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	Obsolete
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	0°C ~ 70°C (TA)
Mounting Type	Surface Mount
Package / Case	208-BFQFP
Supplier Device Package	208-FQFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mcf5307ai90b

ColdFire is a registered trademark and DigitalDNA is a trademark of Motorola, Inc.
I^2C is a registered trademark of Philips Semiconductors

How to reach us:

USA/EUROPE/Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 1–303–675–2140 or 1–800–441–2447

JAPAN: Motorola Japan Ltd.; SPS, Technical Information Center, 3–20–1, Minami–Azabu. Minato–ku, Tokyo 106–8573 Japan. 81–3–3440–3569

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong. 852–26668334

Technical Information Center: 1–800–521–6274

HOME PAGE: http://www.motorola.com/semiconductors

Document Comments: FAX (512) 895-2638, Attn: RISC Applications Engineering

World Wide Web Addresses: http://www.motorola.com/PowerPC
http://www.motorola.com/NetComm
http://www.motorola.com/ColdFire

Freescale Semiconductor, Inc.

CONTENTS

short distances among several devices. The I^2C can operate in master, slave, or multiple-master modes.

1.3.7 System Interface

The MCF5307 processor provides a direct interface to 8-, 16-, and 32-bit FLASH, SRAM, ROM, and peripheral devices through the use of fully programmable chip selects and write enables. Support for burst ROMs is also included. Through the on-chip PLL, users can input a slower clock (16.6 to 45 MHz) that is internally multiplied to create the faster processor clock (33.3 to 90 MHz).

1.3.7.1 External Bus Interface

The bus interface controller transfers information between the ColdFire core or DMA and memory, peripherals, or other devices on the external bus. The external bus interface provides up to 32 bits of address bus space, a 32-bit data bus, and all associated control signals. This interface implements an extended synchronous protocol that supports bursting operations.

Simple two-wire request/acknowledge bus arbitration between the MCF5307 processor and another bus master, such as an external DMA device, is glueless with arbitration logic internal to the MCF5307 processor. Multiple-master arbitration is also available with some simple external arbitration logic.

1.3.7.2 Chip Selects

Eight fully programmable chip select outputs support the use of external memory and peripheral circuits with user-defined wait-state insertion. These signals interface to 8-, 16-, or 32-bit ports. The base address, access permissions, and internal bus transfer terminations are programmable with configuration registers for each chip select. $\overline{CS0}$ also provides global chip select functionality of boot ROM upon reset for initializing the MCF5307.

1.3.7.3 16-Bit Parallel Port Interface

A 16-bit general-purpose programmable parallel port serves as either an input or an output on a pin-by-pin basis.

1.3.7.4 Interrupt Controller

The interrupt controller provides user-programmable control of ten internal peripheral interrupts and implements four external fixed interrupt-request pins. Each internal interrupt can be programmed to any one of seven interrupt levels and four priority levels within each of these levels. Additionally, the external interrupt request pins can be mapped to levels 1, 3, 5, and 7 or levels 2, 4, 6, and 7. Autovector capability is available for both internal and external interrupts.

Table 2-5. ColdFire Effective Addressing Modes

Addressing Modes	Syntax	Mode Field	Reg. Field	Category			
				Data	Memory	Control	Alterable
Register direct Data Address	 Dn An	 000 001	 reg. no. reg. no.	 X —	 — —	 — —	 X X
Register indirect Address Address with Postincrement Address with Predecrement Address with Displacement	 (An) (An)+ –(An) (d_{16}, An)	 010 011 100 101	 reg. no. reg. no. reg. no. reg. no.	 X X X X	 X X X X	 X — — X	 X X X X
Address register indirect with index 8-bit displacement	(d_8, An, Xi)	110	reg. no.	X	X	X	X
Program counter indirect with displacement	(d_{16}, PC)	111	010	X	X	X	—
Program counter indirect with index 8-bit displacement	(d_8, PC, Xi)	111	011	X	X	X	—
Absolute data addressing Short Long	 (xxx).W (xxx).L	 111 111	 000 001	 X X	 X X	 X X	 — —
Immediate	#<xxx>	111	100	X	X	—	—

2.6 Instruction Set Summary

The ColdFire instruction set is a simplified version of the M68000 instruction set. The removed instructions include BCD, bit field, logical rotate, decrement and branch, and integer multiply with a 64-bit result. Nine new MAC instructions have been added.

Table 2-6 lists notational conventions used throughout this manual.

Table 2-6. Notational Conventions

Instruction	Operand Syntax
Opcode Wildcard	
cc	Logical condition (example: NE for not equal)

Table 2-7. User-Mode Instruction Set Summary (Continued)

Instruction	Operand Syntax	Operand Size	Operation
MOVEM	#<list>,<ea-2>x <ea-2>y,#<list>	.L .L	Listed registers → destination Source → listed registers
MOVEQ	#<data>,Dx	.B → .L	Sign-extended immediate data → destination
MSAC	Ry,RxSF	.L - (.W × .W) → .L .L - (.L × .L) → .L	ACC – (Ry × Rx){<< 1 | >> 1} → ACC
MSACL	Ry,RxSF,<ea-1>y,Rw	.L - (.W × .W) → .L, .L .L - (.L × .L) → .L, .L	ACC – (Ry × Rx){<< 1 | >> 1} → ACC; (<ea-1>y{&MASK}) → Rw
MULS	<ea>y,Dx	.W X .W → .L .L X .L → .L	Source × destination → destination Signed operation
MULU	<ea>y,Dx	.W X .W → .L .L X .L → .L	Source × destination → destination Unsigned operation
NEG	Dx	.L	0 – destination → destination
NEGX	Dx	.L	0 – destination – X → destination
NOP	none	Unsized	Synchronize pipelines; PC + 2 → PC
NOT	Dx	.L	~ Destination → destination
OR	<ea>y,Dx Dy,<ea>x	.L	Source | destination → destination
ORI	#<data>,Dx	.L	Immediate data | destination → destination
PEA	<ea-3>y	.L	SP – 4 → SP; Address of <ea> → (SP)
PULSE	none	Unsized	Set PST= 0x4
REMS	<ea-1>,Dx	.L	Dx/<ea>y → Dw {32-bit remainder} Signed operation
REMU	<ea-1>,Dx	.L	Dx/<ea>y → Dw {32-bit remainder} Unsigned operation
RTS	none	Unsized	(SP) → PC; SP + 4 → SP
Scc	Dx	.B	If condition true, then 1s — destination; Else 0s → destination
SUB	<ea>y,Dx Dy,<ea>x	.L .L	Destination – source → destination
SUBA	<ea>y,Ax	.L	Destination – source → destination
SUBI	#<data>,Dx	.L	Destination – immediate data → destination
SUBQ	#<data>,<ea>x	.L	Destination – immediate data → destination
SUBX	Dy,Dx	.L	Destination – source – X → destination
SWAP	Dx	.W	MSW of Dx ←→ LSW of Dx
TRAP	#<vector>	Unsized	SP – 4 → SP;PC → (SP); SP – 2 → SP;SR → (SP); SP – 2 → SP; format → (SP); Vector address → PC
TRAPF	None #<data>	Unsized .W .L	PC + 2 → PC PC + 4 → PC PC + 6 → PC

3. ACR1
4. If an access does not hit in the RAMBAR or the ACRs, the default is provided for all accesses in CACR.

Cache-inhibited write accesses bypass the cache and a corresponding external write is performed. Cache-inhibited reads bypass the cache and are performed on the external bus, except when all of the following conditions are true:

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

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

If ACR*n*[CM] indicates cache-inhibited mode, precise or imprecise, the controller bypasses the cache and performs an external transfer. If a line in the cache matches the address and the mode is cache-inhibited, the cache does not automatically push the line if it is modified, nor does it invalidate the line if it is valid. Before switching cache mode, execute a CPUSHL instruction or set CACR[CINVA] to invalidate the entire cache.

If ACR*n*[CM] indicates precise mode, the sequence of read and write accesses to the region is guaranteed to match the instruction sequence. In imprecise mode, the processor core allows read accesses that hit in the cache to occur before completion of a pending write from a previous instruction. Writes are not deferred past data-read accesses that miss the cache (that is, that must be read from the bus).

Precise operation forces data-read accesses for an instruction to occur only once by preventing the instruction from being interrupted after data is fetched. Otherwise, if the processor is not in precise mode, an exception aborts the instruction and the data may be accessed again when the instruction is restarted. These guarantees apply only when ACR*n*[CM] indicates precise mode and aligned accesses.

CPU space-register accesses, such as MOVEC, are treated as cache-inhibited and precise.

4.9.3 Cache Protocol

The following sections describe the cache protocol for processor accesses and assumes that the data is cacheable (that is, write-through or copyback).

4.9.3.1 Read Miss

A processor read that misses in the cache requests the cache controller to generate a bus transaction. This bus transaction reads the needed line from memory and supplies the required data to the processor core. The line is placed in the cache in the valid state.

Table 4-6. Cache Line State Transitions

Access	Current State					
	Invalid (V = 0)		Valid (V = 1, M = 0)		Modified (V = 1, M = 1)	
Read miss	(C,W)I1	Read line from memory and update cache; supply data to processor; go to valid state.	(C,W)V1	Read new line from memory and update cache; supply data to processor; stay in valid state.	CD1	Push modified line to buffer; read new line from memory and update cache; supply data to processor; write push buffer contents to memory; go to valid state.
Read hit	(C,W)I2	Not possible.	(C,W)V2	Supply data to processor; stay in valid state.	CD2	Supply data to processor; stay in modified state.
Write miss (copy-back)	CI3	Read line from memory and update cache; write data to cache; go to modified state.	CV3	Read new line from memory and update cache; write data to cache; go to modified state.	CD3	Push modified line to buffer; read new line from memory and update cache; write push buffer contents to memory; stay in modified state.
Write miss (write-through)	WI3	Write data to memory; stay in invalid state.	WV3	Write data to memory; stay in valid state.	WD3	Write data to memory; stay in modified state. Cache mode changed for the region corresponding to this line. To avoid this state, execute a CPUSHL instruction or set CACR[CINVA] before switching modes.
Write hit (copy-back)	CI4	Not possible.	CV4	Write data to cache; go to modified state.	CD4	Write data to cache; stay in modified state.
Write hit (write-through)	WI4	Not possible.	WV4	Write data to memory and to cache; stay in valid state.	WD4	Write data to memory and to cache; go to valid state. Cache mode changed for the region corresponding to this line. To avoid this state, execute a CPUSHL instruction or set CACR[CINVA] before switching modes.
Cache invalidate	(C,W)I5	No action; stay in invalid state.	(C,W)V5	No action; go to invalid state.	CD5	No action (modified data lost); go to invalid state.
Cache push	(C,W)I6 (C,W)I7	No action; stay in invalid state.	(C,W)V6	No action; go to invalid state.	CD6	Push modified line to memory; go to invalid state.
			(C,W)V7	No action; stay in valid state.	CD7	Push modified line to memory; go to valid state.

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.

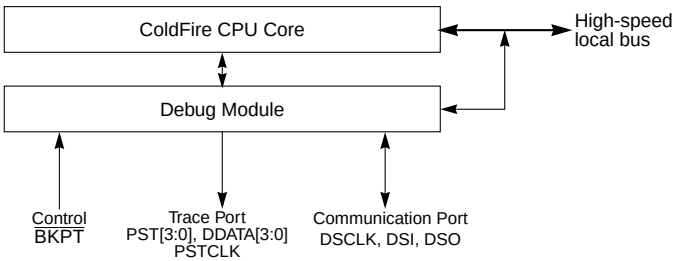

Figure 5-1. Processor/Debug Module Interface

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.”

DSI must meet the required input setup and hold timings and the DSO is specified as a delay relative to the rising edge of the processor clock. See Table 5-1. The development system serves as the serial communication channel master and must generate DSCLK.

The serial channel operates at a frequency from DC to 1/5 of the processor frequency. The channel uses full-duplex mode, where data is sent and received simultaneously by both master and slave devices. The transmission consists of 17-bit packets composed of a status/control bit and a 16-bit data word. As shown in Figure 5-13, all state transitions are enabled on a rising edge of the processor clock when DSCLK is high; that is, DSI is sampled and DSO is driven.

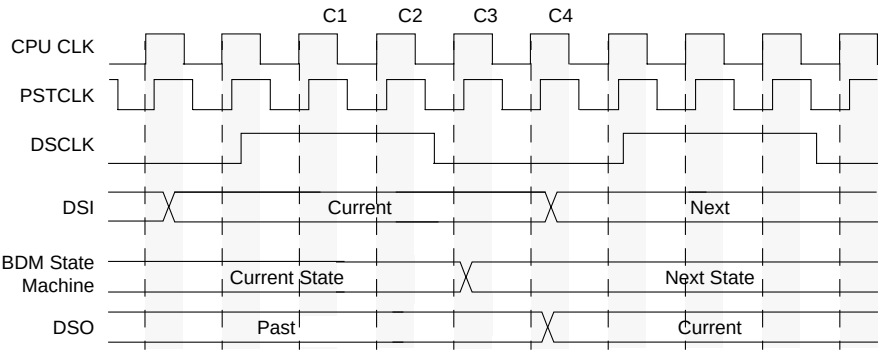

Figure 5-13. BDM Serial Interface Timing

DSCLK and DSI are synchronized inputs. DSCLK acts as a pseudo clock enable and is sampled on the rising edge of the processor CLK as well as the DSI. DSO is delayed from the DSCLK-enabled CLK rising edge (registered after a BDM state machine state change). All events in the debug module's serial state machine are based on the processor clock rising edge. DSCLK must also be sampled low (on a positive edge of CLK) between each bit exchange. The MSB is transferred first. Because DSO changes state based on an internally-recognized rising edge of DSCLK, DSDO cannot be used to indicate the start of a serial transfer. The development system must count clock cycles in a given transfer. C1–C4 are described as follows:

- C1—First synchronization cycle for DSI (DSCLK is high).
- C2—Second synchronization cycle for DSI (DSCLK is high).
- C3—BDM state machine changes state depending upon DSI and whether the entire input data transfer has been transmitted.
- C4—DSO changes to next value.

NOTE:

A not-ready response can be ignored except during a memory-referencing cycle. Otherwise, the debug module can accept a new serial transfer after 32 processor clock periods.

includes a description of signals involved in DRAM operations. The remainder of the chapter is divided between descriptions of asynchronous and synchronous operations.

Suggested Reading

The following literature may be helpful with respect to the topics in Part II:

- *The I^2C Bus Specification, Version 2.1* (January 2000)

Acronyms and Abbreviations

Table II-i contains acronyms and abbreviations are used in Part II.

Table II-i. Acronyms and Abbreviated Terms

Term	Meaning
ADC	Analog-to-digital conversion
BDM	Background debug mode
CODEC	Code/decode
DAC	Digital-to-analog conversion
DMA	Direct memory access
DSP	Digital signal processing
EDO	Extended data output (DRAM)
FIFO	First-in, first-out
GPIO	
I^2C	Inter-integrated circuit
IEEE	Institute for Electrical and Electronics Engineers
IPL	Interrupt priority level
JEDEC	Joint Electron Device Engineering Council
LIFO	Last-in, first-out
LRU	Least recently used
LSB	Least-significant byte
lsb	Least-significant bit
MBAR	Memory base address register
MSB	Most-significant byte
msb	Most-significant bit
Mux	Multiplex

Chapter 7
Phase-Locked Loop (PLL)

This chapter describes configuration and operation of the phase-locked loop (PLL) module. It describes in detail the registers and signals that support the PLL implementation.

7.1 Overview

The basic features of the MCF5307 PLL implementation are as follows:

- The PLL locks to the clock input (CLKIN) frequency. It provides a processor clock (PCLK) that is twice the input clock frequency and a programmable system bus clock output (BCLKO) that is 1/2, 1/3, or 1/4 the PCLK frequency.
- A buffered processor status clock (PSTCLK) is equal to the PCLK frequency, as indicated in Figure 7-1. This signal is made available for system development.

The PLL module has the following three modes of operation:

- Reset mode—In reset mode, the core/bus frequency ratio and other configuration information is sampled. At reset, the PLL asserts the reset out signal, $\overline{\text{RSTO}}$.
- Normal mode—During normal operations, the divide ratio is programmed at reset and is clock-multiplied to provide a maximum frequency of 90 MHz
- Reduced-power mode—In reduced-power mode, the high-speed processor core clocks are turned off without losing the register contents so that the system can be reenabled by an unmasked interrupt or reset.

Figure 7-1 shows the frequency relationships of PLL module clock signals.

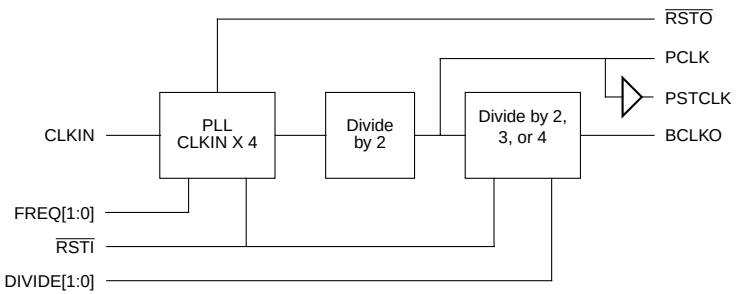

Figure 7-1. PLL Module Block Diagram

The SIM provides the following registers for managing interrupts:

- Each potential interrupt source is assigned one of the 10 interrupt control registers (ICR0–ICR9), which are used to prioritize the interrupt sources.
- The interrupt mask register (IMR) provides bits for masking individual interrupt sources.
- The interrupt pending register (IPR) provides bits for indicating when an interrupt request is being made (regardless of whether it is masked in the IMR).
- The autovector register (AVEC) controls whether the SIM supplies an autovector or executes an external interrupt acknowledge cycle for each IRQ.
- The interrupt port assignment register (IRQPAR) provides the level assignment of the primary external interrupt pins—IRQ5, IRQ3, and IRQ1.

9.2 Interrupt Controller Registers

The interrupt controller register portion of the SIM memory map is shown in Table 9-2.

Table 9-1. Interrupt Controller Registers

MBAR Offset	[31:24]	[23:16]	[15:8]	[7:0]
0x040	Interrupt pending register (IPR) [p. 9-6]			
0x044	Interrupt mask register (IMR) [p. 9-6]			
0x048	Reserved			Autovector register (AVR) [p. 9-5]
Interrupt Control Registers (ICRs) [p. 9-3]				
0x04C	Software watchdog timer (ICR0) [p. 9-3]	Timer0 (ICR1) [p. 9-3]	Timer1 (ICR2) [p. 9-3]	I^2C (ICR3) [p. 9-3]
0x050	UART0 (ICR4) [p. 9-3]	UART1 (ICR5) [p. 9-3]	DMA0 (ICR6) [p. 9-3]	DMA1 (ICR7) [p. 9-3]
0x054	DMA2 (ICR8) [p. 9-3]	DMA3 (ICR9) [p. 9-3]	Reserved	

Each internal interrupt source has its own interrupt control register (ICR0–ICR9), shown in Table 9-2 and described in Section 9.2.1, “Interrupt Control Registers (ICR0–ICR9).”

Table 9-2. Interrupt Control Registers

MBAR Offset	Register	Name
0x04C	ICR0	Software watchdog timer
0x04D	ICR1	Timer0
0x04E	ICR2	Timer1
0x04F	ICR3	I^2C
0x050	ICR4	UART0
0x051	ICR5	UART1
0x052	ICR6	DMA0

Table 11-12. DCR Field Descriptions (Synchronous Mode) (Continued)

Bits	Name	Description
12	COC	Command on SDRAM clock enable (SCKE). Implementations that use external multiplexing (NAM = 1) must support command information to be multiplexed onto the SDRAM address bus. 0 SCKE functions as a clock enable; self-refresh is initiated by the DRAM controller through DCR[IS]. 1 SCKE drives command information. Because SCKE is not a clock enable, self-refresh cannot be used (setting DCR[IS]). Thus, external logic must be used if this functionality is desired. External multiplexing is also responsible for putting the command information on the proper address bit.
11	IS	Initiate self-refresh command. 0 Take no action or issue a SELFX command to exit self refresh. 1 If DCR[COC] = 0, the DRAM controller sends a SELF command to both SDRAM blocks to put them in low-power, self-refresh state where they remain until IS is cleared, at which point the controller sends a SELFX command for the SDRAMs to exit self-refresh. The refresh counter is suspended while the SDRAMs are in self-refresh; the SDRAM controls the refresh period.
10–9	RTIM	Refresh timing. Determines the timing operation of auto-refresh in the DRAM controller. Specifically, it determines the number of clocks inserted between a REF command and the next possible ACTV command. This same timing is used for both memory blocks controlled by the DRAM controller. This corresponds to t_{RC} in the SDRAM specifications. 00 3 clocks 01 6 clocks 1x 9 clocks
8–0	RC	Refresh count. Controls refresh frequency. The number of bus clocks between refresh cycles is (RC + 1) * 16. Refresh can range from 16–8192 bus clocks to accommodate both standard and low-power DRAMs with bus clock operation from less than 2 MHz to greater than 50 MHz. The following example calculates RC for an auto-refresh period for 4096 rows to receive 64 mS of refresh every 15.625 µs for each row (625 bus clocks at 40 MHz). This operation is the same as in asynchronous mode. # of bus clocks = 625 = (RC field + 1) * 16 RC = (625 bus clocks/16) -1 = 38.06, which rounds to 38; therefore, RC = 0x26.

11.4.3.2 DRAM Address and Control Registers (DACR0/DACR1) in Synchronous Mode

The DRAM address and control registers (DACR0 and DACR1), shown in Figure 11-16, contain the base address compare value and the control bits for both memory blocks 0 and 1 of the DRAM controller. Address and timing are also controlled by bits in DACR*n*.

	31 … 18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	BA		—	RE	—	CASL		—	CBM			—	IMRS	PS		IP	PM	—	
Reset	Uninitialized			0	Uninitialized								0	Uninitialized					
R/W	R/W																		
Addr	MBAR+0x108 (DACR0); 0x110(DACR1)																		

Figure 11-16. DACR0 and DACR1 Registers (Synchronous Mode)

Table 11-35. DACR Initialization Values

Bits	Name	Setting	Description
2	PM	1	Indicates continuous page mode
1–0	—		Reserved. Don't care.

11.5.4 DMR Initialization

In this example, again, only the second 512-Kbyte block of each 1-Mbyte space is accessed in each bank. In addition the SDRAM component is mapped only to readable and writable supervisor and user data. The DMRs have the following configuration.

	31														18	17	16
Field	BAM															—	
Setting	0	0	0	0	0	0	0	0	0	1	1	1	0	1		X	X
(hex)	0				0				7				4				

	15						9	8	7	6	5	4	3	2	1	0
Field	—							WP	—	C/I	AM	SC	SD	UC	UD	V
Setting	X	X	X	X	X	X	X	0	X	1	1	1	0	1	0	1
(hex)	0				0				7				5			

Figure 11-28. DMR0 Register

With this configuration, the DMR0 = 0x0074_0075, as described in Table 11-36.

Table 11-36. DMR0 Initialization Values

Bits	Name	Setting	Description
31–16	BAM		With bits 17 and 16 as don't cares, BAM = 0x0074, which leaves bank select bits and upper 512K select bits unmasked. Note that bits 22 and 21 are set because they are used as bank selects; bit 20 is set because it controls the 1-Mbyte boundary address.
15–9	—		Reserved. Don't care.
8	WP	0	Allow reads and writes
7	—		Reserved
6	C/I	1	Disable CPU space access
5	AM	1	Disable alternate master access
4	SC	1	Disable supervisor code accesses
3	SD	0	Enable supervisor data accesses
2	UC	1	Disable user code accesses
1	UD	0	Enable user data accesses
0	V	1	Enable accesses.

Chapter 12
DMA Controller Module

This chapter describes the MCF5307 DMA controller module. It provides an overview of the module and describes in detail its signals and registers. The latter sections of this chapter describe operations, features, and supported data transfer modes in detail.

12.1 Overview

The direct memory access (DMA) controller module provides an efficient way to move blocks of data with minimal processor interaction. The DMA module, shown in Figure 12-1, provides four channels that allow byte, word, or longword operand transfers. Each channel has a dedicated set of registers that define the source and destination addresses (SAR*n* and DAR*n*), byte count (BCR*n*), and control and status (DCR*n* and DSR*n*). Transfers can be dual or single address to off-chip devices or dual address to on-chip devices, such as UART, SDRAM controller, and parallel port.

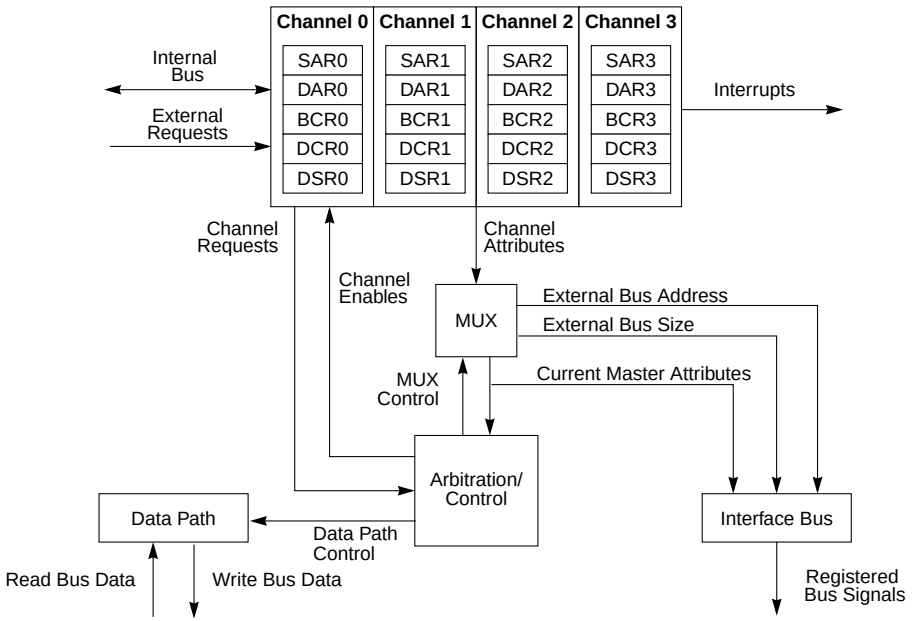

Figure 12-1. DMA Signal Diagram

Table 13-5. Calculated Time-out Values (90-MHz Processor Clock) (Continued)

TMR[PS]		TMR[CLK] = 10 (System Bus Clock/16)			TMR[CLK] = 01 (System Bus Clock/1)		
Decimal	Hex	45 MHz	30 MHz	22.5 MHz	45 MHz	30 MHz	22.5 MHz
126	7E	2.95931	4.43897	5.91863	0.18496	0.27744	0.36991
127	7F	2.98262	4.47392	5.96523	0.18641	0.27962	0.37283
128	80	3.00592	4.50888	6.01184	0.18787	0.2818	0.37574
129	81	3.02922	4.54383	6.05844	0.18933	0.28399	0.37865
130	82	3.05252	4.57878	6.10504	0.19078	0.28617	0.38157
131	83	3.07582	4.61373	6.15165	0.19224	0.28836	0.38448
132	84	3.09912	4.64869	6.19825	0.1937	0.29054	0.38739
133	85	3.12243	4.68364	6.24485	0.19515	0.29273	0.3903
134	86	3.14573	4.71859	6.29146	0.19661	0.29491	0.39322
135	87	3.16903	4.75354	6.33806	0.19806	0.2971	0.39613
136	88	3.19233	4.7885	6.38466	0.19952	0.29928	0.39904
137	89	3.21563	4.82345	6.43127	0.20098	0.30147	0.40195
138	8A	3.23893	4.8584	6.47787	0.20243	0.30365	0.40487
139	8B	3.26224	4.89335	6.52447	0.20389	0.30583	0.40778
140	8C	3.28554	4.92831	6.57108	0.20535	0.30802	0.41069
141	8D	3.30884	4.96326	6.61768	0.2068	0.3102	0.4136
142	8E	3.33214	4.99821	6.66428	0.20826	0.31239	0.41652
143	8F	3.35544	5.03316	6.71089	0.20972	0.31457	0.41943
144	90	3.37874	5.06812	6.75749	0.21117	0.31676	0.42234
145	91	3.40205	5.10307	6.80409	0.21263	0.31894	0.42526
146	92	3.42535	5.13802	6.8507	0.21408	0.32113	0.42817
147	93	3.44865	5.17297	6.8973	0.21554	0.32331	0.43108
148	94	3.47195	5.20793	6.9439	0.217	0.3255	0.43399
149	95	3.49525	5.24288	6.99051	0.21845	0.32768	0.43691
150	96	3.51856	5.27783	7.03711	0.21991	0.32986	0.43982
151	97	3.54186	5.31279	7.08371	0.22137	0.33205	0.44273
152	98	3.56516	5.34774	7.13032	0.22282	0.33423	0.44564
153	99	3.58846	5.38269	7.17692	0.22428	0.33642	0.44856
154	9A	3.61176	5.41764	7.22352	0.22574	0.3386	0.45147
155	9B	3.63506	5.4526	7.27013	0.22719	0.34079	0.45438
156	9C	3.65837	5.48755	7.31673	0.22865	0.34297	0.4573
157	9D	3.68167	5.5225	7.36333	0.2301	0.34516	0.46021
158	9E	3.70497	5.55745	7.40994	0.23156	0.34734	0.46312
159	9F	3.72827	5.59241	7.45654	0.23302	0.34953	0.46603
160	A0	3.75157	5.62736	7.50314	0.23447	0.35171	0.46895
161	A1	3.77487	5.66231	7.54975	0.23593	0.35389	0.47186
162	A2	3.79818	5.69726	7.59635	0.23739	0.35608	0.47477
163	A3	3.82148	5.73222	7.64295	0.23884	0.35826	0.47768
164	A4	3.84478	5.76717	7.68956	0.2403	0.36045	0.4806
165	A5	3.86808	5.80212	7.73616	0.24176	0.36263	0.48351

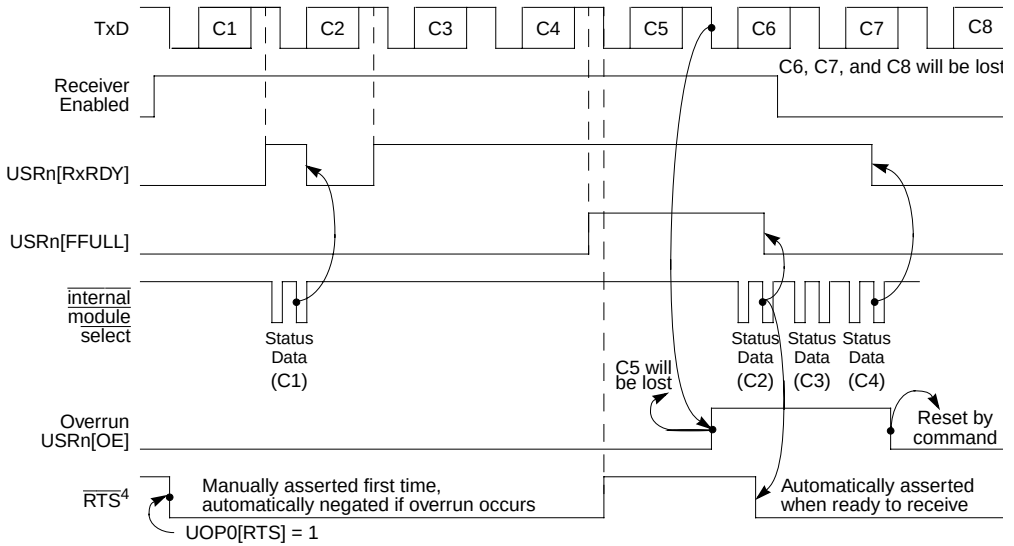

Figure 14-22. Receiver Timing

When the receiver detects a high-to-low (mark-to-space) transition of the start bit on RxD, the state of RxD is sampled each 16× clock for eight clocks, starting one-half clock after the transition (asynchronous operation) or at the next rising edge of the bit time clock (synchronous operation). If RxD is sampled high, the start bit is invalid and the search for the valid start bit begins again.

If RxD is still low, a valid start bit is assumed and the receiver continues sampling the input at one-bit time intervals, at the theoretical center of the bit, until the proper number of data bits and parity, if any, is assembled and one stop bit is detected. Data on the RxD input is sampled on the rising edge of the programmed clock source. The lsb is received first. The data is then transferred to a receiver holding register and USR*n*[RxRDY] is set. If the character is less than eight bits, the most significant unused bits in the receiver holding register are cleared.

After the stop bit is detected, the receiver immediately looks for the next start bit. However, if a non-zero character is received without a stop bit (framing error) and RxD remains low for one-half of the bit period after the stop bit is sampled, the receiver operates as if a new start bit were detected. Parity error, framing error, overrun error, and received break conditions set the respective PE, FE, OE, RB error and break flags in the USR*n* at the received character boundary and are valid only if USR*n*[RxRDY] is set.

If a break condition is detected (RxD is low for the entire character including the stop bit), a character of all zeros is loaded into the receiver holding register (RHR) and USR*n*[RB,RxRDY] are set. RxD must return to a high condition for at least one-half bit time before a search for the next start bit begins.

20.7 I²C Input/Output Timing Specifications

Table 20-10 lists specifications for the I²C input timing parameters shown in Figure 20.8.

Table 20-10. I²C Input Timing Specifications between SCL and SDA

Num	Characteristic	66 MHz		90 MHz		Units
		Min	Max	Min	Max	
I1	Start condition hold time	2	—	2	—	Bus clocks
I2	Clock low period	8	—	8	—	Bus clocks
I3	SCL/SDA rise time (V_{IL} = 0.5 V to V_{IH} = 2.4 V)	—	1	—	1	mS
I4	Data hold time	0	—	0	—	nS
I5	SCL/SDA fall time (V_{IH} = 2.4 V to V_{IL} = 0.5 V)	—	1	—	1	mS
I6	Clock high time	4	—	4	—	Bus clocks
I7	Data setup time	0	—	0	—	nS
I8	Start condition setup time (for repeated start condition only)	2	—	2	—	Bus clocks
I9	Stop condition setup time	2	—	2	—	Bus clocks

Table 20-11 lists specifications for the I²C output timing parameters shown in Figure 20.8.

Table 20-11. I²C Output Timing Specifications between SCL and SDA

Num	Characteristic	66 MHz		90 MHz		Units
		Min	Max	Min	Max	
I1[1]	Start condition hold time	6	—	6	—	Bus clocks
I2 [1]	Clock low period	10	—	10	—	Bus clocks
I3 [2]	SCL/SDA rise time (V_{IL} = 0.5 V to V_{IH} = 2.4 V)	—	—	—	—	µS
I4 [1]	Data hold time	7	—	7	—	Bus clocks
I5 [3]	SCL/SDA fall time (V_{IH} = 2.4 V to V_{IL} = 0.5 V)	—	3	—	3	nS
I6 [1]	Clock high time	10	—	10	—	Bus clocks
I7 [1]	Data setup time	2	—	2	—	Bus clocks
I8 [1]	Start condition setup time (for repeated start condition only)	20	—	20	—	Bus clocks
I9 [1]	Stop condition setup time	10	—	10	—	Bus clocks

1 Note: Output numbers depend on the value programmed into the IFDR; an IFDR programmed with the maximum frequency (IFDR = 0x20) results in minimum output timings as shown in Table 20-11. The I²C interface is designed to scale the actual data transition time to move it to the middle of the SCL low period. The actual position is affected by the prescale and division values programmed into the IFDR; however, the numbers given in Table 20-11 are minimum values.

2 Because SCL and SDA are open-collector-type outputs, which the processor can only actively drive low, the time SCL or SDA take to reach a high level depends on external signal capacitance and pull-up resistor values.

3 Specified at a nominal 50-pF load.

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.