Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	Coldfire V3
Core Size	32-Bit Single-Core
Speed	90MHz
Connectivity	EBI/EMI, I²C, UART/USART
Peripherals	DMA, POR, WDT
Number of I/O	16
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 3.6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	208-BFQFP
Supplier Device Package	208-FQFP (28x28)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mcf5307cai90b

CONTENTS

TABLES

Table 2-6. Notational Conventions (Continued)

Instruction	Operand Syntax
P	Branch prediction
C	Carry
N	Negative
V	Overflow
X	Extend
Z	Zero

2.6.1 Instruction Set Summary

Table 2-7 lists implemented user-mode instructions by opcode.

Table 2-7. User-Mode Instruction Set Summary

Instruction	Operand Syntax	Operand Size	Operation
ADD	Dy,<ea>x <ea>y,Dx	.L .L	Source + destination → destination
ADDA	<ea>y,Ax	.L	Source + destination → destination
ADDI	#<data>,Dx	.L	Immediate data + destination → destination
ADDQ	#<data>,<ea>x	.L	Immediate data + destination → destination
ADDX	Dy,Dx	.L	Source + destination + X → destination
AND	Dy,<ea>x <ea>y,Dx	.L .L	Source & destination → destination
ANDI	#<data>,Dx	.L	Immediate data & destination → destination
ASL	Dy,Dx #<data>,Dx	.L .L	X/C ← (Dx << Dy) ← 0 X/C ← (Dx << #<data>) ← 0
ASR	Dy,Dx #<data>,Dx	.L .L	MSB → (Dx >> Dy) → X/C MSB → (Dx >> #<data>) → X/C
Bcc	<label>	.B,.W	If condition true, then PC + 2 + d_n → PC
BCHG	Dy,<ea>x #<data>,<ea-1>x	.B,.L .B,.L	~(<bit number> of destination) → Z, Bit of destination
BCLR	Dy,<ea>x #<data>,<ea-1>x	.B,.L .B,.L	~(<bit number> of destination) → Z; 0 → bit of destination
BRA	<label>	.B,.W	PC + 2 + d_n → PC
BSET	Dy,<ea>x #<data>,<ea-1>x	.B,.L .B,.L	~(<bit number> of destination) → Z; 1→ bit of destination
BSR	<label>	.B,.W	SP – 4 → SP; next sequential PC→ (SP); PC + 2 + d_n → PC
BTST	Dy,<ea>x #<data>,<ea-1>x	.B,.L .B,.L	~(<bit number> of destination) → Z
CLR	<ea>y,Dx	.B,.W,.L	0 → destination
CMP	<ea>y,Ax	.L	Destination – source
CMPA	<ea>y,Dx	.L	Destination – source

Table 2-20. Fault Status Encodings

FS[3–0]	Definition
0000	Not an access or address error
0001-001x	Reserved
0100	Error on instruction fetch
0101–011x	Reserved
1000	Error on operand write
1001	Attempted write to write-protected space
101x	Reserved
1100	Error on operand read
1101–111x	Reserved

- Vector number—This 8-bit field, vector[7–0], defines the exception type. It is calculated by the processor for internal faults and is supplied by the peripheral for interrupts. See Table 2-18.

2.8.2 Processor Exceptions

Table 2-21 describes MCF5307 exceptions.

Table 2-21. MCF5307 Exceptions

Exception	Description
Access Error	Access errors are reported only in conjunction with an attempted store to write-protected memory. Thus, access errors associated with instruction fetch or operand read accesses are not possible.
Address Error	Caused by an attempted execution transferring control to an odd instruction address (that is, if bit 0 of the target address is set), an attempted use of a word-sized index register (Xi.w) or a scale factor of 8 on an indexed effective addressing mode, or attempted execution of an instruction with a full-format indexed addressing mode.
Illegal Instruction	On Version 2 ColdFire implementations, only some illegal opcodes were decoded and generated an illegal instruction exception. The Version 3 processor decodes the full 16-bit opcode and generates this exception if execution of an unsupported instruction is attempted. Additionally, attempting to execute an illegal line A or line F opcode generates unique exception types: vectors 10 and 11, respectively. ColdFire processors do not provide illegal instruction detection on extension words of any instruction, including MOVEC. Attempting to execute an instruction with an illegal extension word causes undefined results.
Divide by Zero	Attempted division by zero causes an exception (vector 5, offset = 0x014) except when the PC points to the faulting instruction (DIVU, DIVS, REMU, REMS).
Privilege Violation	Caused by attempted execution of a supervisor mode instruction while in user mode. The ColdFire Programmer's Reference Manual lists supervisor- and user-mode instructions.

Normally, cache-inhibited reads bypass the cache and are performed on the external bus. The exception to this normal operation occurs when all of the following conditions are true during a cache-inhibited read:

- The cache-inhibited fill buffer bit, CACR[DNFB], is set.
- The access is an instruction read.
- The access is normal (that is, transfer type (TT) equals 0).

In this case, an entire line is fetched and stored in the fill buffer. It remains valid there, and the cache can service additional read accesses from this buffer until either another fill or a cache-invalidate-all operation occurs.

Valid cache entries that match during cache-inhibited address accesses are neither pushed nor invalidated. Such a scenario suggests that the associated cache mode for this address space was changed. To avoid this, it is generally recommended to use the CPUSHL instruction to push or invalidate the cache entry or set CACR[CINVA] to invalidate the cache before switching cache modes.

4.9.1 Caching Modes

For every memory reference generated by the processor or debug module, a set of effective attributes is determined based on the address and the ACRs. Caching modes determine how the cache handles an access. An access can be cacheable in either write-through or copyback mode; it can be cache-inhibited in precise or imprecise modes. For normal accesses, the ACR*n*[CM] bit corresponding to the address of the access specifies the caching modes. If an address does not match an ACR, the default caching mode is defined by CACR[DCM]. The specific algorithm is as follows:

```
if (address == ACR0-address including mask)
        effective attributes = ACR0 attributes
else if (address == ACR1-address including mask)
                effective attributes = ACR1 attributes
        else effective attributes = CACR default attributes
```

Addresses matching an ACR can also be write-protected using ACR[W]. Addresses that do not match either ACR can be write-protected using CACR[DW].

Reset disables the cache and clears all CACR bits. As shown in Figure 4-4, reset does not automatically invalidate cache entries; they must be invalidated through software.

The ACRs allow the defaults selected in the CACR to be overridden. In addition, some instructions (for example, CPUSHL) and processor core operations perform accesses that have an implicit caching mode associated with them. The following sections discuss the different caching accesses and their associated cache modes.

4.9.1.1 Cacheable Accesses

If ACR*n*[CM] or the default field of the CACR indicates write-through or copyback, the access is cacheable. A read access to a write-through or copyback region is read from the

Chapter 5
Debug Support

This chapter describes the Revision B enhanced hardware debug support in the MC5307. This revision of the ColdFire debug architecture encompasses the earlier revision.

5.1 Overview

The debug module is shown in Figure 5-1.

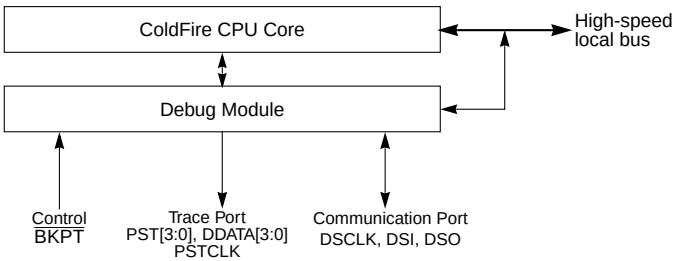


Figure 5-1. Processor/Debug Module Interface

Debug support is divided into three areas:

- Real-time trace support—The ability to determine the dynamic execution path through an application is fundamental for debugging. The ColdFire solution implements an 8-bit parallel output bus that reports processor execution status and data to an external emulator system. See Section 5.3, “Real-Time Trace Support.”
- Background debug mode (BDM)—Provides low-level debugging in the ColdFire processor complex. In BDM, the processor complex is halted and a variety of commands can be sent to the processor to access memory and registers. The external emulator uses a three-pin, serial, full-duplex channel. See Section 5.5, “Background Debug Mode (BDM),” and Section 5.4, “Programming Model.”
- Real-time debug support—BDM requires the processor to be halted, which many real-time embedded applications cannot do. Debug interrupts let real-time systems execute a unique service routine that can quickly save the contents of key registers and variables and return the system to normal operation. The emulator can access saved data because the hardware supports concurrent operation of the processor and BDM-initiated commands. See Section 5.6, “Real-Time Debug Support.”

Table 5-3. BDM/Breakpoint Registers

DRc[4–0]	Register Name	Abbreviation	Initial State	Page
0x00	Configuration/status register	CSR	0x0010_0000	p. 5-10
0x01–0x04	Reserved	—	—	—
0x05	BDM address attribute register	BAAR	0x0000_0005	p. 5-9
0x06	Address attribute trigger register	AATR	0x0000_0005	p. 5-7
0x07	Trigger definition register	TDR	0x0000_0000	p. 5-14
0x08	Program counter breakpoint register	PBR	—	p. 5-13
0x09	Program counter breakpoint mask register	PBMR	—	p. 5-13
0x0A–0x0B	Reserved	—	—	—
0x0C	Address breakpoint high register	ABHR	—	p. 5-8
0x0D	Address breakpoint low register	ABLR	—	p. 5-8
0x0E	Data breakpoint register	DBR	—	p. 5-12
0x0F	Data breakpoint mask register	DBMR	—	p. 5-12

NOTE:

Debug control registers can be written by the external development system or the CPU through the WDEBUG instruction.

CSR is write-only from the programming model. It can be read or written through the BDM port using the RDMREG and WDMREG commands.

5.4.1 Address Attribute Trigger Register (AATR)

The address attribute trigger register (AATR) defines address attributes and a mask to be matched in the trigger. The register value is compared with address attribute signals from the processor's local high-speed bus, as defined by the setting of the trigger definition register (TDR).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	RM	SZM		TTM		TMM			R	SZ		TT		TM		
Reset	0000_0000_0000_0101															
R/W	Write only. AATR is accessible in supervisor mode as debug control register 0x06 using the WDEBUG instruction and through the BDM port using the WDMREG command.															
DRc[4–0]	0x06															

Figure 5-5. Address Attribute Trigger Register (AATR)

Table 5-4 describes AATR fields.

5.5.3.3.11 Write Control Register (WCREG)

The operand (longword) data is written to the specified control register. The write alters all 32 register bits.

Command/Result Formats:

	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Command	0x2	0x8	0x8	0x0
	0x0	0x0	0x0	0x0
	0x0	Rc		
Result	D[31:16]			
	D[15:0]			

Figure 5-38. WCREG Command/Result Formats

Command Sequence:

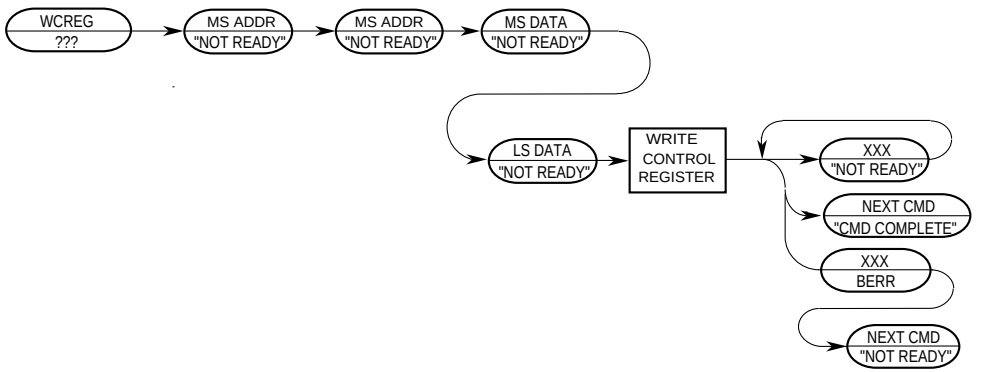


Figure 5-39. WCREG Command Sequence

Operand Data: This instruction requires two longword operands. The first selects the register to which the operand data is to be written; the second contains the data.

Result Data: Successful write operations return 0xFFFF. Bus errors on the write cycle are indicated by the setting of bit 16 in the status message and by a data pattern of 0x0001.

fetches a unique exception vector, 12, from the vector table.

Execution continues at the instruction address in the vector corresponding to the breakpoint triggered. All interrupts are ignored while the processor is in emulator mode. The debug interrupt handler can use supervisor instructions to save the necessary context such as the state of all program-visible registers into a reserved memory area.

When debug interrupt operations complete, the RTE instruction executes and the processor exits emulator mode. After the debug interrupt handler completes execution, the external development system can use BDM commands to read the reserved memory locations.

The generation of another debug interrupt during the first instruction after the RTE exits emulator mode is inhibited. This behavior is consistent with the existing logic involving trace mode where the first instruction executes before another trace exception is generated. Thus, all hardware breakpoints are disabled until the first instruction after the RTE completes execution, regardless of the programmed trigger response.

5.6.1.1 Emulator Mode

Emulator mode is used to facilitate non-intrusive emulator functionality. This mode can be entered in three different ways:

- Setting CSR[EMU] forces the processor into emulator mode. EMU is examined only if $\overline{\text{RSTI}}$ is negated and the processor begins reset exception processing. It can be set while the processor is halted before reset exception processing begins. See Section 5.5.1, “CPU Halt.”
- A debug interrupt always puts the processor in emulation mode when debug interrupt exception processing begins.
- Setting CSR[TRC] forces the processor into emulation mode when trace exception processing begins.

While operating in emulation mode, the processor exhibits the following properties:

- All interrupts are ignored, including level-7 interrupts.
- If CSR[MAP] = 1, all caching of memory and the SRAM module are disabled. All memory accesses are forced into a specially mapped address space signaled by TT = 0x2, TM = 0x5 or 0x6. This includes stack frame writes and the vector fetch for the exception that forced entry into this mode.

The RTE instruction exits emulation mode. The processor status output port provides a unique encoding for emulator mode entry (0xD) and exit (0x7).

5.6.2 Concurrent BDM and Processor Operation

The debug module supports concurrent operation of both the processor and most BDM commands. BDM commands may be executed while the processor is running, except those following operations that access processor/memory registers:

- Read/write address and data registers
- Read/write control registers

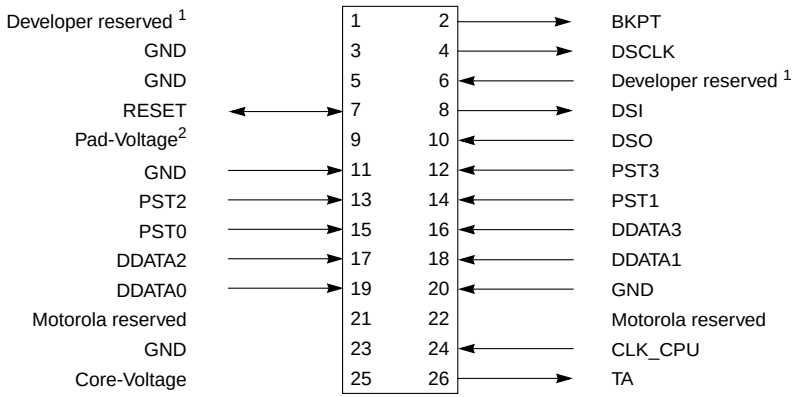
For BDM commands that access memory, the debug module requests the processor's local bus. The processor responds by stalling the instruction fetch pipeline and waiting for current bus activity to complete before freeing the local bus for the debug module to perform its access. After the debug module bus cycle, the processor reclaims the bus.

Breakpoint registers must be carefully configured in a development system if the processor is executing. The debug module contains no hardware interlocks, so TDR should be disabled while breakpoint registers are loaded, after which TDR can be written to define the exact trigger. This prevents spurious breakpoint triggers.

Because there are no hardware interlocks in the debug unit, no BDM operations are allowed while the CPU is writing the debug's registers (DSCLK must be inactive).

5.7 Motorola-Recommended BDM Pinout

The ColdFire BDM connector, Figure 5-44, is a 26-pin Berg connector arranged 2 x 13.



[1]Pins reserved for BDM developer use.
[2]Supplied by target

Figure 5-44. Recommended BDM Connector

5.8 Processor Status, DDATA Definition

This section specifies the ColdFire processor and debug module's generation of the processor status (PST) and debug data (DDATA) output on an instruction basis. In general, the PST/DDATA output for an instruction is defined as follows:

PST = 0x1, {PST = [0x89B], DDATA= operand}

where the {...} definition is optional operand information defined by the setting of the CSR.

Chapter 10
Chip-Select Module

This chapter describes the MCF5307 chip-select module, including the operation and programming model of the chip-select registers, which include the chip-select address, mask, and control registers.

10.1 Overview

The following list summarizes the key chip-select features:

- Eight independent, user-programmable chip-select signals ($\overline{CS}$[7:0]) that can interface with SRAM, PROM, EPROM, EEPROM, Flash, and peripherals
- Address masking for 64-Kbyte to 4-Gbyte memory block sizes
- Programmable wait states and port sizes
- External master access to chip selects

10.2 Chip-Select Module Signals

Table 10-1 lists signals used by the chip-select module.

Table 10-1. Chip-Select Module Signals

Signal	Description
Chip Selects ($\overline{CS}$[7:0])	Each $\overline{CSn}$ can be independently programmed for an address location as well as for masking, port size, read/write burst-capability, wait-state generation, and internal/external termination. Only $\overline{CS0}$ is initialized at reset when it acts as a global chip select that allows boot ROM to be at any defined address space. Port size and termination (internal versus external) and byte enables for $\overline{CS0}$ are configured by the logic levels of D[7:5] when $\overline{RSTI}$ negates.
Output Enable ($\overline{OE}$)	Interfaces to memory or to peripheral devices and enables a read transfer. It is asserted and negated on the falling edge of the clock. $\overline{OE}$ is asserted only when one of the chip selects matches for the current address decode.
Byte Enables/ Byte Write Enables ($\overline{BE}$[3:0]/ $\overline{BWE}$[3:0])	These multiplexed signals are individually programmed through the byte enable mode bit, CSCRn[BEM], described in Section 10.4.1.3, “Chip-Select Control Registers (CSCR0–CSCR7).” These generated signals provide byte data select signals, which are decoded from the transfer size, A1, and A0 signals in addition to the programmed port size and burstability of the memory accessed, as Table 10-2 shows.

Table 10-2 shows the interaction of the byte enable/byte-write enables with related signals.

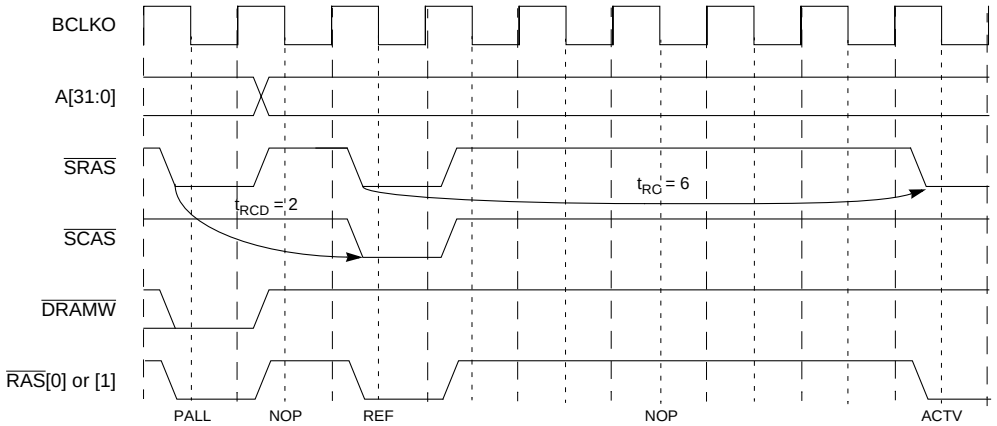


Figure 11-22. Auto-Refresh Operation

11.4.4.6 Self-Refresh Operation

Self-refresh is a method of allowing the SDRAM to enter into a low-power state, while at the same time to perform an internal refresh operation and to maintain the integrity of the data stored in the SDRAM. The DRAM controller supports self-refresh with DCR[IS]. When IS is set, the SELF command is sent to the SDRAM. When IS is cleared, the SELFX command is sent to the DRAM controller. Figure 11-23 shows the self-refresh operation.

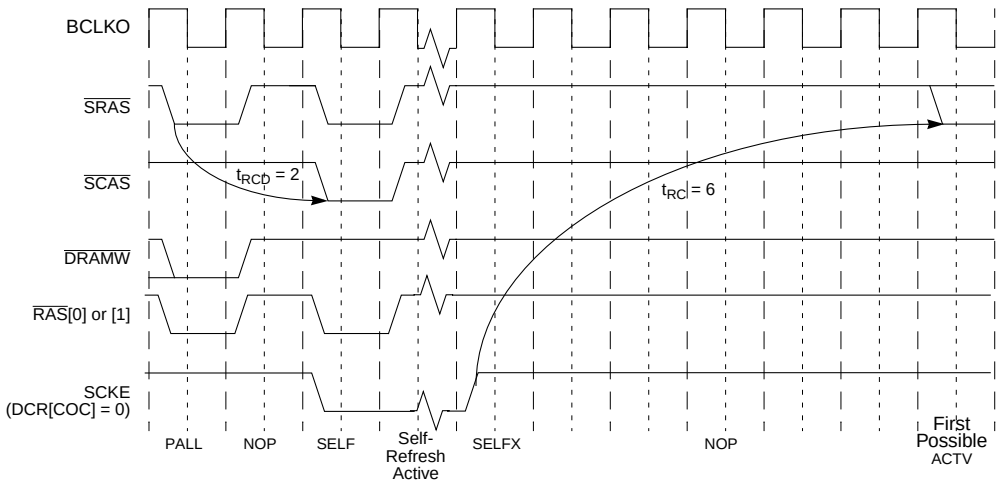


Figure 11-23. Self-Refresh Operation

Table 16-1. Pins 1–52 (Left, Top-to-Bottom) (Continued)

Pin		Alternate Function	I/O	Description	Drive (mA)
No	Name				
13	VCC	—	—	Power input	—
14	A8	—	I/O	Address bus bit	8
15	A9	—	I/O	Address bus bit	8
16	A10	—	I/O	Address bus bit	8
17	GND	—	—	Ground pin	—
18	A11	—	I/O	Address bus bit	8
19	A12	—	I/O	Address bus bit	8
20	A13	—	I/O	Address bus bit	8
21	VCC	—	—	Power input	—
22	A14	—	I/O	Address bus bit	8
23	A15	—	I/O	Address bus bit	8
24	A16	—	I/O	Address bus bit	8
25	GND	—	—	Ground pin	—
26	A17	—	I/O	Address bus bit	8
27	A18	—	I/O	Address bus bit	8
28	A19	—	I/O	Address bus bit	8
29	VCC	—	—	Power input	—
30	A20	—	I/O	Address bus bit	8
31	A21	—	I/O	Address bus bit	8
32	A22	—	I/O	Address bus bit	8
33	GND	—	—	Ground pin	—
34	A23	—	I/O	Address bus bit	8
35	PP8	A24	I/O	Parallel port bit/Address bus bit	8
36	PP9	A25	I/O	Parallel port bit/Address bus bit	8
37	VCC	—	—	Power input	—
38	PP10	A26	I/O	Parallel port bit/Address bus bit	8
39	PP11	A27	I/O	Parallel port bit/Address bus bit	8
40	PP12	A28	I/O	Parallel port bit/Address bus bit	8
41	GND	—	—	Ground pin	—
42	PP13	A29	I/O	Parallel port bit/Address bus bit	8
43	PP14	A30	I/O	Parallel port bit/Address bus bit	8
44	PP15	A31	I/O	Parallel port bit/Address bus bit	8
45	VCC	—	—	Power input	—
46	SIZ0	—	I/O	Size attribute	8
47	SIZ1	—	I/O	Size attribute	8

Table 16-1. Pins 1–52 (Left, Top-to-Bottom) (Continued)

Pin		Alternate Function	I/O	Description	Drive (mA)
No	Name				
48	GND	—	—	Ground pin	—
49	$\overline{OE}$	—	O	Output enable for chip selects	8
50	$\overline{CS0}$	—	O	Chip select	8
51	$\overline{CS1}$	—	O	Chip select	8
52	VCC	—	—	Power input	—

Table 16-2. Pins 53–104 (Bottom, Left-to-Right)

Pin		Alternate Function	I/O	Description	Drive (mA)
No	Name				
53	GND	—	—	Ground pin	—
54	$\overline{CS2}$	—	O	Chip select	8
55	$\overline{CS3}$	—	O	Chip select	8
56	$\overline{CS4}$	—	O	Chip select	8
57	VCC	—	—	Power input	—
58	$\overline{CS5}$	—	O	Chip select	8
59	$\overline{CS6}$	—	O	Chip select	8
60	$\overline{CS7}$	—	O	Chip select	8
61	GND	—	—	Ground pin	—
62	$\overline{AS}$	—	I/O	Address strobe	8
63	R/$\overline{W}$	—	I/O	Read/$\overline{Write}$	8
64	$\overline{TA}$	—	I/O	Transfer acknowledge	8
65	VCC	—	—	Power input	—
66	$\overline{TS}$	—	I/O	Transfer start	8
67	$\overline{RSTI}$	—	I	Reset	—
68	$\overline{IRQ7}$	—	I	Interrupt request	—
69	GND	—	—	Ground pin	—
70	$\overline{IRQ5}$	$\overline{IRQ4}$	I	Interrupt request	—
71	$\overline{IRQ3}$	$\overline{IRQ6}$	I	Interrupt request	—
72	$\overline{IRQ1}$	$\overline{IRQ2}$	I	Interrupt request	—
73	VCC	—	—	Power input	—
74	$\overline{BR}$	—	O	Bus request	8
75	$\overline{BD}$	—	O	Bus driven	8
76	$\overline{BG}$	—	I	Bus grant	—
77	GND	—	—	Ground pin	—

external master access. This condition is indicated by the AM bit in the chip-select mask register (CSMR) being cleared. See Chapter 10, “Chip-Select Module.”

17.2.8 Transfer In Progress ($\overline{\text{TIP}}$/PP7)

The $\overline{\text{TIP}}$/PP7 pin is programmed in the PAR to serve as the transfer-in-progress output or as a parallel port bits. The $\overline{\text{TIP}}$ output is asserted indicating a bus transfer is in progress. It is negated during idle bus cycles if the bus is still granted to the processor. It is three-stated for external master accesses. Note that $\overline{\text{TIP}}$ is held asserted on back-to-back bus cycles.

17.2.9 Transfer Type (TT[1:0]/PP[1:0])

The TT[1:0]/PP[1:0] pins are programmed in the PAR to serve as the transfer type outputs or as two parallel port bits. When the MCF5307 is bus master and TT[1:0] are enabled, these signals are driven as outputs only. If an external master owns the bus and TT[1:0] are enabled, these pins are three-stated by the MCF5307 and can be driven by the external master. Table 17-5 shows the definition of the encodings.

Table 17-5. Bus Cycle Transfer Type Encoding

TT[1:0]	Transfer Type
00	Normal access
01	DMA access
10	Emulator access
11	CPU space or interrupt acknowledge

17.2.10 Transfer Modifier (TM[2:0]/PP[4:2])

The TM[2:0]/PP[4:2] pins are programmed in the PAR to serve as the transfer modifier outputs or as three parallel port bits. These outputs provide supplemental information for each transfer type; see Table 17-6 through Table 17-10.

When the MCF5307 is the bus master and TM[2:0] are enabled, these signals are driven as outputs only. If an external device is bus master and TM[2:0] are enabled, these pins are three-stated by the MCF5307 and can be driven by the external master.

Table 17-6. TM[2:0] Encodings for TT = 00 (Normal Access)

TM[2:0]	Transfer Modifier
000	Cache push access
001	User data access
010	User code access
011–100	Reserved
101	Supervisor data access

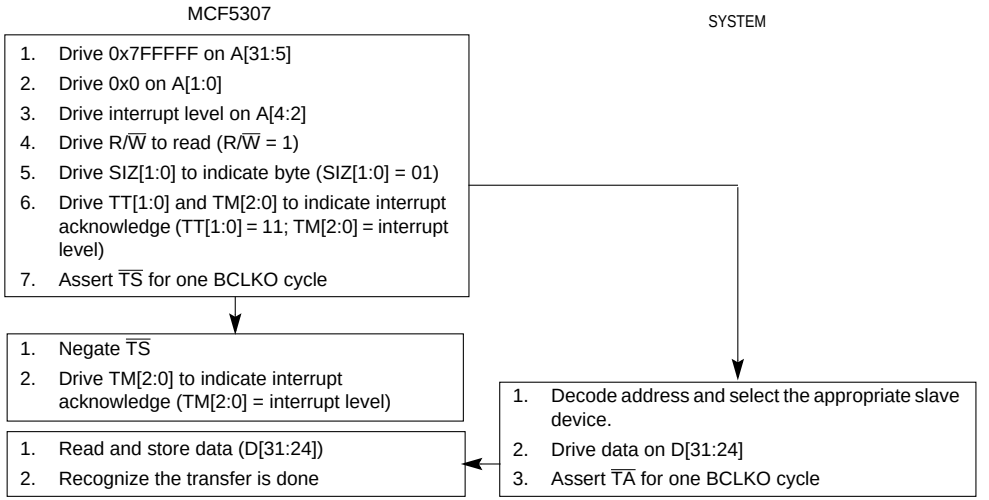


Figure 18-23. Interrupt-Acknowledge Cycle Flowchart

18.8 Bus Arbitration

The MCF5307 bus protocol gives either the MCF5307 or an external device access to the external bus. If more than one external device uses the bus, an external arbiter can prioritize requests and determine which device is bus master. When the MCF5307 is bus master, it uses the bus to fetch instructions and transfer data to and from external memory. When an external device is bus master, the MCF5307 can monitor the external master's transfers and interact through its chip-select, DRAM control, and transfer termination signals. See Section 10.4.1.3, "Chip-Select Control Registers (CSCR0–CSCR7)," and Chapter 11, "Synchronous/Asynchronous DRAM Controller Module."

Two-wire bus arbitration is used where the MCF5307 shares the bus with a single external device. This mode uses $\overline{BG}$ and $\overline{BD}$. The external device can ignore $\overline{BR}$. Three-wire mode is used where the MCF5307 shares the bus with multiple external devices. This requires an external bus arbiter and uses $\overline{BG}$, $\overline{BD}$, and $\overline{BR}$. In either mode, the MCF5307 bus arbiter operates synchronously and transitions between states on the rising edge of BCLKO.

Table 18-6 shows the four arbitration states the MCF5307 can be in during bus operation.

Table 18-6. MCF5307 Arbitration Protocol States

State	Master	Bus	$\overline{BD}$	Description
Reset	None	Not driven	Negated	The MCF5307 enters reset state from any other state when $\overline{RSTI}$ or software watchdog reset is asserted. If both are negated, the MCF5307 enters implicit or external device mastership state, depending on $\overline{BG}$.
Implicit master	MCF5307	Not driven	Negated	The MCF5307 is bus master ($\overline{BG}$ input is asserted) but is not ready to begin a bus cycle. It continues to three-state the bus until an internal bus request.

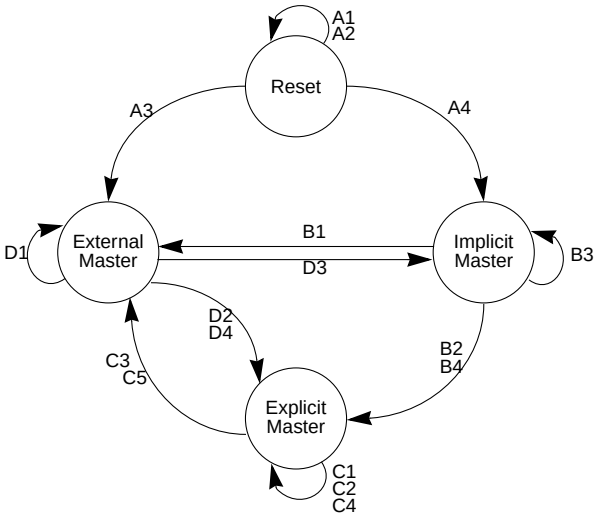


Figure 18-29. MCF5307 Two-Wire Bus Arbitration Protocol State Diagram

Table 18-10 describes the two-wire bus arbitration protocol transition conditions.

Table 18-10. MCF5307 Two-Wire Bus Arbitration Protocol Transition Conditions

Present State	Condition Label	RSTI	Software Watchdog Reset	BG	Bus Request	Transfer in Progress	End of Cycle[1]	Next State
Reset	A1	A[2]	—	—	—	—	—	Reset
	A2	N[3]	A	—	—	—	—	Reset
	A3	N	N	N	—	—	—	EM [4]
	A4	N	N	A	—	—	—	Implicit mas
Implicit Master	B1	N	N	N	—	—	—	EM
	B2	N	N	A	—	—	—	Explicit mas
	B3	N	N	A	N	—	—	Implicit mas
	B4	N	N	A	A	—	—	Explicit mas
Explicit Master	C1	N	N	A	—	—	—	Explicit mas
	C2	N	N	N	—	—	—	Explicit mas
	C3	N	N	N	—	N	—	EM
	C4	N	N	N	—	A	N	Explicit mas
	C5	N	N	N	—	A	A	EM
External Master	D1	N	N	N	—	—	—	EM mas
	D2	N	N	A	—	—	_	Explicit mas
	D3	N	N	A	N	—	—	Implicit mas
	D4	N	N	A	A	—	—	Explicit mas

1 Both normal terminations and terminations due to bus errors generate an end of cycle. Bus cycles resulting from a burst-inhibited transfer are considered part of that original transfer.
2 A means asserted.
3 N means negated.
4 EM means external master.

18.9.2 Multiple External Bus Device Arbitration Protocol (Three-Wire Mode)

Three-wire mode lets the MCF5307 share the external bus with multiple external devices. This mode requires an external arbiter to assign priorities to each potential master and to determine which device accesses the external bus. The arbiter uses the MCF5307 bus arbitration signals, $\overline{BR}$, $\overline{BD}$, and $\overline{BG}$, to control use of the external bus by the MCF5307.

The MCF5307 requests the bus from the external bus arbiter by asserting $\overline{BR}$ when the core requests an access. It continues to assert $\overline{BR}$ until after the transfer starts. It can negate $\overline{BR}$ at any time regardless of the $\overline{BG}$ status. If the MCF5307 is granted the bus when an internal bus request is generated, it asserts $\overline{BD}$ and the access begins immediately. The MCF5307 always drives $\overline{BR}$ and $\overline{BD}$, which cannot be directly wire-ORed with other devices.

The external arbiter asserts $\overline{BG}$ to grant the bus to MCF5307, which can begin a bus cycle after the next rising edge of BCLKO. If $\overline{BG}$ is negated during a bus cycle, the MCF5307 releases the bus when the cycle completes. To guarantee that the bus is released, $\overline{BG}$ must be negated before the rising edge of the BCLKO in which the last $\overline{TA}$ is asserted. Note that the MCF5307 treats any series of burst or a burst-inhibited transfers as a single bus cycle and does not release the bus until the last transfer of the series completes.

When the MCF5307 is granted the bus after it asserts $\overline{BR}$, one of two things can occur. If the MCF5307 has an internal bus request pending, it asserts $\overline{BD}$, indicating explicit bus mastership, and begins the pending bus cycle by asserting $\overline{TS}$. The MCF5307 continues to assert $\overline{BD}$ until the external bus arbiter negates $\overline{BG}$, after which $\overline{BD}$ is negated at the completion of the bus cycle. As long as $\overline{BG}$ is asserted, $\overline{BD}$ remains asserted to indicate that the MCF5307 is bus master, and the MCF5307 continuously drives the address bus, attributes, and control signals.

If no internal request is pending, the MCF5307 takes implicit bus mastership. It does not drive the bus and does not assert $\overline{BD}$ if the bus has an implicit master. If an internal bus request is generated, the MCF5307 assumes explicit bus mastership and immediately begins an access and asserts $\overline{BD}$. Figure 18-30 shows implicit and explicit bus mastership due to generation of an internal bus request.

Operation." Note that Figure 20-4 does not show all signals that apply to each timing specification. See the previous tables for a complete listing.

Figure 20-3 shows AC timings for normal read and write bus cycles.

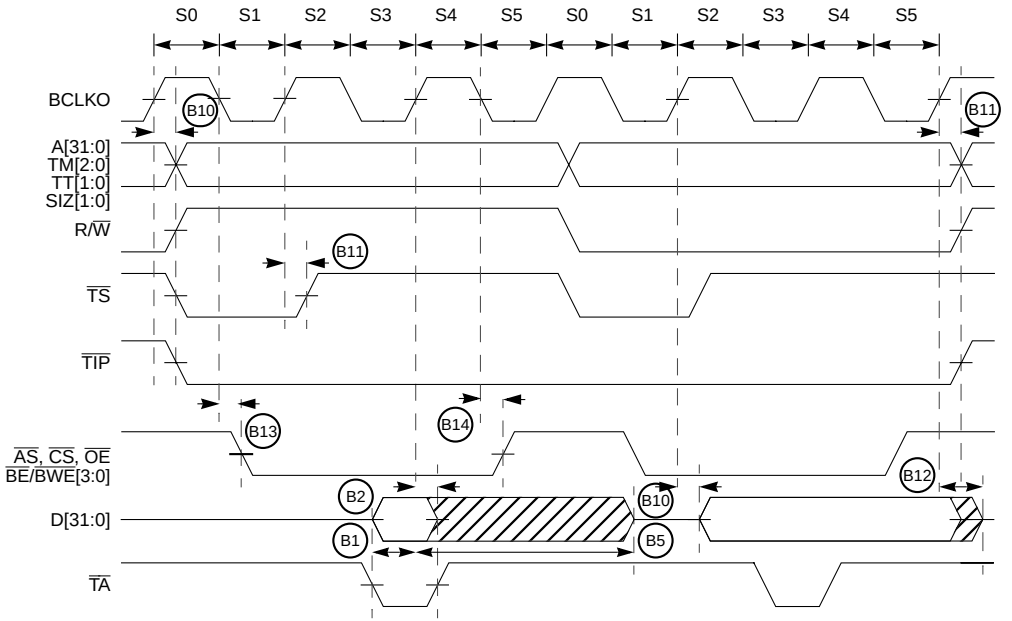


Figure 20-3. AC Timings—Normal Read and Write Bus Cycles

Figure 20-4 shows timings for a read cycle with EDGESEL tied to buffered BCLKO.

Implementation. A particular processor that conforms to the ColdFire architecture, but may differ from other architecture-compliant implementations for example in design, feature set, and implementation of *optional* features. The ColdFire architecture has many different implementations.

Imprecise mode. A memory access mode that allows write accesses to a specified memory region to occur out of order.

Instruction queue. A holding place for instructions fetched from the current instruction stream.

Instruction latency. The total number of clock cycles necessary to execute an instruction and make the results of that instruction available.

Interrupt. An *asynchronous exception*. On ColdFire processors, interrupts are a special case of exceptions. See also asynchronous exception.

Invalid state. State of a cache entry that does not currently contain a valid copy of a cache line from memory.

L

Least-significant bit (lsb). The bit of least value in an address, register, data element, or instruction encoding.

Least-significant byte (LSB). The byte of least value in an address, register, data element, or instruction encoding.

Longword. A 32-bit data element

M

Master. A device able to initiate data transfers on a bus. Bus mastering refers to a feature supported by some bus architectures that allow a controller connected to the bus to communicate directly with other devices on the bus without going through the CPU.

Memory coherency. An aspect of caching in which it is ensured that an accurate view of memory is provided to all devices that share system memory.

Modified state. Cache state in which only one caching device has the valid data for that address.

Most-significant bit (msb). The highest-order bit in an address, registers, data element, or instruction encoding.

Most-significant byte (MSB). The highest-order byte in an address, registers, data element, or instruction encoding.