-17	Y BUS DATA
YA0-5	SELECT FIELD FOR Y DESTINATION
YBASE	BASE REGISTER LOAD ENABLE
YBASEA	BASE REGISTER ADDRESS ENABLE
YBRALL	256 WAY BRANCH ON LOW BYTE ON Y BUS
YBRRL	16 WAY BRANCH ON LEAST SIGNIFICANT NIBBLE ON Y BUS
YBRRU	16 WAY BRANCH HIGH NIBBLE OF LOW BYTE ON Y BUS
YFLAGS	LOAD ENABLE FOR QANTEL FLAG REGISTER
YFRPNL	DATA ENABLE FROM Y BUS TO FRONT PANEL
YIODATA	ENABLE DATA FROM Y BUS TO IO DEVICES
YLARA00-17	LOGICAL ADDRESS FROM INPUT MUX TO LARA
YLARBP00-17	LOGICAL ADDRESS FROM INPUT MUX TO LARB OR LARP
YRAR	LOAD ENABLE FOR REAL ADDRESS REGISTER
YSWAP	SWAP REGISTER ENABLE
YTIMER	PROGRAMMABLE INTERVAL TIMER LOAD ENABLE
YUDATA	ENABLE Y BUS DATA TO UART
Z16	ALU 16-BIT ZERO FLAG
Z4H	ALU 4-BIT ZERO FROM SECOND LS NIBBLE
Z8	ALU 8-BIT ZERO FLAG(LEAST SIGNIFICANT BYTE)
Z8H	ALU 8 BIT ZERO FLAG(MOST SIGNIFICANT BYTE)