Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	CPU32
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	16MHz
Co-Processors/DSP	-
RAM Controllers	DRAM
Graphics Acceleration	No
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5.0V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	144-BQFP
Supplier Device Package	144-QFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68340ab16e

LIST OF ILLUSTRATIONS

2.8.4 Read-Modify-Write Cycle (RMC)

This output signal identifies the bus cycle as part of an indivisible read-modify-write operation. It remains asserted during all bus cycles of the read-modify-write operation to indicate that bus ownership cannot be transferred.

2.9 EXCEPTION CONTROL SIGNALS

These signals are used by the MC68340 to recover from an exception.

2.9.1 Reset (RESET)

This active-low, open-drain, bidirectional signal is used to initiate a system reset. An external reset signal (as well as a reset from the SIM40) resets the MC68340 and all external devices. A reset signal from the CPU32 (asserted as part of the RESET instruction) resets external devices; the internal state of the CPU32 is not affected. The on-chip modules are reset, except for the SIM40. However, the module configuration register for each on-chip module is not altered. When asserted by the MC68340, this signal is guaranteed to be asserted for a minimum of 512 clock cycles. Refer to **Section 3 Bus Operation** for a description of bus reset operation and **Section 5 CPU32** for information about the reset exception.

2.9.2 Halt (HALT)

This active-low, open-drain, bidirectional signal is asserted to suspend external bus activity, to request a retry when used with BERR, or to perform a single-step operation. As an output, HALT indicates a double bus fault by the CPU32. Refer to **Section 3 Bus Operation** for a description of the effects of HALT on bus operation.

2.9.3 Bus Error (BERR)

This active-low input signal indicates that an invalid bus operation is being attempted or, when used with HALT, that the processor should retry the current cycle. Refer to **Section 3 Bus Operation** for a description of the effects of BERR on bus operation.

2.10 CLOCK SIGNALS

These signals are used by the MC68340 for controlling or generating the system clocks. See **Section 4 System Integration Module** for more information on the various clocking methods and frequencies.

2.10.1 System Clock (CLKOUT)

This output signal is the system clock output and is used as the bus timing reference by external devices. CLKOUT can be varied in frequency or slowed in low power stop mode to conserve power.

Table 2-5. Signal Summary

Signal Name	Mnemonic	Input/Output	Active State	Three-State
Address Bus	A23–A0	Out	—	Yes
Address Bus Port A7–A0/ Interrupt Acknowledge	A31–A24	Out/I/O/Out	—/—/Low	Yes
Data Bus	D15–D0	I/O	—	Yes
Function Codes	FC3–FC0	Out	—	Yes
Chip Select 3/Interrupt Request Level/Port B4, B2, B1	CS3–CS1	Out/In/I/O	Low/Low/—	No
Chip Select 0/Autovector	CS0	Out/In	Low/Low	No
Bus Request	BR	In	Low	—
Bus Grant	BG	Out	Low	No
Bus Grant Acknowledge	BGACK	In	Low	—
Data and Size Acknowledge	DSACK1, DSACK0	In	Low	—
Read-Modify-Write Cycle	RMC	Out	Low	Yes
Address Strobe	AS	Out	Low	Yes
Data Strobe	DS	Out	Low	Yes
Size	SIZ1, SIZ0	Out	—	Yes
Read/Write	R/W	Out	High/Low	Yes
Interrupt Request Level/ Port B7, B6, B5, B3	IRQ7, IRQ6, IRQ5, IRQ3	In/I/O	Low/—	—
Reset	RESET	I/O	Low	No
Halt	HALT	I/O	Low	No
Bus Error	BERR	In	Low	—
System Clock	CLKOUT	Out	—	No
Crystal Oscillator	EXTAL, XTAL	In, Out	—	—
External Filter Capacitor	XFC	In	—	—
Clock Mode Select/Port B0	MODCK	In/I/O	—/—	—
Instruction Fetch/ Development Serial In	IFETCH/DSI	Out/In	Low/—	No/—
Instruction Pipe/ Development Serial Out	IPIPE/DSO	Out/Out	Low/—	No/—
Breakpoint/ Development Serial Clock	BKPT/DSCLK	In/In	Low/—	—/—
Freeze	FREEZE	Out	High	No
Receive Data	RxDA, RxDB	In	—	—

State 0—The read cycle starts in state 0 (S0). During S0, the MC68340 places a valid address on A31–A0 and valid function codes on FC3–FC0. The function codes select the address space for the cycle. The MC68340 drives R/W high for a read cycle. SIZ1/SIZ0 become valid, indicating the number of bytes requested for transfer.

State 1—One-half clock later, in state 1 (S1), the MC68340 asserts AS indicating a valid address on the address bus. The MC68340 also asserts DS during S1. The selected device uses R/W, SIZ1 or SIZ0, A0, and DS to place its information on the data bus. One or both of the bytes (D15–D8 and D7–D0) are selected by SIZ1/SIZ0 and A0.

State 2—As long as at least one of the DSACK≈ signals is recognized on the falling edge of S2 (meeting the asynchronous input setup time requirement), data is latched on the falling edge of S4, and the cycle terminates.

State 3—If DSACK≈ is not recognized by the start of state 3 (S3), the MC68340 inserts wait states instead of proceeding to states 4 and 5. To ensure that wait states are inserted, both DSACK1 and DSACK0 must remain negated throughout the asynchronous input setup and hold times around the end of S2. If wait states are added, the MC68340 continues to sample DSACK≈ on the falling edges of the clock until one is recognized.

State 4—At the falling edge of state 4 (S4), the MC68340 latches the incoming data and samples DSACK≈ to get the port size.

State 5—The MC68340 negates AS and DS during state 5 (S5). It holds the address valid during S5 to provide address hold time for memory systems. R/W, SIZ1 and SIZ0, and FC3–FC0 also remain valid throughout S5. The external device keeps its data and DSACK≈ signals asserted until it detects the negation of AS or DS (whichever it detects first). The device must remove its data and negate DSACK≈ within approximately one clock period after sensing the negation of AS or DS. DSACK≈ signals that remain asserted beyond this limit may be prematurely detected for the next bus cycle.

3.4.1 Breakpoint Acknowledge Cycle

The breakpoint acknowledge cycle allows external hardware to insert an instruction directly into the instruction pipeline as the program executes. The breakpoint acknowledge cycle is generated by the execution of a breakpoint instruction (BKPT) or the assertion of the BKPT pin. The T-bit state (shown in Figure 3-10) differentiates a software breakpoint cycle (T = 0) from a hardware breakpoint cycle (T = 1).

When a BKPT instruction is executed (software breakpoint), the MC68340 performs a word read from CPU space, type 0, at an address corresponding to the breakpoint number (bits [2–0] of the BKPT opcode) on A4–A2, and the T-bit (A1) is cleared. If this bus cycle is terminated with BERR (i.e., no instruction word is available), the MC68340 then performs illegal instruction exception processing. If the bus cycle is terminated by DSACK≈, the MC68340 uses the data on D15–D0 (for 16-bit ports) or two reads from D15–D8 (for 8-bit ports) to replace the BKPT instruction in the internal instruction pipeline and then begins execution of that instruction.

When the CPU32 acknowledges a BKPT pin assertion (hardware breakpoint) with background mode disabled, the CPU32 performs a word read from CPU space, type 0, at an address corresponding to all ones on A4–A2 (BKPT#7), and the T-bit (A1) is set. If this bus cycle is terminated by BERR, the MC68340 performs hardware breakpoint exception processing. If this bus cycle is terminated by DSACK≈, the MC68340 ignores data on the data bus and continues execution of the next instruction.

NOTE

The BKPT pin is sampled on the same clock phase as data and is latched with data as it enters the CPU32 pipeline. If BKPT is asserted for only one bus cycle and a pipeline flush occurs before BKPT is detected by the CPU32, BKPT is ignored. To ensure detection of BKPT by the CPU32, BKPT can be asserted until a breakpoint acknowledge cycle is recognized.

The breakpoint operation flowchart is shown in Figure 3-11. Figures 3-12 and 3-13 show the timing diagrams for the breakpoint acknowledge cycle with instruction opcodes supplied on the cycle and with an exception signaled, respectively.

interrupt (as programmed by the SWRI bit in the SYPCR). The address of the interrupt service routine for the software watchdog interrupt is stored in the software interrupt vector register (SWIV). Figure 4-3 shows a block diagram of the software watchdog as well as the clock control circuits for the periodic interrupt timer.

The watchdog clock rate is determined by the SWP bit in the periodic interrupt timer register (PITR) and the SWT bits in the SYPCR. See Table 4-7 for a list of watchdog timeout periods.

The software watchdog service sequence consists of the following steps: 1) write $55 to the software service register (SWSR) and 2) write $AA to the SWSR. Both writes must occur in the order listed prior to the watchdog timeout, but any number of instructions or accesses to the SWSR can be executed between the two writes.

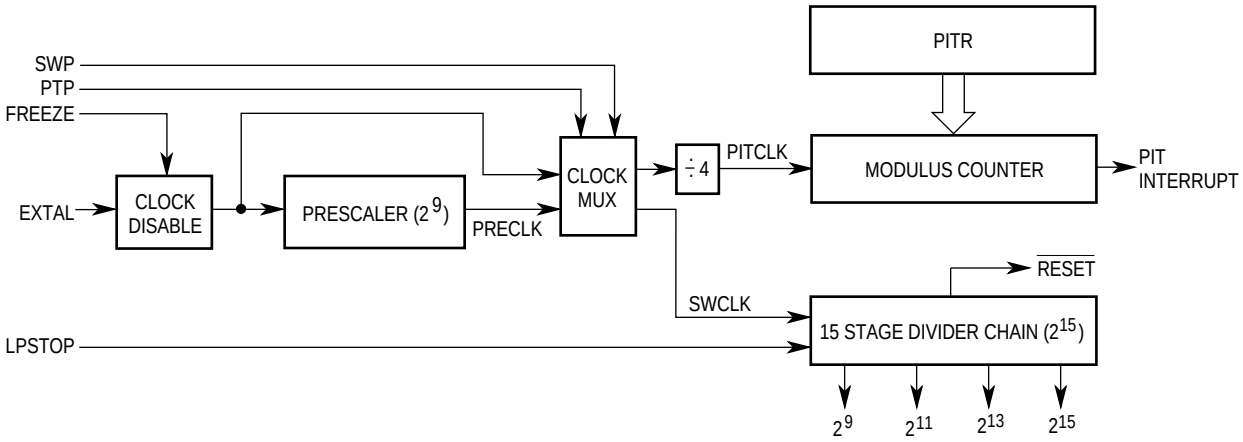


Figure 4-3. Software Watchdog Block Diagram

4.2.2.6 PERIODIC INTERRUPT TIMER. The periodic interrupt timer consists of an 8-bit modulus counter that is loaded with the value contained in the PITR (see Figure 4-3). The modulus counter is clocked by a signal derived from the EXTAL input pin unless an external frequency source is used. When an external frequency source is used (MODCK low during reset), the default state of the prescaler control bits (SWP and PTP) in the PITR is changed to enable both prescalers.

Either clock source (EXTAL or EXTAL ÷ 512) is divided by 4 before driving the modulus counter (PITCLK). When the modulus counter value reaches zero, an interrupt is generated. The level of the generated interrupt is programmed into the PIRQL bits in the periodic interrupt control register (PICR). During the IACK cycle, the SIM40 places the periodic interrupt vector, programmed into the PIV bits in the PICR, onto the internal bus. The value of bits 7–0 in the PITR is then loaded again into the modulus counter, and the counting process starts over. If a new value is written to the PITR, this value is loaded into the modulus counter when the current count is completed.

The number of wait states programmed into the internal wait state generation logic by a chip select can be used even though the pin is not used as a CS≈ signal. The programmed number of wait states in the CS≈ signal applies to the port B pins configured as IRQ≈ or I/O pins. This is done by programming the chip select with the number of wait states to be added, as though it were to be used. The DD1/DD0 and PS1/PS0 bits in the chip select address mask register must be set to add the desired number of wait states (the V-bit in the module base address register should be set).

4.2.6 Low-Power Stop

Executing the LPSTOP instruction provides reduced power consumption when the MC68340 is idle; only the SIM40 remains active. Operation of the SIM40 clock and CLKOUT during LPSTOP is controlled by the STSIM and STEXT bits in the SYNCR (see Table 4-3). LPSTOP disables the clock to the software watchdog in the low state. The software watchdog remains stopped until the LPSTOP mode ends; it begins to run again on the next rising clock edge.

NOTE

When the CPU32 executes the STOP instruction (as opposed to LPSTOP), the software watchdog continues to run. If the software watchdog is enabled, it issues a reset or interrupt when timeout occurs.

The periodic interrupt timer does not respond to an LPSTOP instruction; thus, it can be used to exit LPSTOP as long as the interrupt request level is higher than the CPU32 interrupt mask level. To stop the periodic interrupt timer while in LPSTOP, the PITR must be loaded with a zero value before LPSTOP is executed. The bus monitor, double bus fault monitor, and spurious interrupt monitor are all inactive during LPSTOP.

The STP bit in the MCR of each on-chip module (DMA, timers, and serial modules) should be set prior to executing the LPSTOP instruction. Setting the STP bit stops all clocks within each of the modules, except for the clock from the IMB. The clock from the IMB remains active to allow the CPU32 access to the MCR of each module. The system clock stops on the low phase of the clock and remains stopped until the STP bit is cleared by the CPU32 or until reset. For more information, see the description of the MCR STP bit for each module.

If an external device requires additional time to prepare for entry into LPSTOP mode, entry can be delayed by asserting HALT (see **3.4.2 LPSTOP Broadcast Cycle**).

4.2.7 Freeze

FREEZE is asserted by the CPU32 if a breakpoint is encountered with background mode enabled. Refer to **Section 5 CPU32** for more information on the background mode. When FREEZE is asserted, the double bus fault monitor and spurious interrupt monitor continue to operate normally. However, the software watchdog, the periodic interrupt timer and the internal bus monitor will be affected. When FREEZE is asserted, setting the FRZ1 bit in

4.3.2.8 SOFTWARE SERVICE REGISTER (SWSR). The SWSR is the location to which the software watchdog servicing sequence is written. The software watchdog can be enabled or disabled by the SWE bit in the SYPCR. SWSR can be written at any time, but returns all zeros when read.

SWSR $027

7	6	5	4	3	2	1	0
SWSR7	SWSR6	SWSR5	SWSR4	SWSR3	SWSR2	SWSR1	SWSR0
RESET:							
0	0	0	0	0	0	0	0

Supervisor Only

4.3.3 Clock Synthesizer Control Register (SYNCR)

The SYNCR can be read or written only in supervisor mode. The reset state of SYNCR produces an operating frequency of 8.39 MHz when the PLL is referenced to a 32.768-kHz reference signal. The system frequency is controlled by the frequency control bits in the upper byte of the SYNCR as follows:

$$F_{SYSTEM} = F_{CRYSTAL}\,[2^{(2+2W+X)}] \times (Y+1)$$

SYNCR $004

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W	X	Y5	Y4	Y3	Y2	Y1	Y0	RSVD	0	0	SLIMP	SLOCK	RSTEN	STSIM	STEXT
RESET:															
0	0	1	1	1	1	1	1	0	0	0	U	U	0	0	0

U = Unaffected by reset

Supervisor Only

W—Frequency Control Bit

This bit controls the prescaler tap in the synthesizer feedback loop. Setting the bit increases the VCO speed by a factor of 4, requiring a time delay for the VCO to relock (see equation for determining system frequency).

X—Frequency Control Bit

This bit controls a divide-by-two prescaler, which is not in the synthesizer feedback loop. Setting the bit doubles the system clock speed without changing the VCO speed, as specified in the equation for determining system frequency; therefore, no delay is incurred to relock the VCO.

Y5–Y0—Frequency Control Bits

The Y-bits, with a value from 0–63, control the modulus downcounter in the synthesizer feedback loop, causing it to divide by the value of Y+1 (see the equation for determining system frequency). Changing these bits requires a time delay for the VCO to relock.

Bits 7–5—Reserved

Bit 7 is reserved for factory testing.

5.1.1 Features

Features of the CPU32 are as follows:

- Fully Upward Object-Code Compatible with M68000 Family
- Virtual Memory Implementation
- Loop Mode of Instruction Execution
- Fast Multiply, Divide, and Shift Instructions
- Fast Bus Interface with Dynamic Bus Port Sizing
- Improved Exception Handling for Embedded Control Applications
- Additional Addressing Modes
  - — Scaled Index
  - — Address Register Indirect with Base Displacement and Index
  - — Expanded PC Relative Modes
  - — 32-Bit Branch Displacements
- Instruction Set Additions
  - — High-Precision Multiply and Divide
  - — Trap On Condition Codes
  - — Upper and Lower Bounds Checking
- Enhanced Breakpoint Instruction
- Trace on Change of Flow
- Table Lookup and Interpolate Instruction
- LPSTOP Instruction
- Hardware BKPT Signal, Background Mode
- Fully Static Implementation

A block diagram of the CPU32 is shown in Figure 5-1. The major blocks depicted operate in a highly independent fashion that maximizes concurrences of operation while managing the essential synchronization of instruction execution and bus operation. The bus controller loads instructions from the data bus into the decode unit. The sequencer and control unit provide overall chip control, managing the internal buses, registers, and functions of the execution unit.

5.1.2 Virtual Memory

A system that supports virtual memory has a limited amount of high-speed physical memory that can be accessed directly by the processor and maintains an image of a much larger virtual memory on a secondary storage device. When the processor attempts to access a location in the virtual memory map that is not resident in physical memory, a page fault occurs. The access to that location is temporarily suspended while the necessary data is fetched from secondary storage and placed in physical memory. The

CPU32 uses instruction restart, which requires that only a small portion of the internal machine state be saved. After correcting the page fault, the machine state is restored, and the instruction is refetched and restarted. This process is completely transparent to the application program.

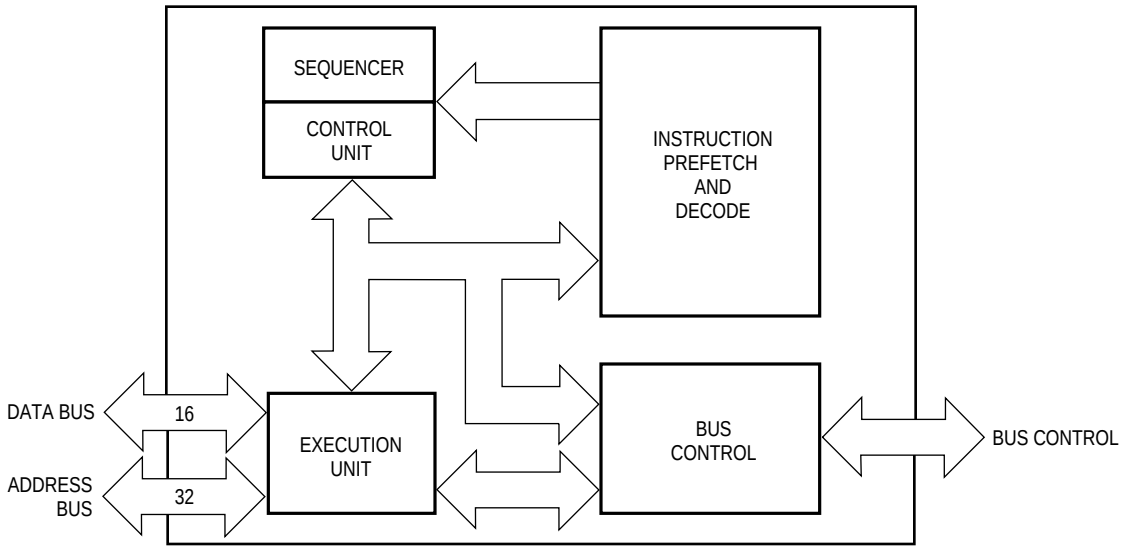


Figure 5-1. CPU32 Block Diagram

5.1.3 Loop Mode Instruction Execution

The CPU32 has several features that provide efficient execution of program loops. One of these features is the DBcc looping primitive instruction. To increase the performance of the CPU32, a loop mode has been added to the processor. The loop mode is used by any single-word instruction that does not change the program flow. Loop mode is implemented in conjunction with the DBcc instruction. Figure 5-2 shows the required form of an instruction loop for the processor to enter loop mode.

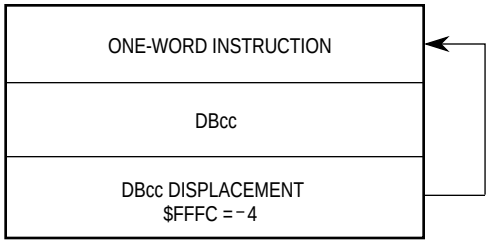


Figure 5-2. Loop Mode Instruction Sequence

The loop mode is entered when the DBcc instruction is executed and the loop displacement is –4. Once in loop mode, the processor performs only the data cycles associated with the instruction and suppresses all instruction fetches. The termination

5.3.3.7 BINARY-CODED DECIMAL (BCD) INSTRUCTIONS. Five instructions support operations on BCD numbers. The arithmetic operations on packed BCD numbers are add decimal with extend (ABCD), subtract decimal with extend (SBCD), and negate decimal with extend (NBCD). Table 5-9 is a summary of the BCD operations.

Table 5-9. Binary-Coded Decimal Operations

Instruction	Operand Syntax	Operand Size	Operation
ABCD	Dn, Dn – (An), – (An)	8 8	$Source_{10} + Destination_{10} + X \Rightarrow Destination$
NBCD	⟨ea⟩	8 8	$0 - Destination_{10} - X \Rightarrow Destination$
SBCD	Dn, Dn – (An), – (An)	8 8	$Destination_{10} - Source_{10} - X \Rightarrow Destination$

5.3.3.8 PROGRAM CONTROL INSTRUCTIONS. A set of subroutine call and return instructions and conditional and unconditional branch instructions perform program control operations. Table 5-10 summarizes these instructions.

Table 5-10. Program Control Operations

Instruction	Operand Syntax	Operand Size	Operation
Conditional			
Bcc	⟨label⟩	8, 16, 32	If condition true, then PC + d ⇒ PC
DBcc	Dn, ⟨label⟩	16	If condition false, then Dn – 1 ⇒ PC; if Dn ≠ (– 1), then PC + d ⇒ PC
Scc	⟨ea⟩	8	If condition true, then destination bits are set to 1; else destination bits are cleared to 0
Unconditional			
BRA	⟨label⟩	8, 16, 32	PC + d ⇒ PC
BSR	⟨label⟩	8, 16, 32	SP – 4 ⇒ SP; PC ⇒ (SP); PC + d ⇒ PC
JMP	⟨ea⟩	none	Destination ⇒ PC
JSR	⟨ea⟩	none	SP – 4 ⇒ SP; PC ⇒ (SP); destination ⇒ PC
NOP	none	none	PC + 2 ⇒ PC
Returns			
RTD	#⟨d⟩	16	(SP) ⇒ PC; SP + 4 + d ⇒ SP
RTR	none	none	(SP) ⇒ CCR; SP + 2 ⇒ SP; (SP) ⇒ PC; SP + 4 ⇒ SP
RTS	none	none	(SP) ⇒ PC; SP + 4 ⇒ SP

The table instruction is executed with the following bit pattern in Dx:

31 … 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NOT USED	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0

Table Entry Offset ⇒ Dx [8:15] = $A3 = 163

Interpolation Fraction ⇒ Dx [0:7] = $80 = 128

Using this information, the table instruction calculates dependent variable Y:

Y = 1669 + (128 (1679 – 1669)) / 256 = 1674

5.3.4.2 TABLE EXAMPLE 2: COMPRESSED TABLE. In Example 2 (see Figure 5-8), the data from Example 1 has been compressed by limiting the maximum value of the independent variable. Instead of the range 0 ≤ X = 65535, X is limited to 0 ≤ X ≤ 1023. The table has been compressed to only five entries, but up to 256 levels of interpolation are allowed between entries.

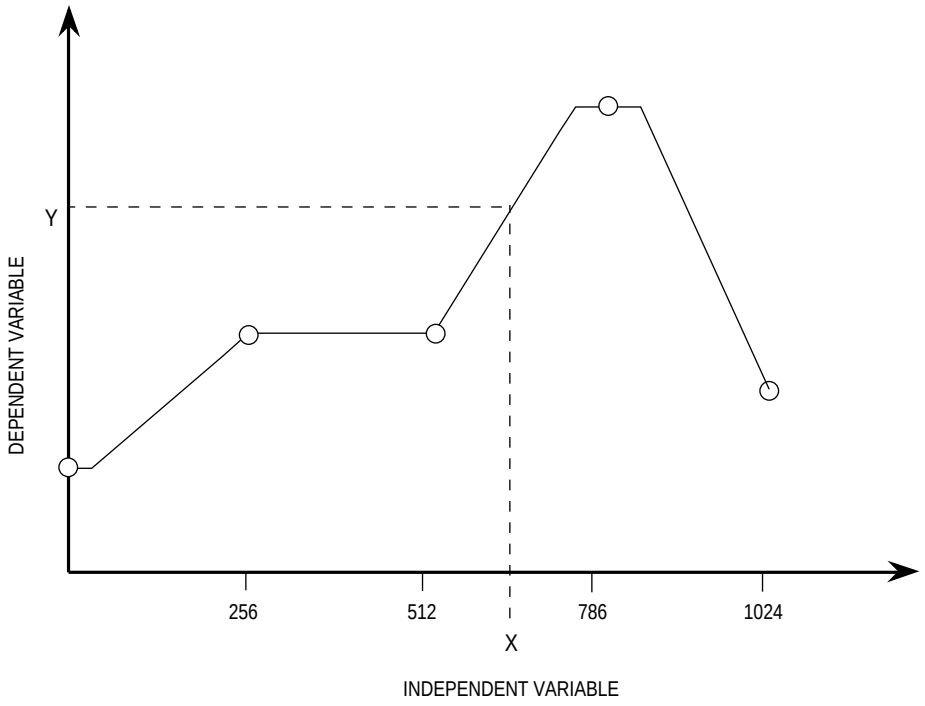


Figure 5-8. Table Example 2

NOTE

Extreme table compression with many levels of interpolation is possible only with highly linear functions. The table entries within the range of interest are listed in Table 5-14.

5.5.2 Processing of Specific Exceptions

The following paragraphs provide details concerning sources of specific exceptions, how each arises, and how each is processed.

5.5.2.1 RESET. Assertion of RESET by external hardware or assertion of the internal RESET signal by an internal module causes a reset exception. The reset exception has the highest priority of any exception. Reset is used for system initialization and for recovery from catastrophic failure. The reset exception aborts any processing in progress when it is recognized, and that processing cannot be recovered. Reset performs the following operations:

1. Clears T0 and T1 in the SR to disable tracing
2. Sets the S-bit in the SR to establish supervisor privilege
3. Sets the interrupt priority mask to the highest priority level (%111)
4. Initializes the VBR to zero ($00000000)
5. Generates a vector number to reference the reset exception vector
6. Loads the first long word of the vector into the interrupt SP
7. Loads the second long word of the vector into the PC
8. Fetches and initiates decode of the first instruction to be executed

Figure 5-11 is a flowchart of the reset exception

After initial instruction prefetches, normal program execution begins at the address in the PC. The reset exception does not save the value of either the PC or the SR.

If a bus error or address error occurs during reset exception processing sequence, a double bus fault occurs, the processor halts, and the HALT signal is asserted to indicate the halted condition.

Execution of the RESET instruction does not cause a reset exception nor does it affect any internal CPU register. The SIM40 registers and the MCR in each internal peripheral module (DMA, timers, and serial modules) are not affected. All other internal peripheral module registers are reset the same as for a hardware reset. The external devices connected to the RESET signal are reset at the completion of the RESET instruction.

DSCLK, the gated serial clock, is normally high, but it pulses low for each bit to be transferred. At the end of the seventeenth clock period, it remains high until the start of the next transmission. Clock frequency is implementation dependent and may range from DC to the maximum specified frequency. Although performance considerations might dictate a hardware implementation, software solutions can be used provided serial bus timing is maintained.

5.6.2.8 COMMAND SET. The following paragraphs describe the command set available in BDM.

5.6.2.8.1 Command Format. The following standard bit format is utilized by all BDM commands.

15 — 10	9	8	7 — 6	5	4	3	2 — 0
OPERATION	0	R/W	OP SIZE	0	0	A/D	REGISTER
EXTENSION WORD(S)							

Bits 15–0—Operation Field

The operation field specifies the commands. This 6-bit field provides for a maximum of 64 unique commands.

R/W Field

The R/W field specifies the direction of operand transfer. When the bit is set, the transfer is from CPU to development system. When the bit is cleared, data is written to the CPU or to memory from the development system.

Operand Size

For sized operations, this field specifies the operand data size. All addresses are expressed as 32-bit absolute values. The size field is encoded as listed in Table 5-22.

Table 5-22. Size Field Encoding

Encoding	Operand Size
00	Byte
01	Word
10	Long
11	Reserved

Address/Data (A/D) Field

The A/D field is used by commands that operate on address and data registers. It determines whether the register field specifies a data or address register. One indicates an address register; zero indicates a data register. For other commands, this field may be interpreted differently.

5.7.3.3 MOVE INSTRUCTION. The MOVE instruction table indicates the number of clock periods needed for the processor to calculate the destination EA and to perform a MOVE or MOVEA instruction. For entries with CEA or FEA, refer to the appropriate table to calculate that portion of the instruction time.

Destination EAs are divided by their formats (see **5.3.4.4 Effective Address Encoding Summary**). The total number of clock cycles is outside the parentheses. The numbers inside parentheses (r/p/w) are included in the total clock cycle number. All timing data assumes two-clock reads and writes.

When using this table, begin at the top and move downward. Use the first entry that matches both source and destination addressing modes.

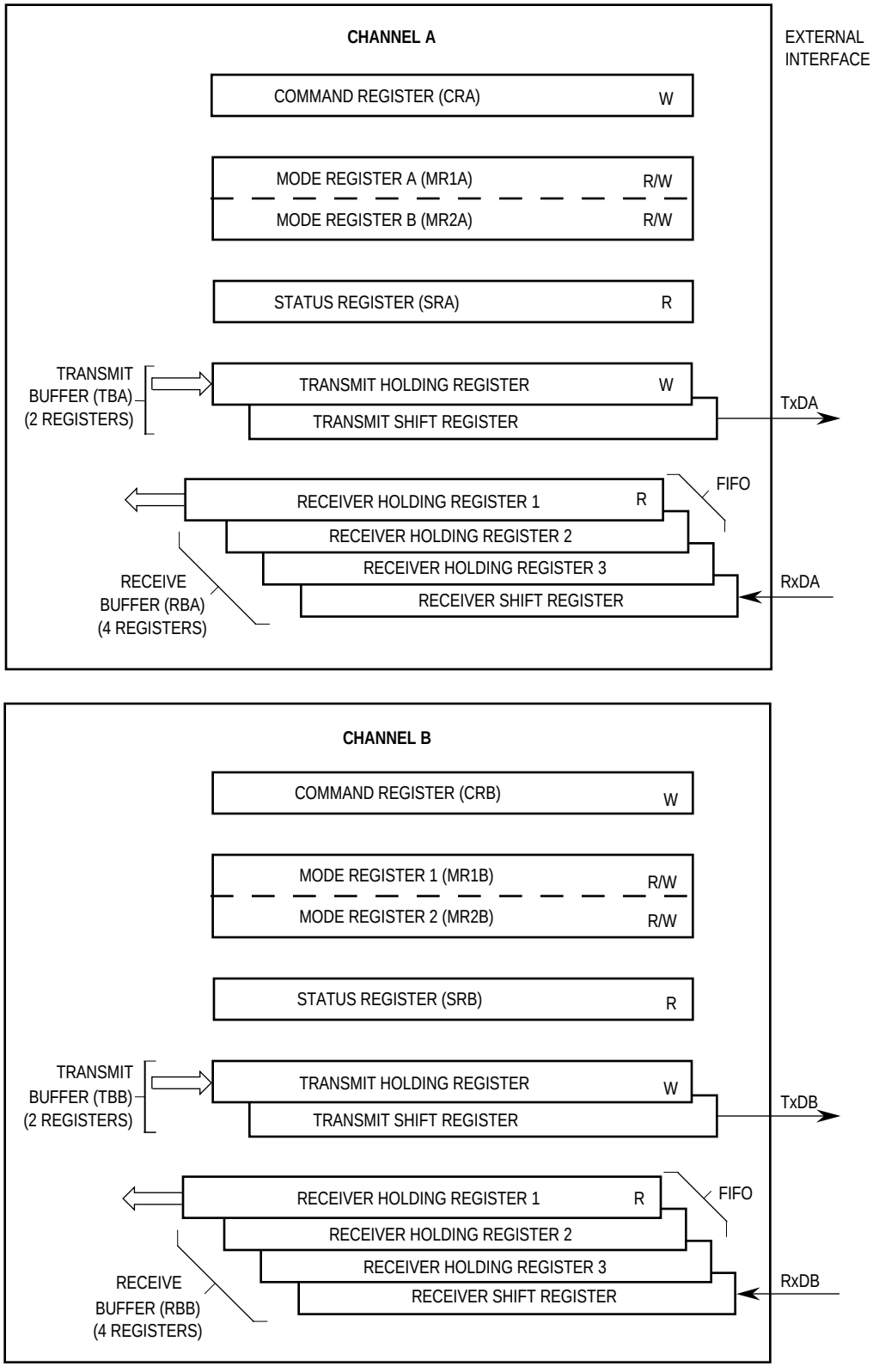
Instruction	Head	Tail	Cycles
MOVE Rn, Rn	0	0	2(0/1/0)
MOVE ⟨FEA⟩, Rn	0	0	2(0/1/0)
MOVE Rn, (Am)	0	2	4(0/1/x)
MOVE Rn, (Am)+	1	1	5(0/1/x)
MOVE Rn, –(Am)	2	2	6(0/1/x)
MOVE Rn, ⟨CEA⟩	1	3	5(0/1/x)
MOVE ⟨FEA⟩, (An)	2	2	6(0/1/x)
MOVE ⟨FEA⟩, (An)+	2	2	6(0/1/x)
MOVE ⟨FEA⟩, –(An)	2	2	6(0/1/x)
MOVE #, ⟨CEA⟩	2	2	6(0/1/x)*
MOVE ⟨CEA⟩, ⟨FEA⟩	2	2	6(0/1/x)

X = There is one bus cycle for byte and word operands and two bus cycles for long-word operands. For long-word bus cycles, add two clocks to the tail and to the number of cycles.

* = An # fetch EA time must be added for this instruction: ⟨FEA⟩+⟨CEA⟩+⟨OPER⟩

NOTE: For instructions not explicitly listed, use the MOVE ⟨CEA⟩, ⟨FEA⟩ entry. The source EA is calculated by the calculate EA table, and the destination EA is calculated by the fetch EA table, even though the bus cycle is for the source EA.

5.7.3.4 SPECIAL-PURPOSE MOVE INSTRUCTION. The special-purpose MOVE instruction table indicates the number of clock periods needed for the processor to fetch, calculate, and perform the special-purpose MOVE operation on control registers or a specified EA. Footnotes indicate when to account for the appropriate EA times. The total number of clock cycles is outside the parentheses. The numbers inside parentheses (r/p/w) are included in the total clock cycle number. All timing data assumes two-clock reads and writes.



NOTE:
R/W = READ/WRITE
R = READ
W = WRITE

Figure 7-4. Transmitter and Receiver Functional Diagram

The MC68340 includes on-chip circuitry to detect the initial application of power to the device. Power-on reset (POR), the output of this circuitry, is used to reset both the system and IEEE 1149.1 logic. The purpose for applying POR to the IEEE 1149.1 circuitry is to avoid the possibility of bus contention during power-on. The time required to complete device power-on is power-supply dependent. However, the IEEE 1149.1 TAP controller remains in the test-logic-reset state while POR is asserted. The TAP controller does not respond to user commands until POR is negated.

The MC68340 features a low-power stop mode that uses a CPU instruction called LPSTOP. The interaction of the IEEE 1149.1 interface with LPSTOP mode is as follows:

1. Leaving the TAP controller test-logic-reset state negates the ability to achieve minimal power consumption, but does not otherwise affect device functionality.
2. The TCK input is not blocked in LPSTOP mode. To consume minimal power, the TCK input should be externally connected to V_{CC} or ground.
3. The TMS and TDI pins include on-chip pullup resistors. In LPSTOP mode, these two pins should remain either unconnected or connected to V_{CC} to achieve minimal power consumption.

9.6 NON-IEEE 1149.1 OPERATION

In non-IEEE 1149.1 operation, there are two constraints. First, the TCK input does not include an internal pullup resistor and should be pulled up externally to preclude mid-level inputs. The second constraint is to ensure that the IEEE 1149.1 test logic is kept transparent to the system logic by forcing the TAP controller into the test-logic-reset state, using either of two methods. During power-on, POR forces the TAP controller into this state. Alternatively, sampling TMS as a logic one for five consecutive TCK rising edges also forces the TAP controller into this state. If TMS either remains unconnected or is connected to V_{CC}, then the TAP controller cannot leave the test-logic-reset state, regardless of the state of TCK.

The MC68340V low voltage parts can operate up to 8.39 MHz or 16.78 MHz with a 3.3 V ±0.3 V supply. Separate part numbers are used to distinguish the operation of the parts according to the supply voltage. Refer to **Section 12 Ordering Information and Mechanical Data** for the part numbering schemes. MC68340 is used throughout this section to refer to the 16.78- or 25.16-MHz parts at 5.0 V ±5%. MC68340V is used throughout this section to refer to the 8.39- or 16.78-MHz parts at 3.3 V ±0.3 V.

NOTE

The electrical specifications in this section for the MC68340 25.16 MHz at 5.0 V ±5% and the 3.3 V ±0.3 V specifications for both the 8.39- and 16.78-MHz parts are preliminary.

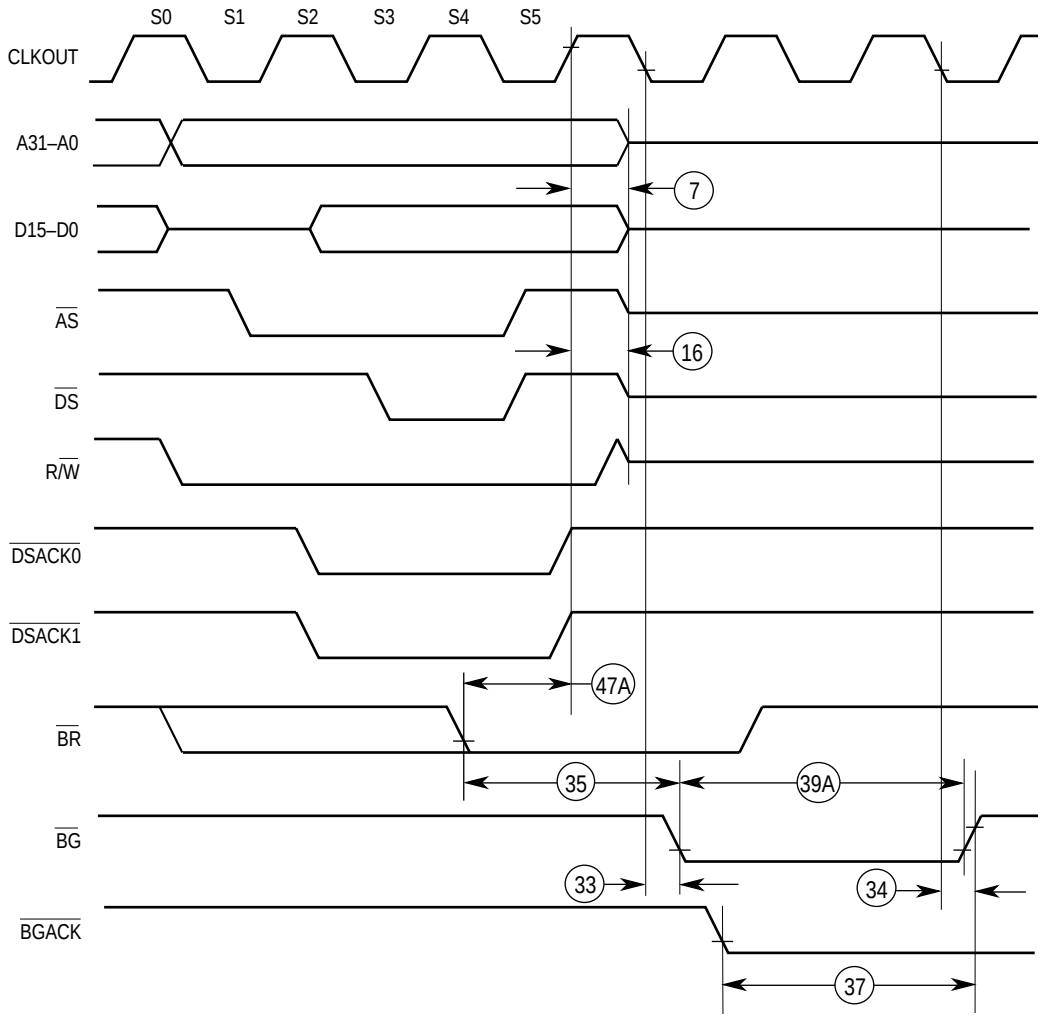


Figure 11-6. Bus Arbitation Timing—Active Bus Case

12.2 PIN ASSIGNMEN — CERAMIC SURFACE MOUNT

12.2.1 144-Lead Ceramic Quad Flat Pack (FE Suffix)