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	Active
Core Processor	68040
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	33MHz
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	182-BEPGA
Supplier Device Package	182-PGA (47.24x47.24)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mc68040rc33v

LIST OF ILLUSTRATIONS (Continued)

Figure Number	Title	Page Number
B-10	Snoop Hit Timing	B-19
B-11	Snoop Miss Timing	B-20
B-12	Other Signal Timing	B-21
C-1	MC68040V and MC68EC040V Functional Signal Groups	C-3
C-2	MC68040V and MC68EC040V Initial Power-On Reset Timing	C-8
C-3	MC68040V and MC68EC040V Normal Reset Timing	C-9
C-4	MC68040V and MC68EC040V Test Logic Block Diagram	C-11
C-5	Bypass Register	C-13
C-6	Output Latch Cell (O.Latch)	C-14
C-7	Input Pin Cell (I.Pin)	C-14
C-8	Output Control Cells (IO.Ctl)	C-15
C-9	General Arrangement of Bidirectional Pins	C-15
C-10	Circuit Disabling IEEE Standard 1149.1A	C-17
C-11	Drive Levels and Test Points for AC Specifications	C-18
C-12	Clock Input Timing Diagram	C-21
C-13	Read/Write Timing	C-24
C-14	Bus Arbitration Timing	C-25
C-15	Snoop Hit Timing	C-26
C-16	Snoop Miss Timing	C-27
C-17	Other Signal Timing	C-28
C-18	Going into LPSTOP with Arbitration	C-29
C-19	LPSTOP no Arbitration, CPU is Master	C-30
C-20	Exiting LPSTOP with Interrupt	C-31
C-21	Exiting of LPSTOP with RESET	C-31

procedure separates task-related supervisor activity from asynchronous, I/O-related supervisor tasks that can only be coincidental to the currently executing task. The MSP can separately maintain task control information for each currently executing user task, and the software updates the MSP when a task switch is performed, providing an efficient means for transferring task-related stack items. The value of the M-bit does not affect execution of privileged instructions. Instructions that affect the M-bit are MOVE to SR, ANDI to SR, EORI to SR, ORI to SR, and RTE. The processor automatically saves the M-bit value and clears it in the SR as part of the exception processing for interrupts.

2.2.2.2 STATUS REGISTER. The SR (see Figure 2-5) stores the processor status. In the supervisor mode, software can access the full SR, including the CCR available in user mode (see **2.2.1.5 Condition Code Register**) and the interrupt priority mask and additional control bits available only in the supervisor mode. These bits indicate the following states for the processor: one of two trace modes (T1, T0), supervisor or user mode (S), and master or interrupt mode (M).

The term SSP refers to the ISP and MSP. The M and S bits of the SR decide which SSP to use. When the S-bit is one and the M-bit is zero, the ISP is the active stack pointer; when the S-bit is one and the M-bit is one, the MSP is the active stack pointer. The ISP is the default mode after reset and corresponds to the MC68000, MC68008, MC68010, and CPU32 supervisor mode.

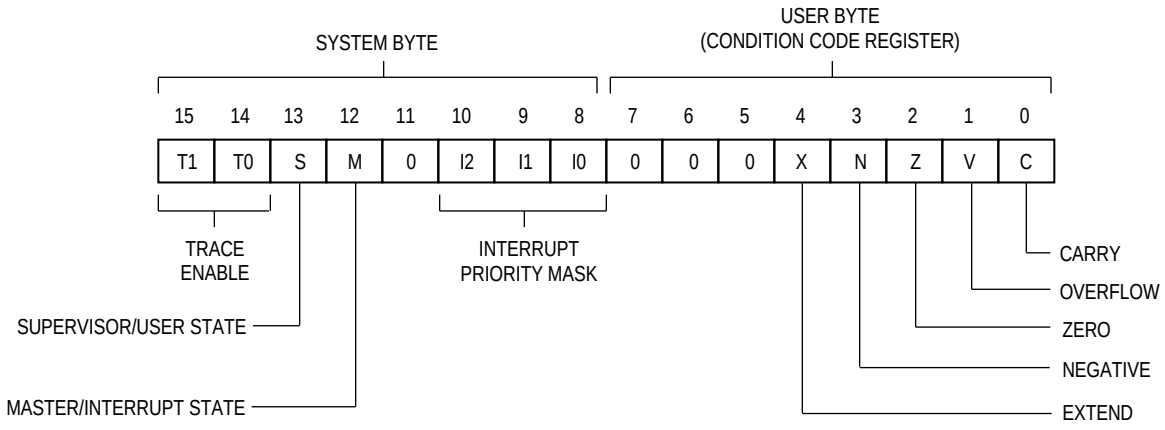


Figure 2-5. Status Register

2.2.2.3 VECTOR BASE REGISTER. The VBR contains the base address of the exception vector table in memory. The displacement of an exception vector is added to the value in this register to access the vector table. Refer to **Section 8 Exception Processing** for information on exception vectors.

2.2.2.4 ALTERNATE FUNCTION CODE REGISTERS. The alternate function code registers contain 3-bit function codes. Function codes can be considered extensions of the 32-bit logical address that optionally provides as many as eight 4-Gbyte address spaces. The processor automatically generates function codes to select address spaces for data and programs at the user and supervisor modes. Certain instructions use the SFC and DFC registers to specify the function codes for operations.

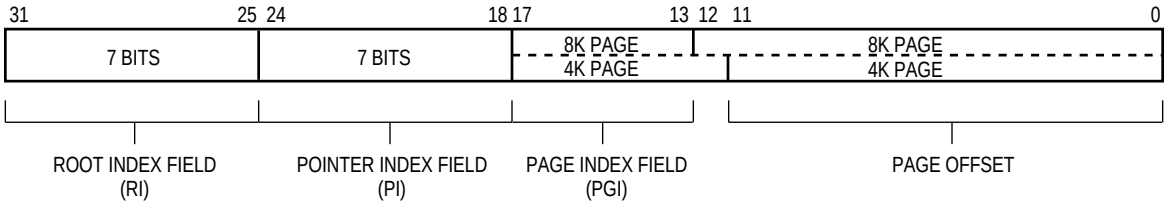


Figure 3-8. Logical Address Format

The seven bits of a logical address PI field are multiplied by 4 (shifted to the left by two bits) and concatenated with the fetched root-level descriptor's upper 23 bits to produce the physical address of the pointer-level table descriptor. Each of the 128 pointer-level table descriptors corresponds to a 256-Kbyte block of memory.

For 8-Kbyte pages, the five bits of the PGI field are multiplied by 4 (shifted to the left by two bits) and concatenated with the fetched pointer-level descriptor's upper 25 bits to produce the physical address of the 8-Kbyte page descriptor. The upper 19 bits of the page descriptor are the page frame's physical address. There are 32 8-Kbyte page descriptors in a page-level table.

Similarly, for 4-Kbyte pages, the six bits of the PGI field are multiplied by 4 (shifted to the left by two bits) and concatenated with the fetched pointer-level descriptor's upper 24 bits to produce the physical address of the 4-Kbyte page descriptor. The upper 20 bits of the page descriptor are the page frame's physical address. There are 64 4-Kbyte page descriptors in a page-level table.

Write-protect status is accumulated from each level's descriptor and combined with the status from the page descriptor to form the ATC entry status. The M68040 creates the ATC entry from the page frame address and the associated status bits and retries the original bus access. Refer to **3.3 Address Translation Caches** for details on ATC entries.

If the descriptor from a page table is an indirect descriptor, the page descriptor pointed to by this descriptor is fetched. Invalid descriptors can be used at any level of the tree except the root. When a table search for a normal translation encounters an invalid descriptor, the processor takes an access fault exception. The invalid descriptor can be used to identify either a page or branch of the tree that has been stored on an external device and is not resident in memory or a portion of the translation table that has not yet been defined. In these two cases, the exception routine can either restore the page from disk or add to the translation table. Figures 3-9 and 3-10 illustrate detailed flowcharts of table search and descriptor fetch operations.

A table search terminates successfully when a page descriptor is encountered. The occurrence of an invalid descriptor or a transfer error acknowledge also terminates a table search, and the M68040 takes an exception on the retry of the cycle because of these conditions. The exception handler should distinguish between anticipated conditions and true error conditions. The exception handler can correct an invalid descriptor that indicates a nonresident page or one that identifies a portion of the translation table yet to be allocated. An access error due to a system malfunction can require the exception handler to write an error message and terminate the task.

MC68030 and MC68851 cause F-line unimplemented instruction exceptions if executed in either supervisor or user mode by the M68040.

3.7.4 Register Programming Considerations

If the entries in the ATCs are no longer valid when a reset operation occurs (as is normally expected), an explicit flush operation must be specified by the system software. The assertion of RSTI disables translation by clearing the E-bits of the TCR, DTTRx, and ITTRx, but it does not flush the ATCs. Reading or writing any of the MMU registers (URP, SRP, TCR, MMUSR, DTTR0, DTTR1, ITTR0, ITTR1) does not flush the ATCs. Since a write to these registers can cause some or all the address translations to change, the write should be followed by a PFLUSH operation to flush the ATCs if necessary.

The status bits in the MMUSR indicate conditions to which the operating system should respond. In a typical access error exception handler, the flowchart illustrated in Figure 3-23 can be used to determine the cause of an MMU fault. The PTEST instruction sets the bits in the MMUSR appropriately, and the program can branch to the appropriate code segment for the condition.

6.2 INSTRUCTION SHIFT REGISTER

The M68040 IEEE standard 1149.1A implementation includes a 3-bit instruction shift register without parity. The register shifts one of eight instructions, which can either select the test to be performed or access a test data register, or both. Data is transferred from the instruction shift register to latched decoded outputs during the update-IR state. The instruction shift register is reset to all ones in the TAP controller test-logic-reset state, which is equivalent to selecting the BYPASS instruction. During the capture-IR state, the binary value 001 is loaded into the parallel inputs of the instruction shift register.

The M68040 IEEE standard 1149.1A implementation includes three mandatory public instructions (BYPASS, SAMPLE/PRELOAD, and EXTEST) and four manufacturer's public instructions. The four manufacturer's public instructions provide the capability to disable all device output drivers, operate the device in a BYPASS configuration without a system clocking requirement, and select one of two output drive capabilities on a pin-by-pin basis. The M68040 implementation does not support the optional standard public instructions. Table 6-1 lists the three bits used in the instruction shift register to decode the instructions and their related encodings. Note that the least significant bit of the instruction (bit 0) is the first bit to be shifted into the instruction shift register.

Table 6-1. IEEE Standard 1149.1A Instructions

Bit 2	Bit 1	Bit 0	Instruction Selected	Test Data Register Accessed
0	0	0	EXTEST	Boundary Scan
0	0	1	HIGHZ	Bypass
0	1	0	SAMPLE/PRELOAD	Boundary Scan
0	1	1	DRVCTL.T	Boundary Scan
1	0	0	SHUTDOWN	Bypass
1	0	1	PRIVATE	Bypass
1	1	0	DRVCTL.S	Boundary Scan
1	1	1	BYPASS	Bypass

EXTEST, HIGHZ, DRVCTL.T, SHUTDOWN, and PRIVATE have a PCLK and BCLK restriction. Failure to comply with this restriction results in potential internal damage to the device (see **6.4 Restrictions**). Once the restriction is complied with, SHUTDOWN, EXTEST, HIGHZ, and DRVCTL.T can be entered regardless of order. The system clocks (PCLK and BCLK) must be kept running while in the SAMPLE/PRELOAD, DRVCLT.S, and BYPASS instructions. Failure to do so could result in potential internal damage to the device.

6.2.1 EXTEST

The external test instruction (EXTEST) selects the 184-bit boundary scan register. This instruction also activates two internal functions that are intended to protect the device from potential damage while performing boundary scan operations.

6.6 MOTOROLA M68040 BSDL DESCRIPTION (VERSION 2.2)

Revision List:

1. LOCK and LOCKE controlled by io.1 vice io.0 (4D98D).
3. No other changes to Version 2.1 BSDL.
2. Instruction opcodes changed for SAMPLE, SHUTDOWN, and BYPASS.
3. New instructions DRVCTL.T, DRVCTL.S and PRIVATE added.
4. New instructions DRVCTL.T and DRVCTL.S renamed to DRVCTL_T and DRVCTL_S for syntax compatibility.
5. Register access specified for DRVCTL_T, DRVCTL_S, and PRIVATE instructions.
6. No other changes to Version 1.0 BSDL.

Package Type: 18 x 18 PGA

This BSDL is for the newer MC68040 mask sets of E26A and after (roughly after the second half of 1992). It does not include the 0.8-µm mask sets D43B, D50D, and D98D. For MC68LC040 and MC68EC040, two pin names have changed. To make the necessary modifications, change all occurrences of DLE to JS0 and `MDIS` to JS1.

```
entity MC68040 is

generic(PHYSICAL_PIN_MAP:string := "PGA_18x18");

port  (TDI:     in          bit;
       TDO:     out         bit;
       TMS:     in          bit;
       TCK:     in          bit;
       TRST:    in          bit;
       RSTO:    buffer      bit;
       IPEND:   buffer      bit;
       CIOUT:   out         bit;
       UPA:     out         bit_vector(0 to 1);
       TT:      inout       bit_vector(0 to 1);
       A:       inout       bit_vector(0 to 31);
       D:       inout       bit_vector(0 to 31);
       LOCKE:   out         bit;
       LOCK:    out         bit;
       R_W:     inout       bit;
       TLN:     out         bit_vector(0 to 1);
       TM:      out         bit_vector(0 to 2);
       SIZ:     inout       bit_vector(0 to 1);
       MI:      buffer      bit;
       BR:      buffer      bit;
       TS:      inout       bit;
       BB:      inout       bit;
       TIP:     out         bit;
       PST:     buffer      bit_vector(0 to 3);
       TA:      inout       bit;
       TEA:     in          bit;
       BG:      in          bit;
       SC:      in          bit_vector(0 to 1);
       TBI:     in          bit;
       AVEC:    in          bit;
       TCI:     in          bit;
```

JTAG Timing Specifications (All Operating Frequencies)

Num	Characteristic	Min	Max	Unit
	TCK Frequency of Operation	0	10	MHz
1	TCK Cycle Time	100	—	ns
2	TCK Clock Pulse Width Measured at 1.5 V	40	—	ns
3	TCK Rise and Fall Times	0	10	ns
4	TRST Setup Time to TCK Falling Edge	40	—	ns
5	TRST Assert Time	100	—	ns
6	Boundary Scan Input Data Setup Time	50	—	ns
7	Boundary Scan Input Data Hold Time	50	—	ns
8	TCK to Output Data Valid	0	50	ns
9	TCK to Output High Impedance	0	50	ns
10	TMS, TDI Data Setup Time	20	—	ns
11	TMS, TDI Data Hold Time	5	—	ns
12	TCK to TDO Data Valid	0	20	ns
13	TCK to TDO High Impedance	0	20	ns

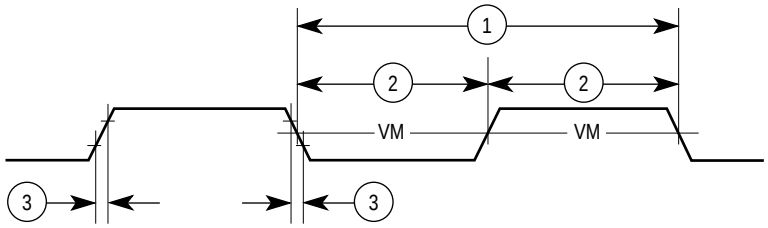


Figure 6-8. Clock Input Timing Diagram

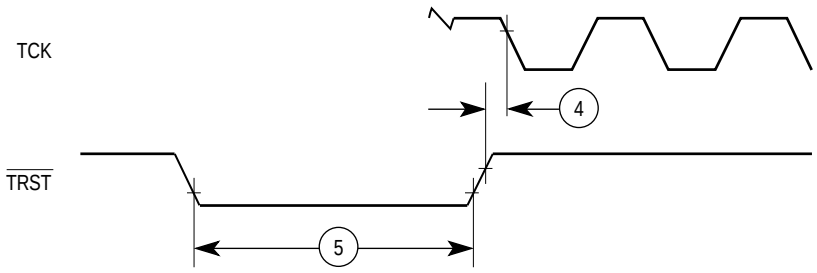
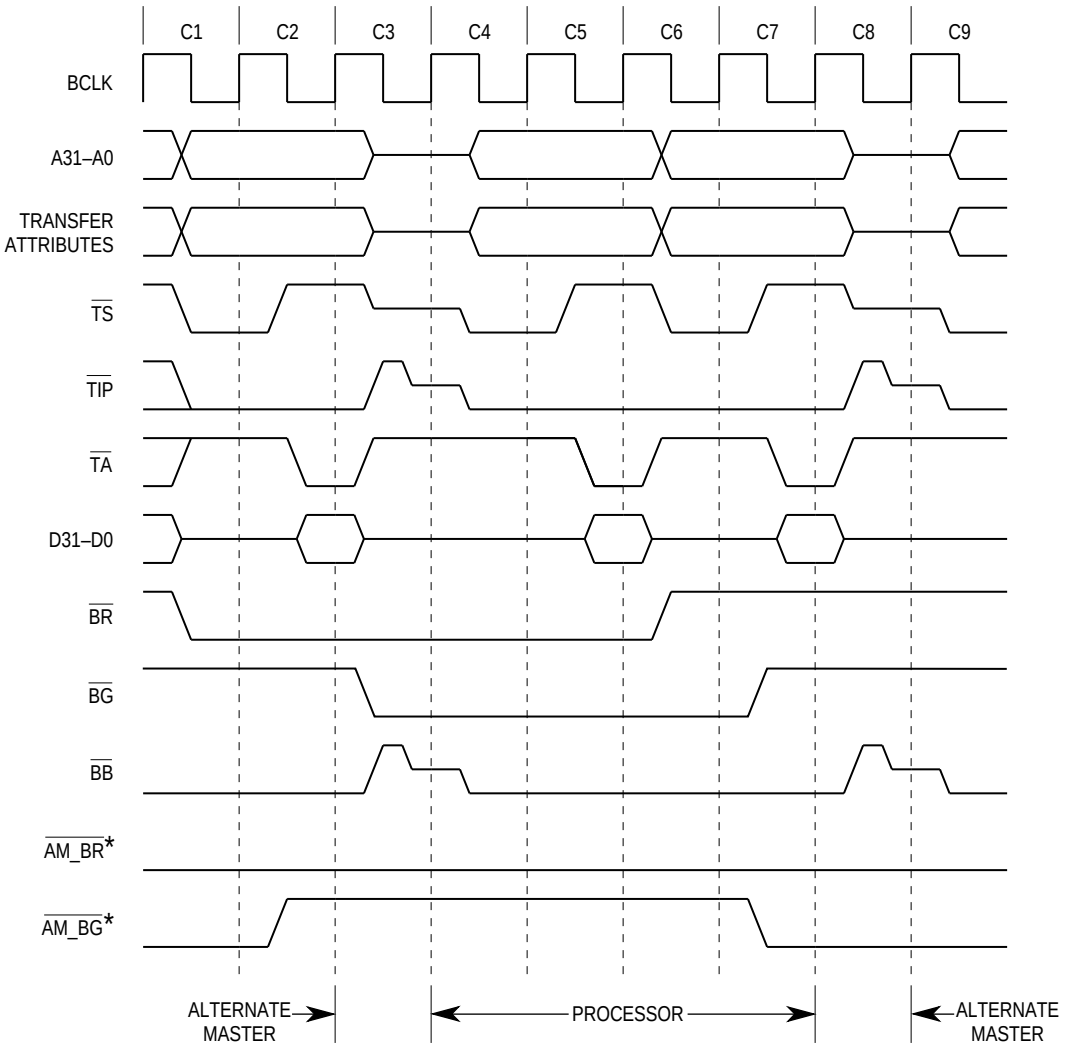


Figure 6-9. TRST Timing Diagram

require a pullup resistor to maintain a logic-one level between bus master tenures. The alternate bus master should negate these signals before three-stating to minimize rise time of the signals and ensure that the processor recognizes the correct level on the next BCLK rising edge. At the end of C3, the processor recognizes the bus grant and bus idle conditions ($\overline{BG}$ asserted and $\overline{BB}$ negated) and assumes ownership of the bus by asserting $\overline{BB}$ and immediately beginning a bus cycle during C4. During C6, the processor begins the second bus cycle for the misaligned operand and negates $\overline{BR}$ since no other accesses are pending. During C7, the external bus arbiter grants the bus back to the alternate bus master that is waiting for the processor to relinquish the bus. The processor negates $\overline{BB}$ and $\overline{TIP}$ before three-stating these and all other bus signals during C8. Finally, the alternate bus master recognizes the bus grant and idle conditions at the end of C8 and is able to resume bus activity during C9.



*AM indicates the alternate bus master.

Figure 7-32. Processor Bus Request Timing

reset or a restore operation of the null state sets FP7–FP0 to positive, nonsignaling not-a-numbers (NANs).

9.2.2 Floating-Point Control Register (FPCR)

The FPCR (see Figure 9-2) contains an exception enable (ENABLE) byte that enables or disables traps for each class of floating-point exceptions and a mode control (MODE) byte that sets the user-selectable modes. The user can read or write to the FPCR. Motorola reserves bits 31–16 for future definition; these bits are always read as zero and are ignored during write operations. The reset function or a restore operation of the null state clears the FPCR. When cleared, this register provides the IEEE 754 standard defaults.

9.2.2.1 EXCEPTION ENABLE BYTE. Each bit of the ENABLE byte (see Figure 9-2) corresponds to a floating-point exception class. The user can separately enable traps for each class of floating-point exceptions.

9.2.2.2 MODE CONTROL BYTE. The MODE byte (see Figure 9-2) controls the user-selectable rounding modes and precisions. Zeros in this byte select the IEEE 754 standard defaults. The rounding mode (RND) specifies how inexact results are rounded, and the rounding precision (PREC) selects the boundary for rounding the mantissa.

The processor supports four rounding modes specified by the IEEE 754 standard. These modes are: round to nearest (RN), round toward zero (RZ), round toward plus infinity (RP), and round toward minus infinity (RM). The RP and RM modes are directed rounding modes that are useful in interval arithmetic. Rounding is accomplished through the intermediate result. Single-precision results are rounded to a 24-bit boundary; double-precision results are rounded to a 53-bit boundary; and extended-precision results are rounded to a 64-bit boundary. Table 9-1 lists the encodings for the FPCR.

Table 9-1. Floating-Point Control Register Encodings

Rounding Mode (RND Field)	Encoding		Rounding Precision (PREC Field)
To Nearest (RN)	0	0	Extend (X)
Toward Zero (RZ)	0	1	Single (S)
Toward Minus Infinity (RM)	1	0	Double (D)
Toward Plus Infinity (RP)	1	1	Undefined

The following tie-case example illustrates how the 67-bit mantissa allows the FPU to meet the error bound of the IEEE specification:

Result	Integer	63-Bit Fraction	Guard	Round	Sticky
Intermediate	x	xxx…x00	1	0	0
Rounded-to-Nearest	x	xxx…x00	0	0	0

The least significant bit of the rounded result does not increment even though the guard bit is set in the intermediate result. The IEEE 754 standard specifies that tie cases should be handled in this manner. If the destination data format is extended and there is a difference between the infinitely precise intermediate result and the round-to-nearest result, the relative difference is 2^{-64} (the value of the guard bit). This error is equal to one-half of the least significant bit's value and is the worst case error that can be introduced when using the RN mode. Thus, the term one-half unit in the last place correctly identifies the error bound for this operation. This error specification is the relative error present in the result; the absolute error bound is equal to $2^{exponent} \times 2^{-64}$. The following example illustrates the error bound for the other rounding modes:

Result	Integer	63-Bit Fraction	Guard	Round	Sticky
Intermediate	x	xxx…x00	1	1	1
Rounded-to-Nearest	x	xxx…x00	0	0	0

The difference between the infinitely precise result and the rounded result is $2^{-64} + 2^{-65} + 2^{-66}$, which is slightly less than 2^{-63} (the value of the least significant bit). Thus, the error bound for this operation is not more than one unit in the last place. For all arithmetic operations, the FPU meets these error bounds, providing accurate and repeatable results.

9.5 POSTPROCESSING OPERATION

Most operations end with a postprocessing step. The FPU provides two steps in postprocessing. First, the condition code bits in the FPSR are set or cleared at the end of each arithmetic operation or move operation to a single floating-point data register. The condition code bits are consistently set based on the result of the operation. Second, the FPU supports 32 conditional tests that allow floating-point conditional instructions to test floating-point conditions in exactly the same way as the integer conditional instructions test the integer condition codes. The combination of consistently set condition code bits and the simple programming of conditional instructions gives the MC68040 a very flexible, high-performance method of altering program flow based on floating-point results. While reading the summary for each instruction, it should be assumed that an instruction performs postprocessing unless the summary specifically states that the instruction does not do so. The following paragraphs describe postprocessing in detail.

4. Examining the conditional predicate and setting the FPCC NAN bit accordingly prevents the exception from being taken again. This technique gives the most control since it is possible to pre-determine the direction of program flow. Bit 7 of the F-line operation word indicates where the conditional predicate is located. If bit 7 is set, the conditional predicate is the lower six bits of the F-line operation word. Otherwise, the conditional predicate is the lower six bits of the instruction word, which immediately follows the F-line operation word. Using the conditional predicate and the table for IEEE nonaware test in **9.5.2 Conditional Testing**, the condition codes can be set to return a known result indication when the conditional instruction is reexecuted.

Prior to exiting the user BSUN exception handler, the exception handler discards the floating-point state frame.

9.7.1.2 NONMASKABLE EXCEPTION CONDITIONS. There are no conditions.

9.7.2 Signaling Not-a-Number (SNAN)

An SNAN is used as an escape mechanism for a user-defined, non-IEEE data type. The processor never creates an SNAN as a result of an operation; a NAN created by an operand error exception is always a nonsignaling NAN. When an operand is an SNAN involved in an arithmetic instruction, the SNAN bit is set in the FPSR EXC byte. Since the FMOVEM, FMOVE FPCR, and FSAVE instructions do not modify the status bits, they cannot generate exceptions. Therefore, these instructions are useful for manipulating SNANs.

9.7.2.1 MASKABLE EXCEPTION CONDITIONS. When an SNAN is encountered, if the destination is a floating-point data register or is in memory (or an integer data register) and the format is single, double, or extended precision, the SNAN is maskable and may or may not take an exception.

a. If the user SNAN exception is disabled, the processor clears the SNAN bit in the NAN data format and the resulting nonsignaling NAN is transferred to the destination. No bits other than the SNAN bit of the NAN data format are modified, although the input NAN is truncated if necessary. Instruction execution continues without taking any exceptions.

b. If the user SNAN exception handler is enabled, the processor posts an exception and another floating-point instruction is eventually encountered; a pre-instruction exception is reported at that time. The SNAN entry in the processor's vector table points to the M68040FPSP SNAN exception handler. Once the M68040FPSP SNAN exception handler recognizes the operand error as a maskable condition, it does not modify the destination or pass control to the user SNAN exception handler.

9.7.2.2 NONMASKABLE EXCEPTION CONDITIONS. When an SNAN is encountered, if the destination is either in memory or an integer data register and the format is byte, word, or long word, a nonmaskable post-instruction exception occurs and is taken immediately. The SNAN entry in the processor's vector table points to the M68040FPSP SNAN exception handler.

inexact error to occur that is signaled as INEX1 exception. Furthermore, the subsequent divide could also produce an inexact result and cause INEX2 to be set in the FPCR EXC byte. Note that only one inexact exception vector number is generated by the processor. If either of the two inexact exceptions is enabled, the processor fetches the inexact exception vector, and the user INEX exception handler is initiated. INEX refers to both exceptions in the following paragraphs.

The INEX2 exception is the condition that exists when any operation, except the input of a packed decimal number, creates a floating-point intermediate result whose infinitely precise mantissa has too many significant bits to be represented exactly in the selected rounding precision or in the destination data format. If this condition occurs, the INEX2 bit is set in the FPSR EXC byte, and the infinitely precise result is rounded. Table 9-15 lists these rounding mode values.

Table 9-15. Divide by Zero Rounding Mode Values

Rounding Mode	Result
RN	The representable value nearest to the infinitely precise intermediate value is the result. If the two nearest representable values are equally near (a tie), then the one with the least significant bit equal to zero (even) is the result. This is sometimes referred to as “round nearest, even.”
RZ	The result is the value closest to and no greater in magnitude than the infinitely precise intermediate result. This is sometimes referred to as the “chip mode,” since the effect is to clear the bits to the right of the rounding point.
RM	The result is the value closest to and no greater than the infinitely precise intermediate result (possibly minus infinity).
RP	The result is the value closest to and no less than the infinitely precise intermediate result (possibly plus infinity).

The INEX1 and INEX2 exceptions are always maskable. Therefore, any INEX exception goes directly to the user INEX exception handler. The M68040FPSP does not provide any special handling for the INEX exception. When an INEX2 or INEX1 bit in the FPSR EXC byte is set, the processor stores the rounded result (listed in Table 9-15), to the destination. The FPCR MODE byte determines the rounding mode, and the PREC byte determines the rounding precision if the destination is a floating-point data register. Otherwise, if the destination is memory or an integer data register, the destination format determines the rounding precision. If one of the instructions has a forced precision, the instruction determines the rounding precision. If the INEX2 or INEX1 condition exists and if the corresponding INEX bit in the FPCR ENABLE byte is set, then the user INEX exception handler is taken.

a. If the user INEX exception handler is disabled, result is rounded and normal processing continues.

b. If the user INEX exception handler is enabled, the exception is taken. The INEX entry in the processor’s vector table points to the user INEX exception handler.

The user INEX exception handler must execute an FSAVE as its first floating-point instruction. At this point, the destination contains the rounding mode values as listed in

10.5 MISCELLANEOUS INTEGER UNIT INSTRUCTION TIMINGS

Instruction	Condition	<ea> Calculate	Execute
ABCD	Dy,Dx –(Ay),–(Ax)	1 3	3 1_L + 3
ADDX	Dy,Dx –(Ay),–(Ax)	1 3	1 1_L + 2
ANDI #<xxx>,CCR	—	1	4
ANDI #<xxx>,SR[a]	—	9	1_L + 8
Bcc	Branch Taken Branch Not Taken	2 3	2 3
BRA	Branch Taken Branch Not Taken	2 3	2 3
BSR <offset>	—	2	1_L + 1
CAS2[b]	True False	56 51	6_L + 49 6_L + 44
CMPM	—	3	1_L + 2
DBcc[c]	False, Count > –1 False, Count = –1 True	3 4 4	3 4 4
EORI #<xxx>,CCR	—	1	4
EORI #<xxx>,SR[a]	—	9	1_L + 8
EXG	Dy,Dx Ay,Ax Dy,Ax	1 2 1	1 1_L + 1 1
EXT	Word Long Word	1 1	2 1
EXTB	Long Word	1	1
ILLEGAL[a]	A-Line Unimplemented F-Line Unimplemented	16 16	16 16
LINK	—	3	2_L + 1
MOVE USP	USP,An An,USP[a]	3 7	2_L + 1 1_L + 6
MOVE16[c,d]	(Ax)+,(Ay)+ xxx.L,(An) xxx.L,(An)+ (An),xxx.L (An)+,xxx.L	6 4 5 4 4	1_L+ 7 7 8 7 7
MOVEC[b]	Rn,Rc Rc,Rn	7 11	1_L + 6 1_L + 10
MOVEP[c]	MOVEP.W Dn,d16(An) MOVEP.L Dn,d16(An) MOVEP.W d16(An),Dn MOVEP.L d16(An),Dn	11 13 4 8	2_L + 9 2_L + 11 2_L + 5 2_L + 8
MOVEQ	—	1	1
NOP[a]	—	8	1_L + 7

10.5 MISCELLANEOUS INTEGER UNIT INSTRUCTION TIMINGS (Continued)

Instruction	Condition	<ea> Calculate	Execute
ORI #<xxx>,CCR	—	1	4
ORI #<xxx>,SR[a]	—	9	1_L + 8
PACK	Dx,Dy,#<xxx> –(Ay),–(Ax),#<xxx>	1 3	3 2_L + 3
PFLUSH[b]	—	11	1_L + 10
PFLUSHA[b]	—	11	1_L + 10
PFLUSHAN[b]	—	27	1_L + 26
PFLUSHN (An)[b]	—	11	1_L + 10
PTESTR, PTESTW[e]	—	25	11_L + 14
RESET[a]	—	521	521
RTD[c]	—	6	1_L + 5
RTE[a]	Stack Format $0 Stack Format $1 Stack Format $2 Stack Format $3 Stack Format $4 Stack Format $7	2 4 2 3 2 4	13 23 14 20 15 23
RTR[c]	—	7	1_L + 6
RTS[c]	—	5	5
SBCD	Dy,Dx –(Ay),–(Ax)	1 3	3 1_L + 3
SUBX	Dy,Dx –(Ay),–(Ax)	1 3	1 1_L + 2
SWAP	—	1	2
TRAP#[a]	—	16	16
TRAPcc[f]	Taken Not Taken	19 5	19 5
TRAPV[f]	Taken Not Taken	19 5	19 5
UNLK	—	2	1_L + 1
UNPK	Dx,Dy,# –(Ay),–(Ax),#	1 3	4 2_L + 4

NOTES:

a. Times listed are minimum. This instruction interlocks the <ea> calculate and execute stages and synchronizes some portions of the processor before execution.
b. Times listed are typical. This instruction interlocks the <ea> calculate and execute stages and synchronizes some portions of the processor before execution.
c. This instruction interlocks the <ea> calculate and execute stages.
d. Successive in-line MOVE16 instructions each add eight clocks to the <ea> calculate and execute times.
e. Typical measurement for three-level table search with no descriptor writes, no entries cached, and four-clock memory access times.
f. This instruction interlocks the <ea> calculate and execute stages. For the exception taken, this instruction also synchronizes some portions of the processor before execution; times listed are minimum in this case.

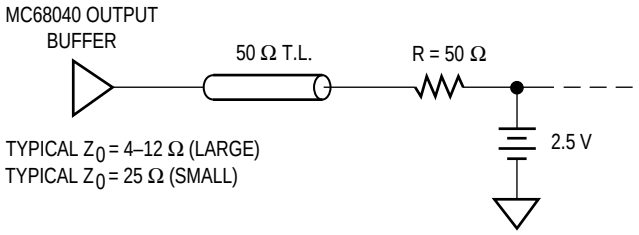


Figure 11-8. MC68040 Termination Network

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.

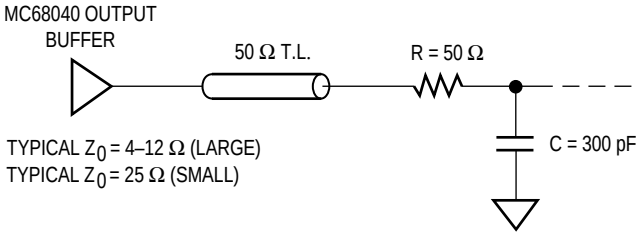


Figure 11-9. Typical Configuration for RC Termination Network

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

A.2 INTERRUPT PRIORITY LEVEL ($\overline{IPL2}$–$\overline{IPL0}$)

The $\overline{IPL2}$–$\overline{IPL0}$ pins do not have any affect on the selection of output buffer impedance.

A.3 JTAG SCAN (JS0)

The MC68040 DLE pin name has been changed to JS0. During normal operation, the JS0 pin cannot float, it must be tied to GND or Vcc directly or through a resistor. During board testing, this pin retains the functionality of the JTAG scan of the MC68040 for compatibility purposes. Refer to **Section 6 IEEE 1149.1A Test Access Port (JTAG)** for details concerning IEEE *1149.1 Standard Test Access Port and Boundary Scan Architecture*.

A.4 DATA LATCH AND MULTIPLEXED BUS MODES

The MC68LC040 does not implement the data latch or multiplexed modes of operation. The $\overline{CDIS}$ pin is ignored at the rising edge of reset. All timing and drive capabilities of the MC68LC040 are equivalent to those of the MC68040 in small output buffer impedance mode.

A.5 FLOATING-POINT UNIT (FPU)

The FPU is not implemented on the MC68LC040. All floating-point instructions cause an unimplemented floating-point exception to be taken with a new eight-word stack frame (format $4). The stack frame contains the status register (SR), program counter (PC), vector offset, effective address of the operand (where applicable), and PC value of the unimplemented floating-point instruction.

A.5.1 Unimplemented Floating-Point Instructions and Exceptions

All legal MC68040 and MC68881/MC68882 floating-point instructions are defined as unimplemented floating-point instructions on the MC68LC040. These instructions generate a format $4 stack frame during exception processing before taking an F-line exception. These instructions trap as an F-line exception, and the F-line exception handler can emulate them in software to maintain user-object-code compatibility.

The MC68LC040 assists the emulation process by distinguishing unimplemented floating-point instructions from other unimplemented F-line instructions. To aid emulation, the effective address is calculated and saved in the format $4 stack frame. This simplifies and speeds up the emulation process by eliminating the need for the emulation routine to determine the effective address and by providing information required to emulate the instruction on the exception stack frame in the supervisor address space. However, the floating-point instruction can reside in user space; therefore, the floating-point unimplemented exception handler may need to access user instruction space. The following processing steps occur for an unimplemented floating-point instruction:

1. When an unimplemented floating-point instruction is encountered, the instruction is partially decoded, and the effective address is calculated, if required.
2. The processor waits for all previous integer instructions, write-backs, and associated exception processing to complete before beginning exception processing for the unimplemented floating-point instruction. Any access error that occurs in completing the write-backs causes an access error exception, and the resulting stack frame indicates

A.6.8 Input AC Timing Specifications (See Figures A-5 to A-9)

Num	Characteristic	20 MHz		25 MHz		33 MHz		40 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
15	Data-In Valid to BCLK (Setup)	6	—	5	—	4	—	3	—	nS
16	BCLK to Data-In Invalid (Hold)	5	—	4	—	4	—	3	—	nS
17	BCLK to Data-In High Impedance (Read Followed by Write)	—	61	—	49	—	36.5	—	30.25	nS
22a	$\overline{TA}$ Valid to BCLK (Setup)	12.5	—	10	—	10	—	8	—	nS
22b	$\overline{TEA}$ Valid to BCLK (Setup)	12.5	—	10	—	10	—	9	—	nS
22c	$\overline{TCI}$ Valid to BCLK (Setup)	12.5	—	10	—	10	—	9	—	nS
22d	$\overline{TBI}$ Valid to BCLK (Setup)	14	—	11	—	10	—	9	—	nS
23	BCLK to $\overline{TA}$, $\overline{TEA}$, $\overline{TCI}$, $\overline{TBI}$ Invalid (Hold)	2.5	—	2	—	2	—	2	—	nS
24	$\overline{AVEC}$ Valid to BCLK (Setup)	6	—	5	—	5	—	5	—	nS
25	BCLK to $\overline{AVEC}$ Invalid (Hold)	2.5	—	2	—	2	—	2	—	nS
41a	$\overline{BB}$ Valid to BCLK (Setup)	8	—	7	—	7	—	7	—	nS
41b	$\overline{BG}$ Valid to BCLK (Setup)	10	—	8	—	7	—	7	—	nS
41c	$\overline{CDIS}$, $\overline{MDIS}$ Valid to BCLK (Setup)	12.5	—	10	—	8	—	8	—	nS
41d	$\overline{IPLA}$ Valid to BCLK (Setup)	5	—	4	—	3	—	3	—	nS
42	BCLK to $\overline{BB}$, $\overline{BG}$, $\overline{CDIS}$, $\overline{MDIS}$, $\overline{IPLA}$ Invalid (Hold)	2.5	—	2	—	2	—	2	—	nS
44a	Address Valid to BCLK (Setup)	10	—	8	—	7	—	7	—	nS
44b	SIZx Valid to BCLK (Setup)	15	—	12	—	8	—	8	—	nS
44c	TTx Valid to BCLK (Setup)	7.5	—	6	—	8.5	—	8.5	—	nS
44d	R/$\overline{W}$ Valid to BCLK (Setup)	7.7	—	6	—	5	—	5	—	nS
44e	SCx Valid to BCLK (Setup)	12.5	—	10	—	11	—	8	—	nS
45	BCLK to Address SIZx, TTx, R/$\overline{W}$, SCx Invalid (Hold)	2.5	—	2	—	2	—	2	—	nS
46	$\overline{TS}$ Valid to BCLK (Setup)	6	—	5	—	9	—	7	—	nS
47	BCLK to $\overline{TS}$ Invalid (Hold)	2.5	—	2	—	2	—	2	—	nS
49	BCLK to $\overline{BB}$ High Impedance (MC68LC040 Assumes Bus Mastership)	—	11	—	9	—	9	—	9	nS
51	$\overline{RSTI}$ Valid to BCLK	6	—	5	—	4	—	4	—	nS
52	BCLK to $\overline{RSTI}$ Invalid	2.5	—	2	—	2	—	2	—	nS

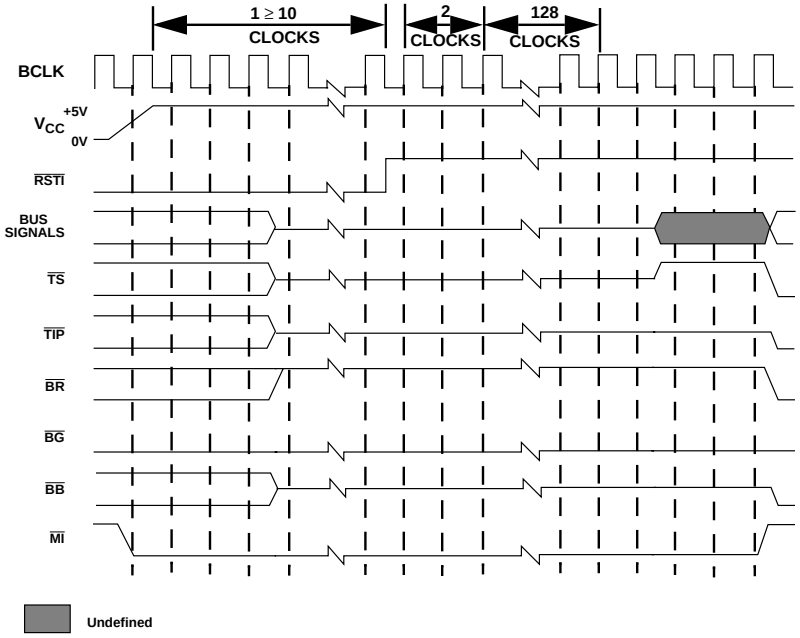


Figure B-5. MC68EC040 Initial Power-On Reset Timing

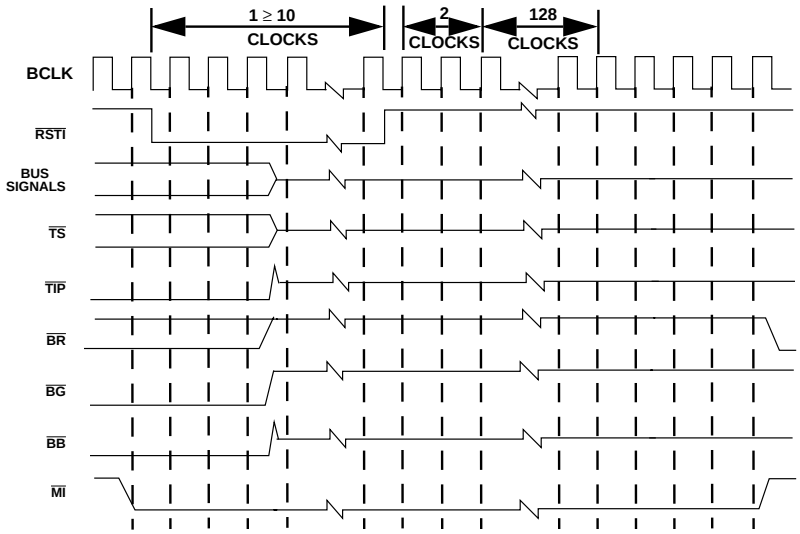


Figure B-6. MC68EC040 Normal Reset Timing

When a RESET instruction is executed, the processor drives the reset out ($\overline{\text{RSTO}}$) signal for 512 BCLK cycles. In this case, the processor resets the external devices of the system, and the internal registers of the processor are unaffected. The external devices connected to

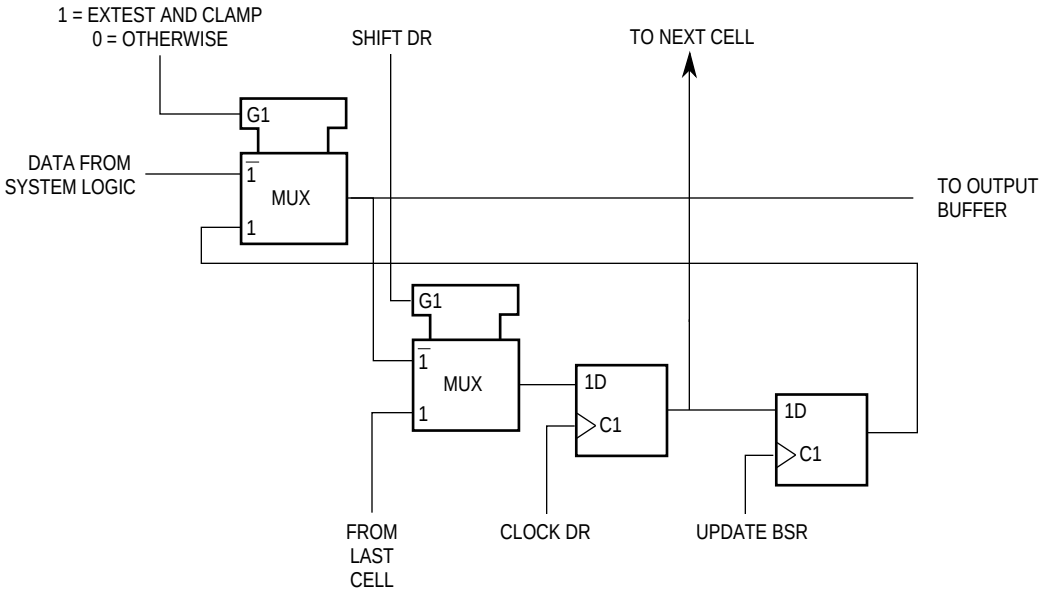


Figure C-6. Output Latch Cell (O.Latch)

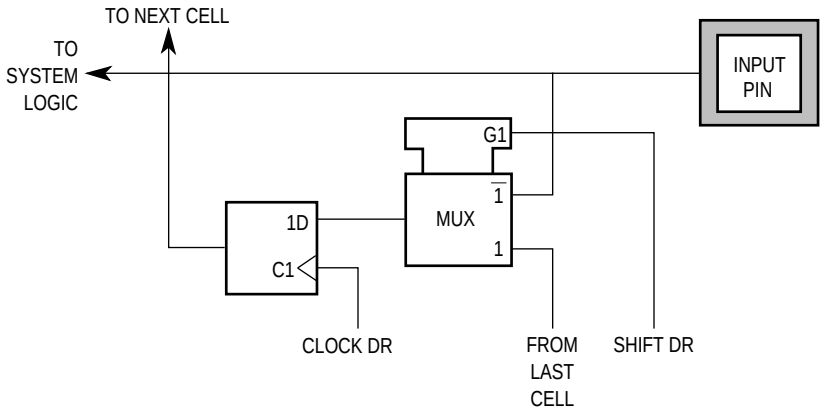


Figure C-7. Input Pin Cell (I.Pin)

–C–

Cache, 1-4, 2-8
  Burst Mode Operations, 4-11
  Data, 2-3, 2-8, 3-1, 3-12, 7-44, 8-7, 8-18
  Exceptions, 8-7, 8-18
  Instruction Prefetches, 4-13
  Instruction, 3-1, 8-7, 8-18
  Misaligned Accesses, 4-11
  Page Descriptors, 4-5
  Replacement Algorithm, 4-4
  Retry Operation, 4-12
  Shared Data, 4-9, 4-10
Cache Coherency, 4-10
Cache Controller, 3-2, 3-28, 4-4, 4-8, 4-12
Cache Inhibited, *see* Caching Modes
Cache Line, 4-3
  D-Bit, 4-6
  Dirty, 4-3
  Format, 4-2
  Invalid, 4-3; Timing, 10-8
  V-Bit, 4-3
  Valid, 4-3
Caching Modes, 4-6
  Cache Inhibited, 4-7
  Copyback, 4-6, 7-60
  Default, 4-6
  Nonserialized, 4-6
  Serialized, 4-6
  Write-Through, 4-6
Caching Operation, 4-3
Calculate Stage, *see* Integer Unit Pipeline
CM Field, 4-6, 5-8, *see also* Descriptors
Conditional Branch, 7-50
Conditional Tests, 9-15, 9-17
  Floating-Point IEEE Tests, 9-17, 9-18, 9-25
  Unordered Conditions, 9-17, 9-18
Control Signals, 7-1, 7-9
Copyback, *see* Caching Modes

–D–

Data Bus, 7-1, 7-3
Data Format, 1-9, 9-7
  Extended Precision, 9-12, 9-21, 9-23, 9-24
  Floating-Point Conversion of, 9-12
  Packed Decimal Real, 9-22
Data Latch Enable (DLE) Mode, 1-2, 5-5, 5-14, 7-70, A-5
Data Registers, 2-4
Data Types, 9-7
  Denormalized Numbers, 1-9, 9-12, 9-22, 9-23, 9-16
  Infinities, 1-9
  NANs, 1-9, 9-17
  Normalized Numbers, 1-9, 9-16, 9-33
  Unnormalized Numbers, 9-12, 9-22, 9-23
  Zeros, 1-9
Decode Stage, *see* Integer Unit Pipeline
Demand Memory, 3-1
Denormalized Numbers, *see* Data Types
Descriptors, 3-8, 3-12
  CM Field, 4-6, 5-8
  Field Definitions, 3-13
  Indirect, 3-9, 3-14; PDT Field, 3-17
  Invalid, 3-9, 3-14
  M-Bit, 3-21
  Page, 3-12, 3-13, 3-17, 3-23, 3-24, 4-5
  Resident, 3-14
  S-Bits, 3-23
  Table, 3-12, 3-13, 3-24; UDT Field, 3-19
  U-Bit, 3-21
  W-Bits, 3-24
Direct Memory Access (DMA), 7-56
Dirty Data, 4-1, 5-8
Disabling JTAG, 6-13
Disregard Request Condition, 7-55
Double Bus Fault, 7-43, 8-8, 8-18
DRVCTL.T, 6-3, 6-12
Dynamic Bus Sizing, 7-3

–E–

Effective Address (<ea>), 2-3
Execute Stage, *see* Integer Unit Pipeline
Exception Handler, 8-4
Exception Processing, 1-6, 2-5, 7-36, 7-37, 7-43, A-6
Exception Vector, 2-7
  Table, 8-1, 8-4
Exceptions
  Access Error, 1-5, 3-23, 3-24, 5-14, 7-37, 7-43, 9-21, A-6
  Access Fault, 3-9, 8-6, 8-7
  Address Error, 7-6, 7-43, 8-8