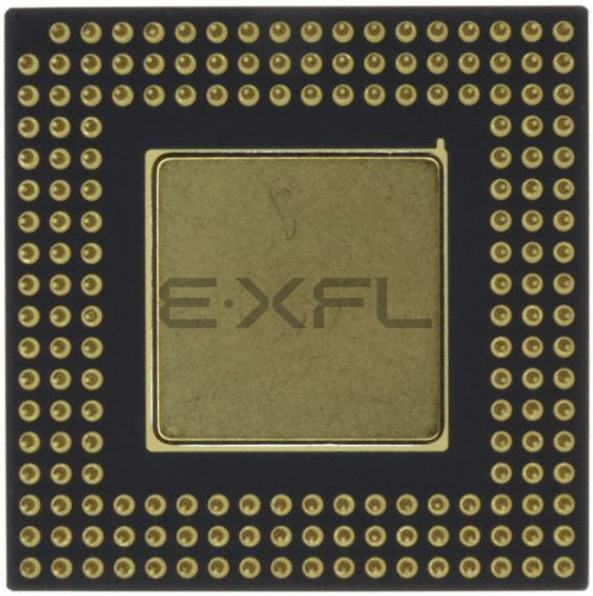


Welcome to E-XFL.COM

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	68040
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	40MHz
Co-Processors/DSP	-
RAM Controllers	-
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	179-BEPGA
Supplier Device Package	179-PGA (47.24x47.24)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mc68040rc40a

3.1.3 Transparent Translation Registers

The data transparent translation registers (DTTR0 and DTTR1) and instruction transparent translation registers (ITTR0 and ITTR1) are 32-bit registers that define blocks of logical address space. The TTRs operate independently of the E-bit in the TCR and the state of the MDIS signal. Data transfers to and from these registers are long-word transfers. The TTR fields are defined following Figure 3-5, which illustrates TTR format. Bits 12–10, 7, 4, 3, 1, and 0 always read as zero.

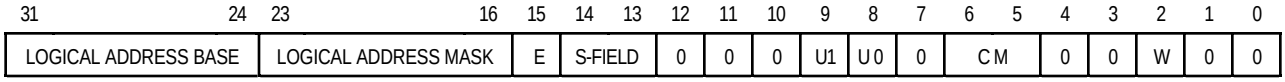


Figure 3-5. Transparent Translation Register Format

Logical Address Base

This 8-bit field is compared with address bits A31–A24. Addresses that match in this comparison (and are otherwise eligible) are transparently translated.

Logical Address Mask

Since this 8-bit field contains a mask for the Logical Address Mask field, setting a bit in this field causes the corresponding bit in the Logical Address Base field to be ignored. Blocks of memory larger than 16 Mbytes can be transparently translated by setting some of the logical address mask bits to ones. The low-order bits of this field can be set to define contiguous blocks larger than 16 Mbytes.

E—Enable

This bit enables or disables transparent translation of the block defined by this register:

0 = Transparent translation disabled
1 = Transparent translation enabled

S—Supervisor Mode

This field specifies the way FC2 is used in matching an address:

00 = Match only if FC2 = 0 (user mode access)
01 = Match only if FC2 = 1 (supervisor mode access)
1X = Ignore FC2 when matching

U0, U1—User Page Attributes

The user defines these bits, and the M68040 does not interpret them. U0 and U1 are echoed to the UPA0 and UPA1 signals, respectively, if an external bus transfer results from an access. These bits can be programmed by the user to support external addressing, bus snooping, or other applications.

Each ATC entry consists of a physical address, attribute information from a corresponding page descriptor, and a tag that contains a logical address and status information. Figure 3-21, which illustrates the entry and tag fields, is followed by field definitions listed in alphabetical order.

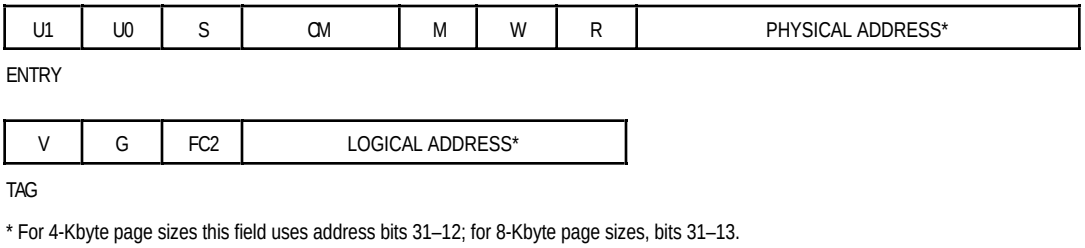


Figure 3-21. ATC Entry and Tag Fields

CM—Cache Mode

This field selects the cache mode and accesses serialization as follows:

00 = Cachable, Write-through
01 = Cachable, Copyback
10 = Noncachable, Serialized
11 = Noncachable

Section 4 Instruction and Data Caches provides detailed information on caching modes, and **Section 7 Bus Operation** provides information on serialization.

FC2—Function Code Bit 2 (Supervisor/User)

This bit contains the function code corresponding to the logical address in this entry. FC2 is set for supervisor mode accesses and cleared for user mode accesses.

G—Global

When set, this bit indicates the entry is global. Global entries are not invalidated by the PFLUSH instruction variants that specify nonglobal entries, even when all other selection criteria are satisfied.

Logical Address

This 13-bit field contains the most significant logical address bits for this entry. All 16 bits of this field are used in the comparison of this entry to an incoming logical address when the page size is 4 Kbytes. For 8-Kbytes pages, the least significant bit of this field is ignored.

M—Modified

The modified bit is set when a valid write access to the logical address corresponding to the entry occurs. If the M-bit is clear and a write access to this logical address is attempted, the M68040 suspends the access, initiates a table search to set the M-bit in the page descriptor, and writes over the old ATC entry with the current page descriptor information. The MMU then allows the original write access to be performed. This

By appropriately configuring a TTR, flexible transparent mappings can be specified (refer to **3.1.3 Transparent Translation Registers** for field identification). For instance, to transparently translate the user address space, the S-field is set to $0, and the logical address mask is set to $FF in both an instruction and data TTR. To transparently translate supervisor accesses of addresses $00000000–$0FFFFFFF with write protection, the logical base address field is set to $0x, the logical address mask is set to $0F, the W-bit is set to one, and the S-field is set to $1. The inclusion of independent TTRs in both the instruction and data MMUs provides an exception to the merged instruction and data address space, allowing different translations for instruction and operand accesses. Also, since the instruction memory unit is only used for instruction prefetches, different instruction and data TTRs can cause PC relative operand fetches to be translated differently from instruction prefetches.

If either of the TTRs matched during an access to a memory unit (either instruction or data), the access is transparently translated. If both registers match, the TT0 status bits are used for the access. Transparent translation can also be implemented by the translation tables of the translation tables if the physical addresses of pages are set equal to their logical addresses.

3.5 ADDRESS TRANSLATION SUMMARY

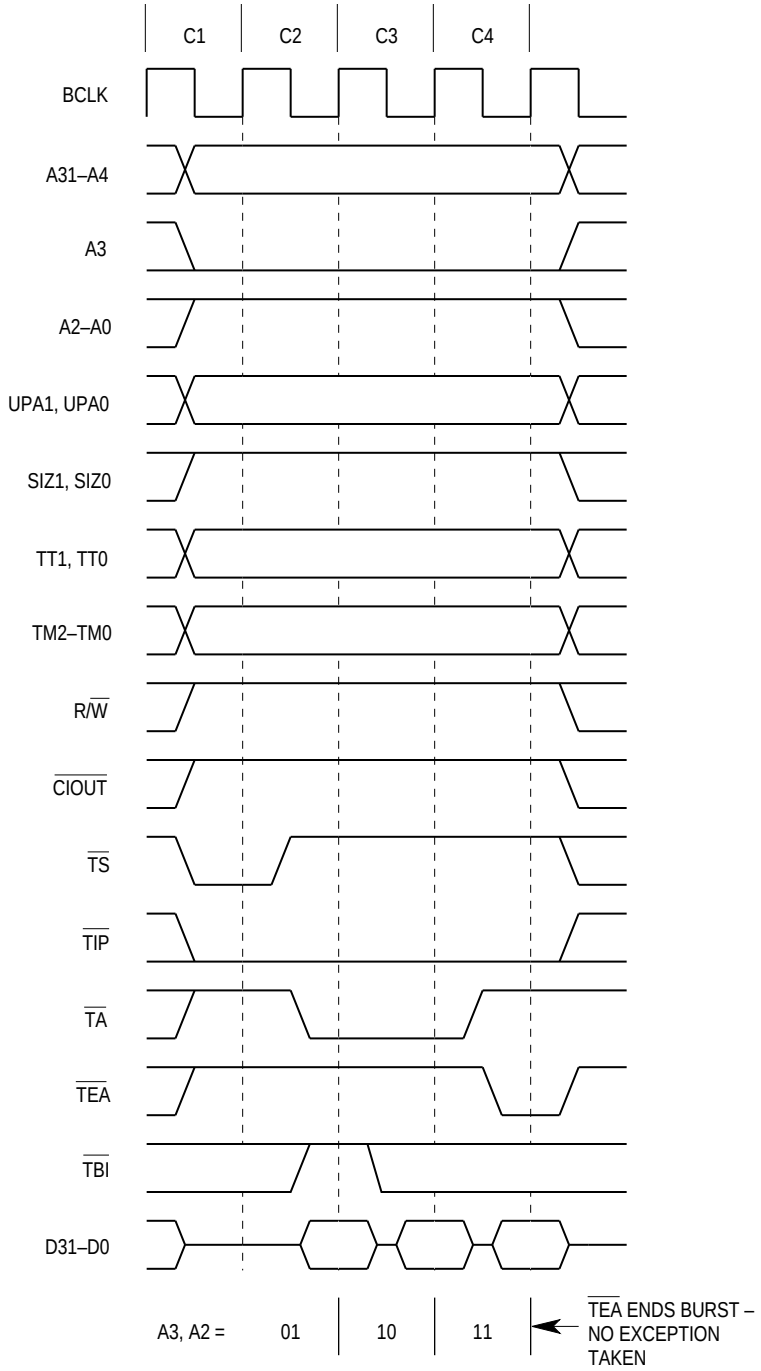
The instruction and data MMUs process translations by first comparing the logical address and privilege mode with the parameters of the TTRs. If there is a match, the MMU uses the logical address as a physical address for the access. If there is no match, the MMU compares the logical address and privilege mode with the tag portions of the entries in the ATC and uses the corresponding physical address for the access when a match occurs. When neither a TTR nor a valid ATC entry matches, the MMU initiates a table search operation to obtain the corresponding physical address from the translation table. When a table search is required, the processor suspends instruction execution activity and, at the end of a successful table search, stores the address mapping in the appropriate ATC and retries the access. The MMU creates a valid ATC entry for the logical address, and the access is retried. If an access hits in the ATC but an access error or invalid page descriptor was detected during the table search that created the ATC entry, the access is aborted, and a bus error exception is taken.

If a write or read-modify-write access results in an ATC hit but the page is write protected, the access is aborted, and an access error exception is taken. If the page is not write protected and the modified bit of the ATC entry is clear, a table search proceeds to set the modified bit in both the page descriptor in memory and in the ATC; the access is retried. The ATC provides the address translation for the access if the modified bit of the ATC entry is set for a write or read-modify-write access to an unprotected page, if the resident bit is set (indicating the table search for the entry completed successfully), and if none of the TTRs (instruction or data, as appropriate) match.

An ATC access error is not reported immediately, if the last 16 bits of a page is either an A-line, illegal, CHK, or unimplemented instruction and the next page is non-resident. Instead, the M68040 attempts to prefetch the next instruction on the missing page, then the ATC access error exception is reported. The stacked PC points to the exceptional

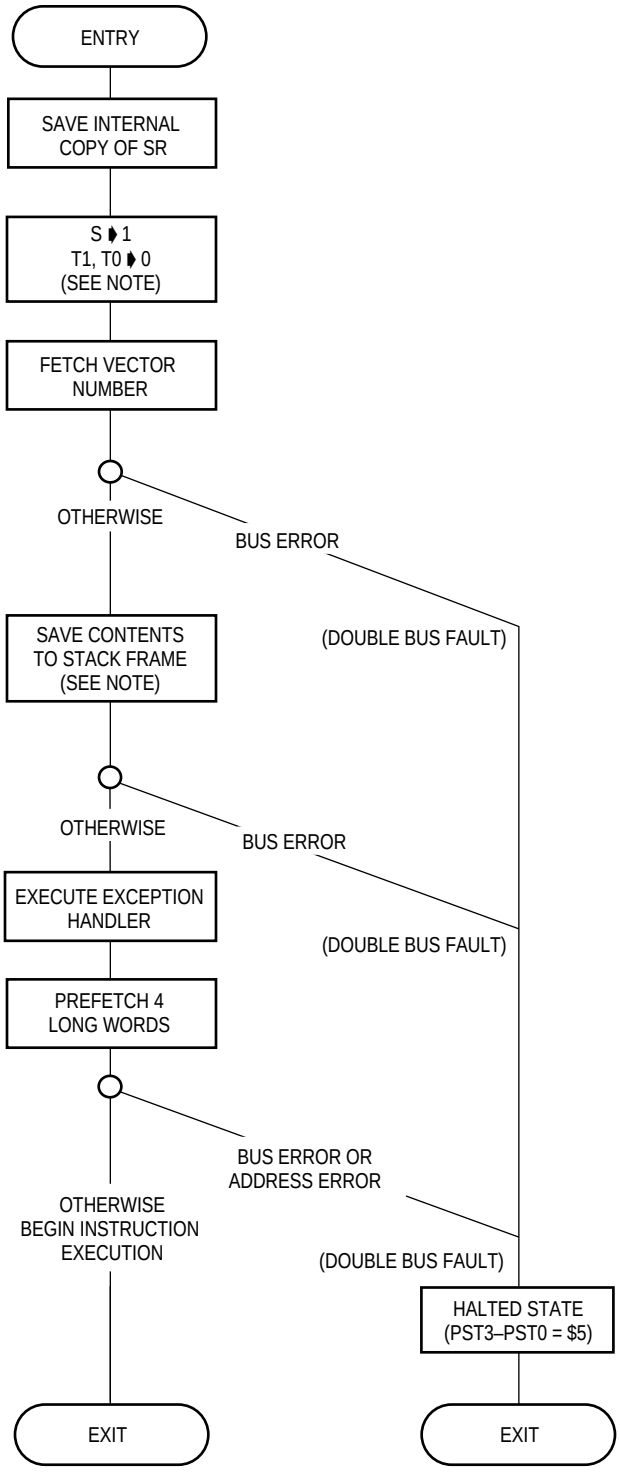
Table 4-4. Data-Cache Line State Transitions (Continued)

Cache Operation	Current State					
	Invalid Cases		Valid Cases		Dirty Cases	
Alternate Master Read Hit (Snoop Control = 10 — Invalidate)	I10	Not Possible	V10	No action; go to invalid state.	D10	Inhibit memory and source data; go to invalid state
Alternate Master Write Hit (Snoop Control = 10 —Invalidate)	I11	Not Possible	V11	No action; go to invalid state.	D11	No action; go to invalid state.
Alternate Master Write Hit (Snoop Control = 01 — Sink Data and Size ≠ Line)	I12	Not Possible	V12	No action; go to invalid state.	D12	Inhibit memory and sink data; set Dn bits of modified long words; remain in current state.
Alternate Master Write Hit (Snoop Control = 01 — Sink Data and Size = Line)	I13	Not Possible	V13	No action; go to invalid state.	D13	No action; go to invalid state.



NOTE: The selected device increments the value on A3 and A2.

Figure 7-27. Line Read Access Terminated with TEA Timing



NOTE: These blocks vary for reset and interrupt exceptions.

Figure 8-1. General Exception Processing Flowchart

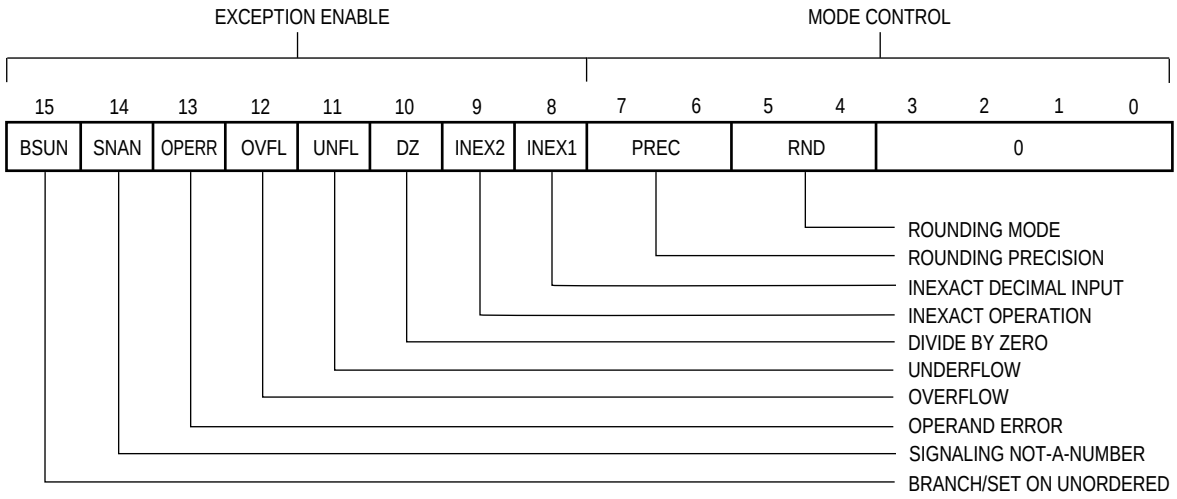


Figure 9-2. Floating-Point Control Register

9.2.3 Floating-Point Status Register (FPSR)

The FPSR (see Figure 9-1) contains a floating-point condition code (FPCC) byte, a quotient byte, a floating-point exception status byte (EXC), and a floating-point accrued exception byte (AEXC). The user can read or write to all bits in the FPSR. Execution of most floating-point instructions modifies this register. The reset function or a restore operation of the null state clears the FPSR. Floating-point conditional operations are not guaranteed if the FPSR is written directly, because the FPSR is only valid as a result of a floating-point instruction.

9.2.3.1 FLOATING-POINT CONDITION CODE BYTE. The FPCC byte (see Figure 9-3) contains four condition code bits that are set at the end of all arithmetic instructions involving the floating-point data registers. These bits are sign of mantissa (N), zero (Z), infinity (I), and NAN. The FMOVE FPm,<ea>, FMOVEM FPm, and FMOVE FPCR instructions do not affect the FPCC.

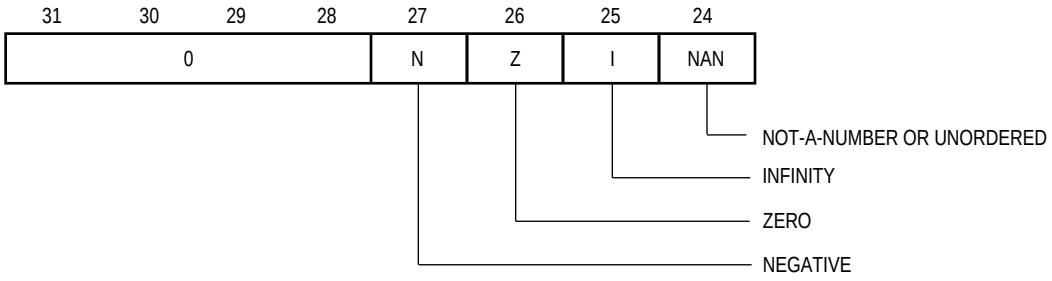


Figure 9-3. FPSR Condition Code Byte

To aid programmers of floating-point subroutine libraries, the MC68040 implements the four FPCC bits in hardware instead of only implementing the four IEEE conditions. An instruction derives the IEEE conditions when needed. For example, the programmers of a complex arithmetic multiply subroutine usually prefer to handle special data types such as

Table 9-15, and the user INEX exception handler can choose to modify these values. The E3 and E1 of the floating-point state frame bits need to be examined to determine which fields in the floating-point state frame are valid. E3 always takes precedence and must always be serviced first. Table 9-16 lists the floating-point state frame fields for INEX exceptions with E3 set or with E3 clear and E1 set. It is possible for an FADD, FSUB, FMUL, and FDIV to report a post-instruction exception, although these instructions normally generate a pre-instruction exception. The following example shows why a post-instruction exception is generated.

```
FADD      FP2,FP0      ; this instruction generates an inexact exception
FMOVE     FP0, <ea>    ; this instruction is executing when inexact occurs
```

For this example, assume that the FMOVE instruction starts once the FADD instruction generates an underflow. Given the register dependency on FP0, the destination of the FADD instruction, FP0 needs to be resolved prior to the FMOVE instruction execution. For this example, there is no choice but to have the FADD instruction report a post-instruction exception immediately. Note that for this case, even though the T-bit of the floating-point state frame is set (post instruction exception), it does not imply an FMOVE OUT instruction. Therefore, the effective address field in the format $3 stack frame is invalid.

The FMOVE OUT instruction generates a post-instruction exception. For this case, the effective address field in the format $3 stack frame points to the destination memory location. If the destination is an integer data register, the FPIAR points to the F-line word of the offending instruction, and the F-line word contains the integer data register number. If the MC68040FPSP unimplemented instruction exception handler is used, there can be some other cases in which an inexact exception is reported.

The user INEX exception handler examines the E3 bit of the floating-point state frame to exit from this exception handler. If the E3 bit is set, it must be cleared prior to restoring the floating-point frame via the FRESTORE instruction. If the E3 bit is clear and the E1 bit is set, the floating-point frame is discarded. The RTE instruction must be executed to return to normal instruction flow.

NOTE

The IEEE 754 standard specifies that inexactness should be signaled on overflow as well as for rounding. The processor implements this via the INEX bit in the FPSR AEXC byte. However, the standard also indicates that the inexact exception should be taken if an overflow occurs with the OVFL bit disabled and the INEX bit enabled in the FPSR AEXC byte. Therefore, the processor takes the inexact exception if this combination of conditions occurs, even though the INEX1 or INEX2 bit may not be set in the FPSR EXC byte. In this case, the INEX bit is set in the FPSR AEXC byte, and the OVFL bit is set in both the FPSR EXC and AEXC bytes.

Table 9-16. State Frame Field Information (Concluded)

FSAVE State Frame Field	Contents
INEX (FMOVE to Register, FABS, and FNEG)	
CMDREG1B	Exception Instruction Command Word
FPTEMP	Unrounded, Extended-Precision Intermediate Result
STAG	Source Operand Tag = Normalized
E1	Always 1
T	Always 0
INEX (FADD, FSUB, FMUL, FDIV, and FSQRT)	
CMDREG3B	Encoded Exception Instruction Command Word
WBTEMP	WBTS, WBTE, and WBTM = intermediate result sign, biased 15-bit exponent, and 64-bit mantissa prior to rounding.
WBTE15	Either 1 or 0, generally useless for INEX exceptions.
WBTM1, WBTM0, SBIT	Guard, round, and sticky of intermediate result's 67-bit mantissa.
E3	Always 1
T	Either 1 or 0
INEX (FMOVE to Memory)	
CMDREG1B	FMOVE Instruction Command Word
FPTEMP	Intermediate result with mantissa prior to rounding.
STAG	Source Operand Tag = Normalized
E1	Always 1
T	Always 1

NOTE: If the M68040FPSP unimplemented exception handler is used, the above state frame information applies. The CMDREG1B or CMDREG3B fields of the state frame are modified as appropriate to encode the unimplemented instruction opcode. It is the user exception handler's responsibility to use the E3 and E1 field encodings to recognize which state frame information applies. When E3 = 1 and E1 = 1, E3 takes priority and the state frame information for E3 = 1 must be used.

10.1 OVERVIEW

Refer to **Section 2 Integer Unit** for information on the integer unit pipeline. The <ea> fetch timing is not listed in the following tables because most instructions require one clock in the <ea> fetch stage for each memory access to obtain an operand. An instruction requires one clock to pass through the <ea> fetch stage even if no operand is fetched. Table 10-2 summarizes the number of memory fetches required to access an operand using each addressing mode for long-word aligned accesses. The user must perform his own calculations for <ea> fetch timing for misaligned accesses.

Table 10-2. Number of Memory Accesses

Addressing Mode	Evaluate <ea> And Fetch Operand	Evaluate <ea> And Send To Execution Stage
Dn	0	0
An	0	0
(An)	1	0
(An)+	1	0
–(An)	1	0
(d_{16},An)	1	0
(d_{16},PC)	1	0
(xxx).W, (xxx).L	1	0
#<xxx>	0	0
(d_8,An,Xn)	1	0
(d_8,PC,Xn)	1	0
(BR,Xn)	1	0
(bd,BR,Xn)	1	0
([bd,BR,Xn])	2	1
([bd,BR,Xn],od)	2	1
([bd,BR],Xn)	2	1
([bd,BR],Xn,od)	2	1

In the instruction timing tables, the <ea> calculate column lists the number of clocks required for the instruction to execute in the <ea> calculate stage of the integer unit pipeline. Dual effective address instructions such as ABCD –(Ay),–(Ax) require two calculations in the <ea> calculate stage and two memory fetches. Due to pipelining, the fetch of the first operand occurs in the same clock as the <ea> calculation for the second operand.

The execute column lists the number of clocks required for the instruction to execute in the execute stage of the integer unit pipeline. This number is presented as a lead time and a base time. The lead time is the number of clocks the instruction can stall when entering the execution stage without delaying the instruction execution. If the previous instruction is still executing in the execution stage when the current instruction is ready to move from the <ea> fetch stage, the current instruction stalls until the previous one completes. For

10.7.3 Timings in the Floating-Point Unit (Continued)

Instruction	Opclass	Size	Precision	Operands	Conversion	Execution	Normalization
FDIV	2	S,D	Any	— ,NAN	4	0	0
	2	X	Any	Norm,Norm	3(4)	37.5	2(3)
	2	X	Any	— ,Zero	5	0	0
	2	—	Any	— ,Inf	5	0	0
	2	X	Any	— ,NAN	5	0	0
FSQRT	0	—	Any	Norm	2(3)	103	2(3)
	0	—	Any	(Zero|Inf|NAN)	4	0	0
	2	S,D	Any	Norm	2(3)	103	2(3)
	2	S,D	Any	(Zero|Inf|NAN)	4	0	0
	2	X	Any	Norm	3(4)	103	2(3)
	2	X	Any	(Zero|Inf|NAN)	5	0	0
FMOVE,	0	—	X	(Norm|Zero|Inf)	2	0	0
FABS,	0	—	X	NAN	3	0	0
FNEG	0	—	S,D	Norm	5	0	0
	0	—	S,D	(Zero|Inf)	3	0	0
	0	—	S,D	NAN	4	0	0
	2	S	Any	(Norm|Zero|Inf)	3	0	0
	2	S	Any	NAN	4	0	0
	2	D	D,X	(Norm|Zero|Inf)	3	0	0
	2	D	D,X	NAN	4	0	0
	2	D	S	Norm	5	0	0
	2	D	S	(Zero|Inf)	4	0	0
	2	D	S	NAN	5	0	0
	2	X	X	(Norm|Zero|Inf)	4	0	0
	2	X	X	NAN	5	0	0
	2	X	S,D	Norm	6	0	0
	2	X	S,D	(Zero|Inf)	5	0	0
	2	X	S,D	NAN	6	0	0
	2	B,W	Any	(+Norm|Zero)	1.5(11)	4.5	2
	2	L	D,X	(+Norm|Zero)	1.5(11)	4.5	2
	2	L	S	(+Norm|Zero)	1.5(12.5)	4.5	2
	2	B,W	Any	—Norm	1.5(11.5)	5	2
	2	L	D,X	—Norm	1.5(11.5)	5	2
	2	L	S	—Norm	1.5(13)	5	2
FMOVE	3	S,D	Any	Any	3	0	0
	3	X	Any	Any	4	0	0
	3	B,W,L	Any	+(Norm|Zero)	3(9)	1.5	3.5
	3	B,W,L	Any	–(Norm|Zero)	3(10)	1.5	4.5

11.3 DC ELECTRICAL SPECIFICATIONS (V_{CC} = 5.0 VDC ±5 %)

Characteristic	Symbol	Min	Max	Unit
Input High Voltage	V_{IH}	2	V_{CC}	V
Input Low Voltage	V_{IL}	GND	0.8	V
Undershoot	—	—	0.8	V
Input Leakage Current @ 0.5/2.4 V $\overline{AVEC}$, BCLK, $\overline{BG}$, $\overline{CDIS}$, $\overline{MDIS}$, $\overline{IPLx}$, PCLK, $\overline{RSTI}$, SCx, $\overline{TBI}$, TLNx , $\overline{TCI}$, TCK, $\overline{TEA}$	I_{in}	20	20	µA
Hi-Z (Off-State) Leakage Current @ 0.5/2.4 V An, $\overline{BB}$, $\overline{CIOUT}$, Dn, $\overline{LOCK}$, $\overline{LOCKE}$, R/$\overline{W}$, SIZx, $\overline{TA}$, TDO, $\overline{TIP}$, TMx, TLNx, $\overline{TS}$, TTx, UPAx	I_{TSI}	20	20	µA
Signal Low Input Current, V_{IL} = 0.8 V TMS, TDI, $\overline{TRST}$	I_{IL}	–1.1	–0.18	mA
Signal High Input Current, V_{IH} = 2.0 V TMS, TDI, $\overline{TRST}$	I_{IH}	–0.94	–0.16	mA
Output High Voltage, I_{OH} = 5 mA (Small Buffer Mode)	V_{OH}	2.4	—	V
Output Low Voltage, I_{OL} = 5 mA (Small Buffer Mode)	V_{OL}	—	0.5	V
Output High Voltage, I_{OH} = 55 mA (Large Buffer Mode)	V_{OH}	2.4	—	V
Output Low Voltage, I_{OL} = 55 mA (Large Buffer Mode)	V_{OL}	—	0.5	V
Capacitance*, V_{in} = 0 V, f = 1 MHz	C_{in}	—	25	pF

*Capacitance is periodically sampled rather than 100% tested.

11.4 POWER DISSIPATION

Buffer Mode	25 MHz	33 MHz	40 MHz
Worst Case (V_{CC} = 5.25 V , T_A = 0°C)			
Small Unterminated, I_{OL} = I_{OH} = 5 mA	4.9 W	6.2 W	7.2 W
Large Unterminated, I_{OL} = I_{OH} = 5 mA	5.1 W	6.6 W	7.7 W
Large Terminated, 50 Ω, 2.5 V, I_{OL} = I_{OH} = 55 mA	6.5 W	8.0 W	9.1 W
Typical Values (V_{CC} = 5 V, T_J = 90°C)*			
Small	3.0 W	4.1 W	4.5 W
Large Unterminated	3.3 W	4.4 W	4.8 W
Large Terminated, 50 Ω, 2.5 V	4.7 W	5.8 W	6.2 W

*This information is for system reliability purposes.

rated at a particular maximum operating frequency is dependent upon the power-up mode. Therefore, the MC68040 is rated at a maximum power dissipation for either the large or small buffers at a particular frequency. This allows for control of some of the thermal management upon reset. The following equation provides a rough method to calculate the maximum power consumption for a chosen output buffer mode:

$$P_D = P_{DSB} + (P_{DLB} - P_{DSB}) \times (PINS_{LB} \div PINS_{CLB})$$

where:

P_D = Maximum Power Dissipation for Output Buffer Mode Selected
P_{DSB} = Maximum Power Dissipation for Small Buffer Mode (All Outputs)
P_{DLB} = Maximum Power Dissipation for Large Buffer Mode (All Outputs)
$PINS_{LB}$ = Number of Pins Large Buffer Mode
$PINS_{CLB}$ = Number of Pins Capable of the Large Buffer Mode

Table 11-1 lists the simplified relationship on the maximum power dissipation for eight possible configurations of output buffer modes.

Table 11-1. Maximum Power Dissipation for Output Buffer Mode Configurations

Output Configuration			Maximum Power Dissipation
Data Bus	Address Bus and Transfer Attributes	Control Signals	
Small*	Small	Small	P_{DSB}
Small	Small	Large	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 13\%$
Small	Large	Small	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 52\%$
Small	Large	Large	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 65\%$
Large	Small	Small	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 35\%$
Large	Small	Large	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 48\%$
Large	Large	Small	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 87\%$
Large	Large	Large	$P_{DSB} + (P_{DLB} - P_{DSB}) \times 100\%$

*The MC68LC040 and MC68EC040 only utilize this row of information.

To calculate the specific power dissipation of a design, the termination method of each signal must be considered. For example, a signal output that is not connected would not dissipate any additional power if it were configured in the large rather than the small buffer mode. Since the maximum operating junction temperature is specified as 110°C, the maximum case temperature (T_C) in °C can be obtained from the following equation:

$$T_C = T_J - P_D \times \theta_{JC}$$

where:

T_C = Maximum Case Temperature
T_J = Maximum Junction Temperature
P_D = Maximum Power Dissipation of the Device
θ_{JC} = Thermal Resistance between the Junction of the Die and the Case

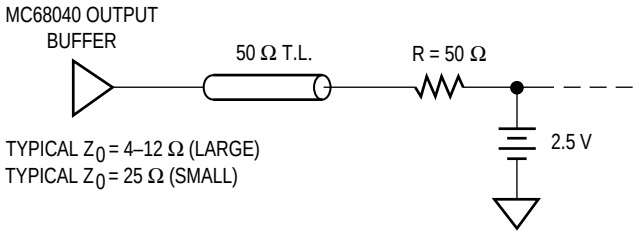


Figure 11-8. MC68040 Termination Network

If a designer uses alternative standard termination methods, such as RC termination network (see Figure 11-9), Thévenin termination network (not illustrated), or no termination method at all, which is not recommended, then the power dissipation of the MC68040 will be significantly less than the large buffer terminated values. For termination networks other than that illustrated in Figure 11-31, the designer must calculate the component of power dissipated in the output buffer and add this value to the small buffer unterminated value.

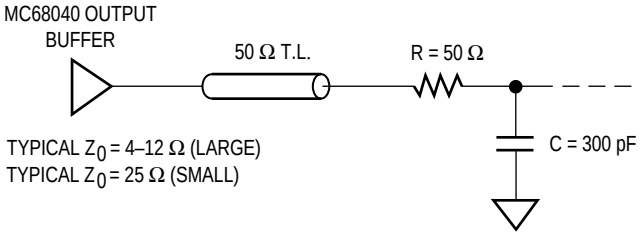


Figure 11-9. Typical Configuration for RC Termination Network

The following paragraphs describe how the large buffer terminated values were calculated. The MC68040 termination network causes current flow through the output buffer of the MC68040, regardless of whether the MC68040 is driving a logic one or a logic zero. The following equation gives the large buffer termination network power dissipation for a given pin:

$$I = (V \div (R + Z_o)) + 5\ mA$$

$$P = I^2 R_{eff}$$

R_{eff} is the effective average output resistance, including typical pullup resistance, typical pulldown resistance, and a duty cycle average of how often the pin is high, low, or three-stated. Typical values for Z_o are 6 Ω for large buffer low output, 12 Ω for large buffer high output, and 25 Ω for small buffer output. Using these values and duty cycle assumptions based on sequential burst write cycles, R_{eff} calculates to 7.7 Ω for the MC68040 large buffer mode and 25 Ω for the small buffer mode.

Maximum termination current in the large buffer mode occurs for output:

Low: $I_{tl} = (2.5\ V \div (50 + 6\ \Omega)) + 5\ mA = 49.6\ mA$

High: $I_{th} = (2.75\ V \div (50 + 12\ \Omega)) + 5\ mA = 50.8\ mA$

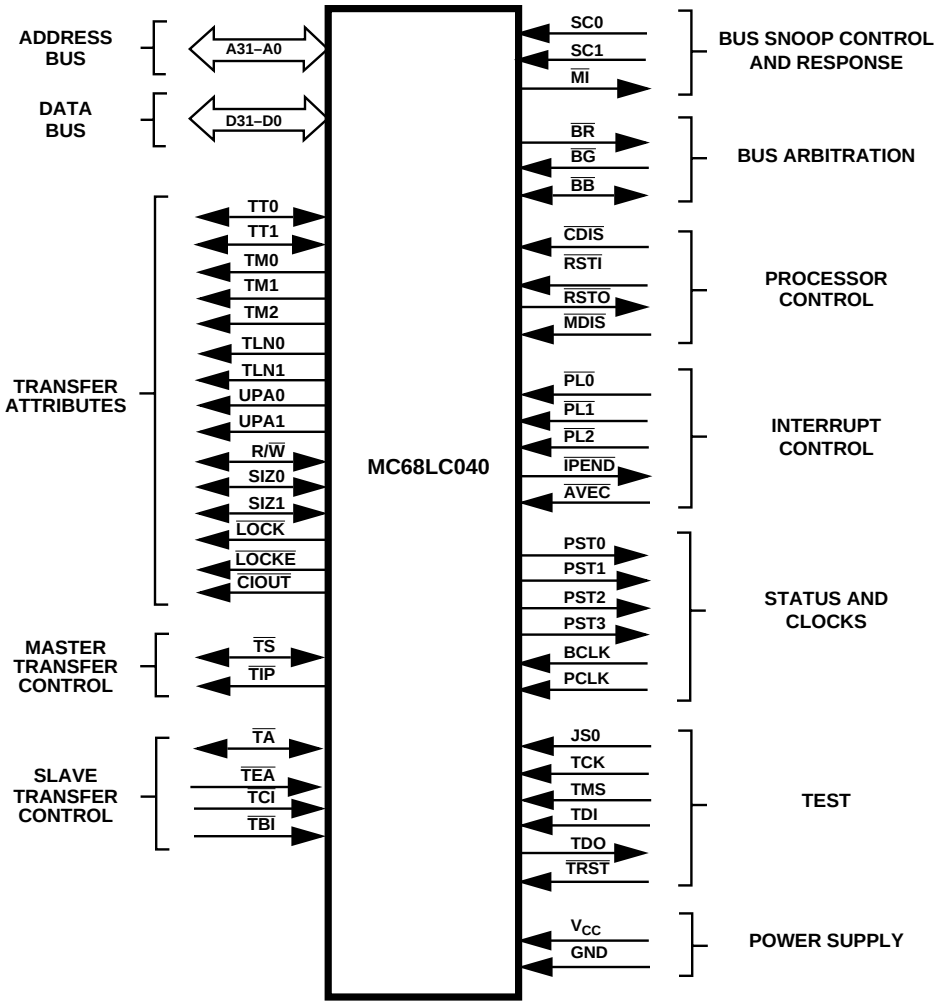
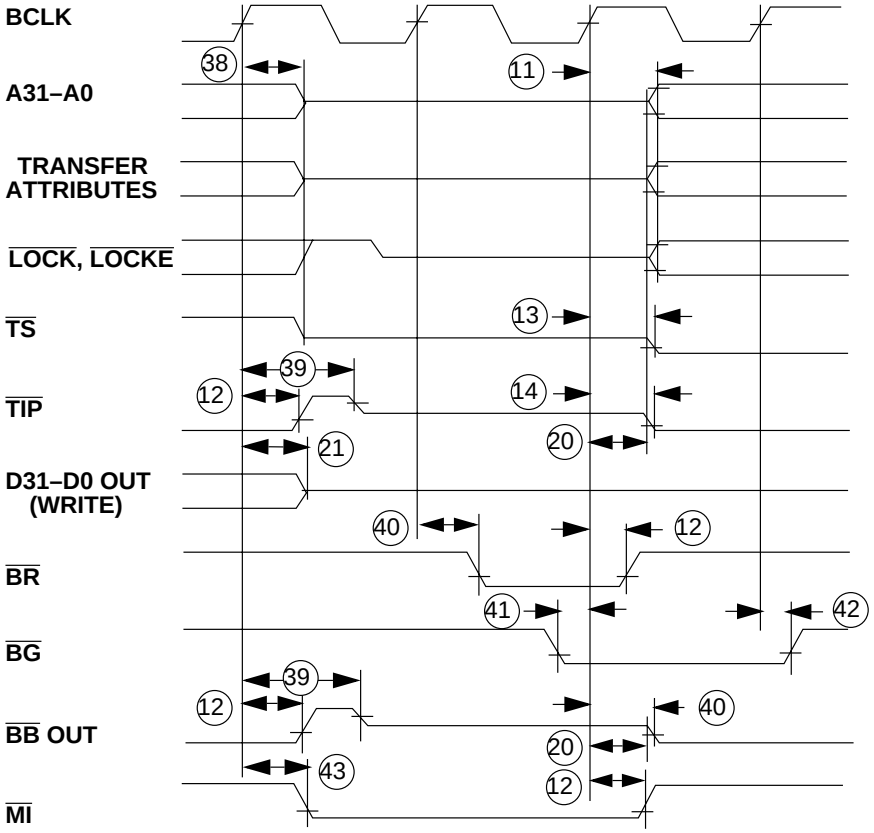


Figure A-3. MC68LC040 Functional Signal Groups

A.1 MC68LC040 DIFFERENCES

The following differences exist between the MC68LC040 and MC68040:

- The MC68LC040 does not implement the small output bufferr impedance selection mode.
- The DLE pin name has been changed to JS0
- The MC68LC040 does not implement the data latch (DLE) or multiplexed bus modes of operation. All timing and drive capabilities of the MC68LC040 are equivalent to those of the MC68040 in small output buffer impedance mode.
- The MC68LC040 does not contain an FPU, which causes unimplemented floating-point exceptions to occur using a new eight-word stack frame format.



NOTE: Transfer Attribute Signals = UPAx, SIZx, TTx, TMx, TLNx, R/$\overline{W}$, and $\overline{CIOUT}$

Figure A-6. Bus Arbitration Timing

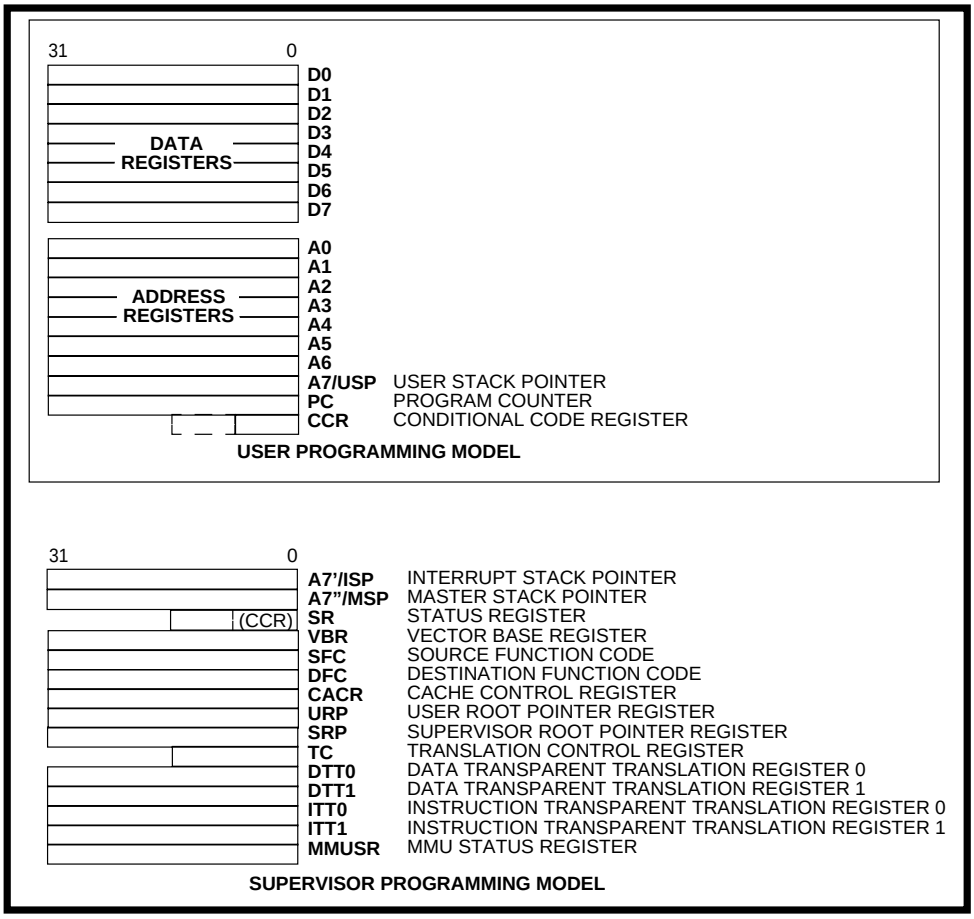


Figure B-2. MC68EC040 Programming Model

- PTEST and PFLUSH instructions cause an indeterminate result (i.e., an undetermined number of bus cycles); the user should not execute them on the MC68EC040.
- The MC68EC040 does not contain an FPU which causes unimplemented floating-point exceptions to occur using a new stack frame format.
- The DLE and $\overline{\text{MDIS}}$ pin names have been changed to JS0 and JS1, respectively.
- The MC68EC040 does not implement the DLE mode, multiplexed, or output buffer impedance selection modes of operation. The MC68EC040 implements only the small output buffer mode of operation. All timing and drive capabilities of the MC68EC040 are equivalent to those of the MC68040 in the small buffer mode of operation.

B.2 JTAG SCAN (JS1–JS0)

The MC68040 $\overline{\text{MDIS}}$ and DLE pin names have been changed to JS1 and JS0 respectively. During normal operation, the JS1 and JS0 pin cannot float, they must be tied to GND or Vcc directly or through a resistor. During board testing, these pins retain the functionality of the JTAG scan of the MC68040 for compatibility purposes. Refer to **Section 6 IEEE 1149.1A**

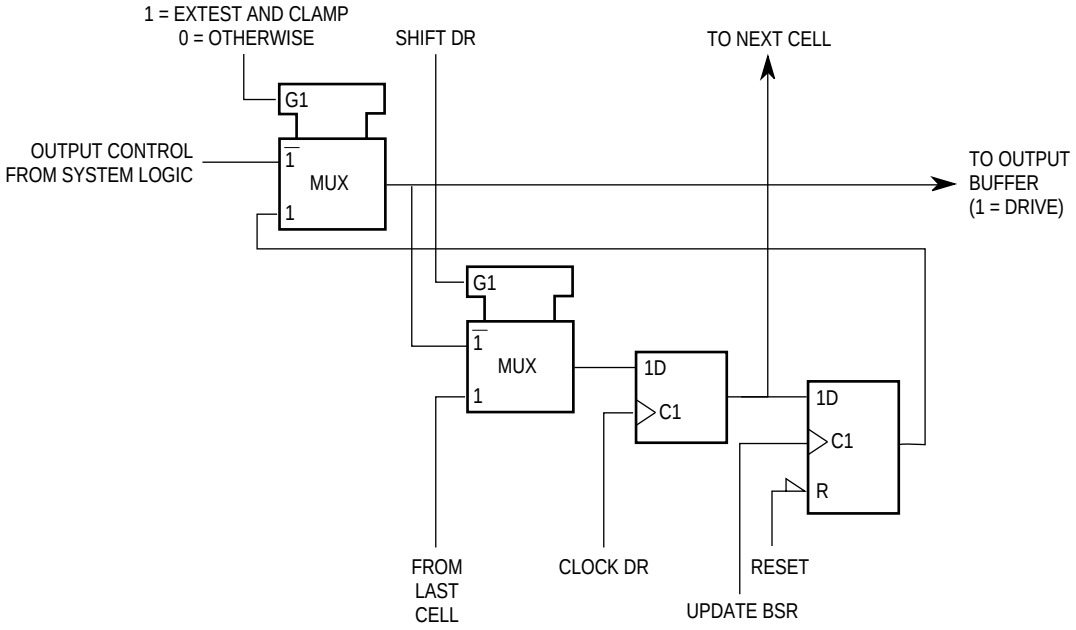


Figure C-8. Output Control Cells (IO.Ctl)

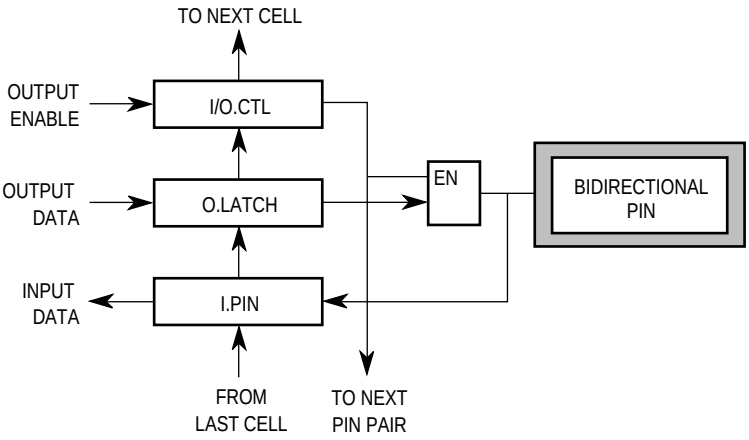
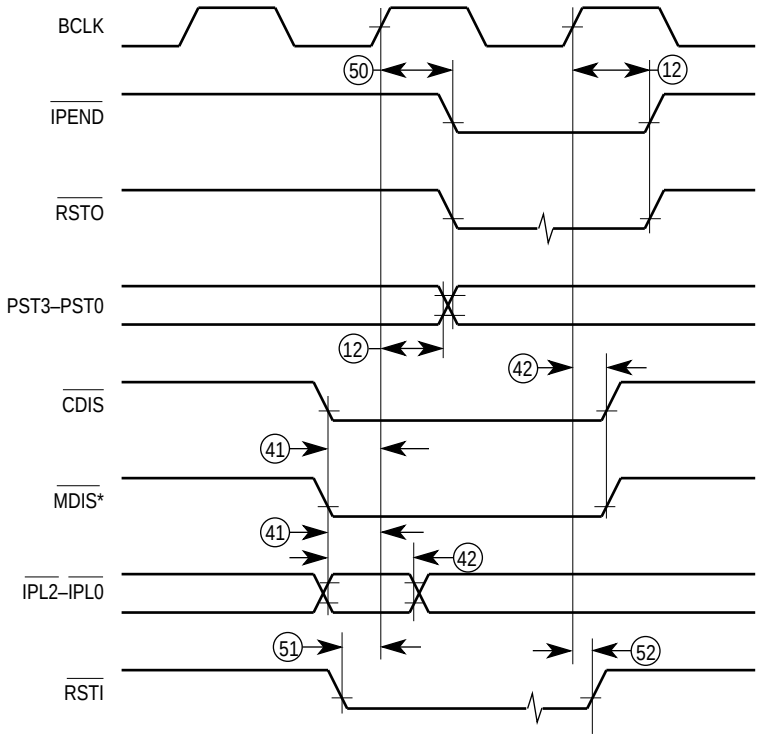


Figure C-9. General Arrangement of Bidirectional Pins



NOTE: *Not on MC68EC040V.

Figure C-17. Other Signal Timing

MC68020, MC68030, and MC68040 Instruction Set Extensions

Instruction	Notes	Applies To		
		MC68020	MC68030	MC68040
Bcc	Supports 32-Bit Displacements			
BFxxxx	Bit Field Instructions (BCHG, BFCLR, BFEXTS, BFEXTU, BFFFO, BFINS, BFSET, BFTST)			
BKPT	New Instruction Functionally			
BRA	Supports 32-Bit Displacement			
BSR	Supports 32-Bit Displacement			
CALLM	New Instruction			
CAS, CAS2	New Instructions			
CHK	Supports 32-Bit Operands			
CHK2	New Instruction			
CINV	Cache Maintenance Instruction			
CMPI	Supports Program Counter Relative Addressing Modes			
CMP2	New Instruction			
CPUSH	Cache Maintenance Instruction			
cp	Coprocessor Instructions			
DIVS/DIVU	Supports 32-Bit and 64-Bit Operands			
EXTB	Supports 8-Bit Extend to 32-Bits			
FABS	New Instruction			
FADD	New Instruction			
FBcc	New Instruction			
FCMP	New Instruction			
FDBcc	New Instruction			
FDIV	New Instruction			
FMOVE	New Instruction			
FMOVEM	New Instruction			
FMUL	New Instruction			
FNEG	New Instruction			
FNOP	New Instruction			
FRESTORE	New Instruction			
FSGLDIV	New Instruction			
FSGLMUL	New Instruction			
FSAVE	New Instruction			
FScc	New Instruction			
FSQRT	New Instruction			
FSUB	New Instruction			
FTRAPcc	New Instruction			
FTST	New Instruction			
LINK	Supports 32-Bit Displacement			