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 V4
Core Size	32-Bit Single-Core
Speed	162MHz
Connectivity	EBI/EMI, I²C, UART/USART
Peripherals	DMA, WDT
Number of I/O	16
Program Memory Size	-
Program Memory Type	ROMless
EEPROM Size	-
RAM Size	4K x 8
Voltage - Supply (Vcc/Vdd)	1.65V ~ 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=mcf5407cft162

ILLUSTRATIONS

1.3.2.1 16-Kbyte Instruction Cache/8-Kbyte Data Cache

The MCF5407 Harvard architecture includes a 16-Kbyte instruction cache and an 8-Kbyte data cache. These four-way, set-associative caches provide pipelined, single-cycle access on cached instructions and operands.

As with all ColdFire caches, the cache controllers implement a non-lockup, streaming design. The use of processor-local memories decouples performance from external memory speeds and increases available bandwidth for external devices or the on-chip 4-channel DMA.

Both caches implement line-fill buffers to optimize 16-byte line burst accesses. Additionally, the data cache supports copyback, write-through, or cache-inhibited modes. A 4-entry, 32-bit buffer is used for cache line push operations and can be configured for deferred write buffering in write-through or cache-inhibited modes.

The INTOUCH instruction can be used to prefetch instructions that, when used with the cache locking feature, cannot be displaced from the instruction cache by instruction cache misses. This function may be desirable in systems where deterministic real-time performance is critical.

1.3.2.2 Internal 2-Kbyte SRAMs

Two 2-Kbyte on-chip SRAM modules are also connected to the Harvard memory architecture and provide pipelined, single-cycle access to memory regions mapped to these devices. Each memory can be independently mapped to any 0-modulo-2K location in the 4-Gbyte address space and can be configured either for instruction or data accesses. Time-critical functions can be mapped onto the instruction memory bus, while the system stack or heavily-referenced data operands can be mapped onto the data bus.

1.3.3 DRAM Controller

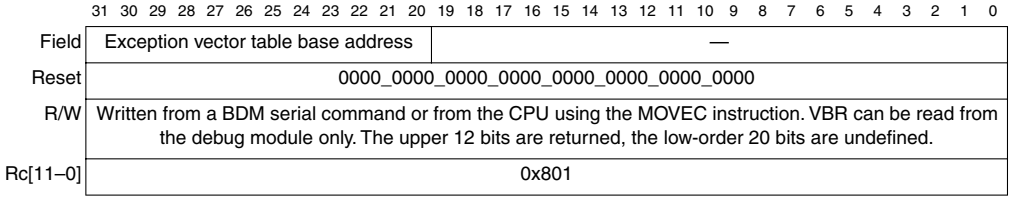
The MCF5407 DRAM controller provides a direct interface for up to two blocks of DRAM. The controller supports 8-, 16-, or 32-bit memory widths and can easily interface to PC-100 DIMMs. A unique addressing scheme allows for increases in system memory size without rerouting address lines and rewiring boards. The controller operates in normal mode or in page mode and supports SDRAMs and EDO DRAMs.

1.3.4 DMA Controller

The MCF5407 provides four fully programmable DMA channels for quick data transfer. Dual- and single-address modes support bursting and cycle steal. Data transfers are 32 bits long with packing and unpacking supported along with an auto-alignment option for efficient block transfers. Automatic block transfers from on-chip serial UARTs are also supported through the DMA channels.

2.2.2.2 Vector Base Register (VBR)

The VBR holds 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. VBR[19–0] are not implemented and are assumed to be zero, forcing the vector table to be aligned on a 0-modulo-1-Mbyte boundary.



	31 30 29 28 27 26 25 24 23 22 21 20	19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Field	Exception vector table base address	—
Reset	0000_0000_0000_0000_0000_0000_0000_0000	
R/W	Written from a BDM serial command or from the CPU using the MOVEC instruction. VBR can be read from the debug module only. The upper 12 bits are returned, the low-order 20 bits are undefined.	
Rc[11–0]	0x801	

Figure 2-6. Vector Base Register (VBR)

2.2.2.3 Cache Control Register (CACR)

The CACR controls operation of both the instruction and data cache memory. It includes bits for enabling, freezing, and invalidating cache contents. It also includes bits for defining the default cache mode and write-protect fields. See Section 4.10.1, “Cache Control Register (CACR).”

2.2.2.4 Access Control Registers (ACR0–ACR3)

The access control registers (ACR0–ACR3) define attributes for four user-defined memory regions. ACR0 and ACR1 control data memory space and ACR2 and ACR3 control instruction memory space. Attributes include definition of cache mode, write protect and buffer write enables. See Section 4.10.2, “Access Control Registers (ACR0–ACR3).”

2.2.2.5 RAM Base Address Registers (RAMBAR0 and RAMBAR1)

The RAMBAR registers determine the base address location of the internal SRAM modules and indicate the types of references mapped to each. Each RAMBAR includes a base address, write-protect bit, address space mask bits, and an enable. The RAM base address must be aligned on a 0-module-2-Kbyte boundary. See Section 4.4.1, “SRAM Base Address Registers (RAMBAR0/RAMBAR1).”

2.2.2.6 Module Base Address Register (MBAR)

The module base address register (MBAR) defines the logical base address for the memory-mapped space containing the control registers for the on-chip peripherals. See Section 6.2.2, “Module Base Address Register (MBAR).”

BSR — Branch to Subroutine — BSR

Operation: SP – 4 → SP; PC → (SP); PC + d_n → PC

Assembler Syntax: BSR <label>

Attributes: Size = byte, word, long

Description: Pushes the word address of the instruction immediately following the BSR instruction onto the system stack. The PC contains the address of the instruction word, plus two. Program execution then continues at location (PC) + displacement. The displacement is a two's complement integer that represents the relative distance in bytes from the current PC to the destination PC. If the 8-bit displacement field in the instruction word is 0, a 16-bit displacement (the word immediately following the instruction) is used. If the 8-bit displacement field in the instruction word is all ones (0xFF), the 32-bit displacement (longword immediately following the instruction) is used.

Condition Codes: Not affected

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	1	8-bit displacement							
16-bit displacement if 8-bit displacement = 0x00															
32-bit displacement if 8-bit displacement = 0xFF															

Instruction Fields:

- 8-bit displacement field—Two's complement integer specifying the number of bytes between the branch instruction and the next instruction to be executed.
- 16-bit displacement field—Used for displacement when the 8-bit displacement contains 0x00.
- 32-bit displacement field—Used for displacement when the 8-bit displacement contains 0xFF.

NOTE:

A branch to the next immediate instruction automatically uses the 16-bit displacement format because the 8-bit displacement field contains 0x00 (zero offset).

BSR	V2, V3 Core	V4 Core
Opcode present	Yes	Yes
Operand sizes supported	.b, .w	.b, .w, .l

Table 4-1. RAMBARn Field Description (Continued)

Bits	Name	Description
6	—	Reserved, should be cleared.
5–1	C/I, SC, SD, UC, UD	Address space masks (ASn). These fields allow certain types of accesses to be masked, or inhibited from accessing the SRAM module. These bits are useful for power management as described in Section 4.6, “Power Management.” In particular, C/I is typically set. The address space mask bits are follows: C/I = CPU space/interrupt acknowledge cycle mask. Note that C/I must be set if BA = 0. SC = Supervisor code address space mask SD = Supervisor data address space mask UC = User code address space mask UD = User data address space mask For each ASn bit: 0 An access to the SRAM module can occur for this address space 1 Disable this address space from the SRAM module. If a reference using this address space is made, it is inhibited from accessing the SRAM module and is processed like any other non-SRAM reference.
0	V	Valid. Enables/disables the SRAM module. V is cleared at reset. 0 RAMBAR contents are not valid. 1 RAMBAR contents are valid.

The mapping of a given access into the RAM uses the following algorithm to determine if the access hits in the memory:

```
if (RAMBAR[0] = 1)

if (((access = instructionFetch) & (RAMBAR[7] = 1)) |
        ((access = dataReference)    & (RAMBAR[7] = 0)))
                if (requested address[31:10] = RAMBAR[31:10])
        if (requested address[31:n] = RAMBAR[31:n]
                if (ASn of the requested type = 0)
                        Access is mapped to the RAM module
                        if (access = read)
                                Read the RAM and return the data
                        if (access = write)
                                if (RAMBAR[8] = 0)
                                        Write the data into the RAM
                                else Signal a write-protect access error
```

AS*n* refers to the five address space mask bits: C/I, SC, SD, UC, and UD.

4.5 SRAM Initialization

After a hardware reset, the contents of each SRAM module are undefined. The valid bits, RAMBAR*n*[V], are cleared, disabling the SRAM modules. If the SRAM requires initialization with instructions or data, the following steps should be performed:

1. Load RAMBAR*n* with bit 7 = 0, mapping the SRAM module to the desired location. Clearing RAMBAR*n*[7] logically connects the SRAM module to the processor’s data bus.

program counter register when the appropriate valid bit is set and TDR and/or XTDR are configured appropriately. PBR bits are masked by clearing corresponding PBMR bits. Results are compared with the processor's program counter register, as defined in TDR and/or XTDR. PBR1–PBR3 are not masked. Figure 5-11 shows the PC breakpoint register.

	31	1	0
Field	Program Counter		V [1]
Reset	—		0
R/W	Write. PC breakpoint registers are accessible in supervisor mode using the WDEBUG instruction and through the BDM port using the RDMREG and WDMREG commands using values shown in Section 5.5.3.3, "Command Set Descriptions."		
DRc[4–0]	0x08 (PBR); 0x18 (PBR1); 0x1A (PBR2); 0x1B (PBR3)		

[1] PBR does not have a valid bit. PBR[0] is read as 0 and should be cleared.

Figure 5-11. Program Counter Breakpoint Registers (PBR, PBR1, PBR2, PBR3)

Table 5-15 describes PBR, PBR1, PBR2, and PBR3 fields.

Table 5-15. PBR, PBR1, PBR2, PBR3 Field Descriptions

Bits	Name	Description
31–1	Address	PC breakpoint address. The 31-bit address to be compared with the PC as a breakpoint trigger. PBR does not have a valid bit.
0	V	Valid. Breakpoint registers are compared with the processor's program counter register when the appropriate valid bit is set and TDR and/or XTDR are configured appropriately. This bit is not implemented on PBR.

PBMR is accessible in supervisor mode as debug control register 0x09 using the WDEBUG instruction and via the BDM port using the WDMREG command. Figure 5-12 shows PBMR.

	31	1	0
Field	Mask		—
Reset	—		—
R/W	Write. PBMR is accessible in supervisor mode as debug control register 0x09 using the WDEBUG instruction and via the BDM port using the wdmreg command.		
DRc[4–0]	0x09		

Figure 5-12. Program Counter Breakpoint Mask Register (PBMR)

Table 5-16 describes PBMR fields.

Table 5-16. PBMR Field Descriptions

Bits	Name	Description
31–0	Mask	PC breakpoint mask. A zero in a bit position causes the corresponding PBR bit to be compared to the appropriate PC bit. Set PBMR bits cause PBR bits to be ignored.

5.5.3.3.2 Write A/D Register (WAREG/WDREG)

The operand longword data is written to the specified address or data register. A write alters all 32 register bits. A bus error response is returned if the CPU core is not halted.

Command Format:

	15 12	11 8	7 4	3	2 0
Command	0x2	0x0	0x8	A/D	Register
Result	D[31:16]				
	D[15:0]				

Figure 5-22. WAREG/WDREG Command Format

Command Sequence

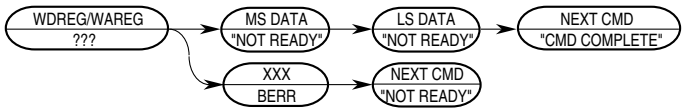


Figure 5-23. WAREG/WDREG Command Sequence

Operand Data — Longword data is written into the specified address or data register. The data is supplied most-significant word first.

Result Data — Command complete status is indicated by returning 0xFFFF (with S cleared) when the register write is complete.

5.5.3.3.10 Read Control Register (RCREG)

Read the selected control register and return the 32-bit result. Accesses to the processor/memory control registers are always 32 bits wide, regardless of register width. The second and third words of the command form a 32-bit address, which the debug module uses to generate a special bus cycle to access the specified control register. The 12-bit Rc field is the same as that used by the MOVEC instruction.

Command/Result Formats:

	15 12	11 8	7 4	3 0
Command	0x2	0x9	0x8	0x0
	0x0	0x0	0x0	0x0
	0x0	Rc		
Result	D[31:16]			
	D[15:0]			

Figure 5-38. RCREG Command/Result Formats

Rc encoding:

Table 5-23. Control Register Map

Rc	Register Definition
0x002	Cache control register (CACR)
0x004	Access control register 0 (ACR0)
0x005	Access control register 1 (ACR1)
0x006	Access control register 2 (ACR2)
0x007	Access control register 2 (ACR3)
0x801	Vector base register (VBR)
0x804	MAC status register (MACSR)

Rc	Register Definition
0x805	MAC mask register (MASK)
0x806	MAC accumulator (ACC)
0x80E	Status register (SR)
0x80F	Program register (PC)
0xC04	RAM base address register (RAMBAR0)
0xC05	RAM base address register (RAMBAR1)
0xC0F	Memory base address (MBAR)

Command Sequence:

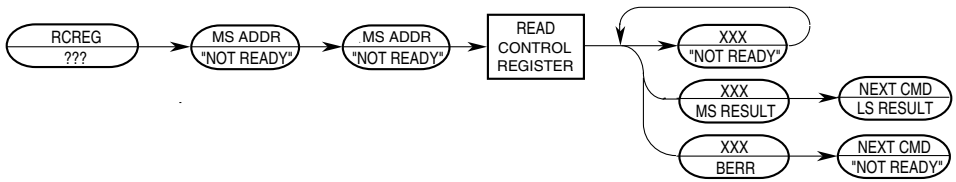


Figure 5-39. RCREG Command Sequence

Operand Data: The only operand is the 32-bit Rc control register select field.

Result Data: Control register contents are returned as a longword, most-significant word first. The implemented portion of registers smaller than 32 bits is guaranteed correct; other bits are undefined.

Chapter 6
SIM Overview

This chapter provides detailed operation information regarding the system integration module (SIM). It describes the SIM programming model, bus arbitration, and system-protection functions for the MCF5407.

6.1 Features

The SIM, shown in Figure 6-1, provides overall control of the bus and serves as the interface between the ColdFire core processor complex and the internal peripheral devices.

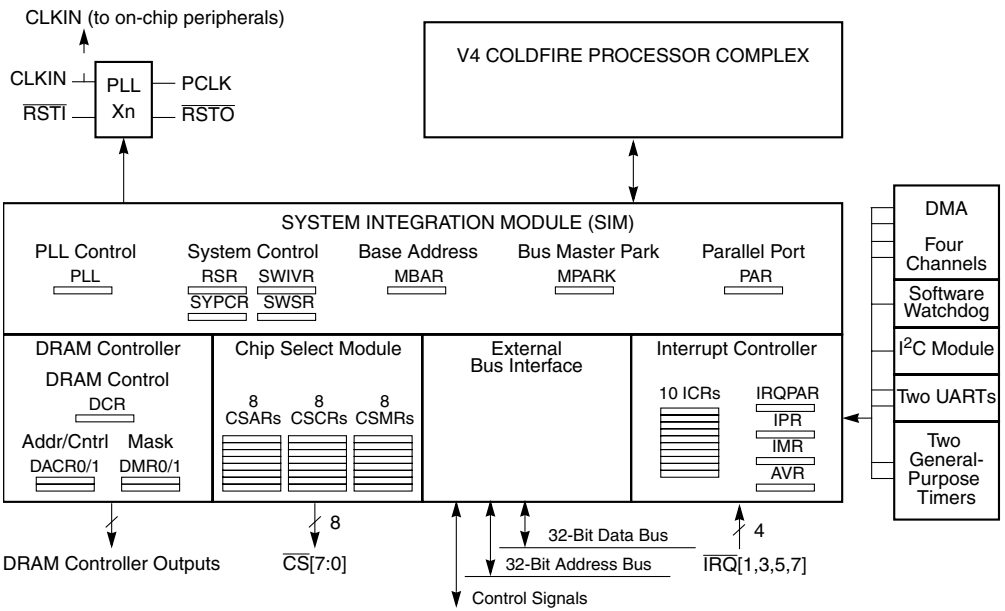


Figure 6-1. SIM Block Diagram

6.2 Programming Model

The following sections describe the registers incorporated into the SIM.

6.2.1 SIM Register Memory Map

Table 6-1 shows the memory map for the SIM registers. The internal registers in the SIM are memory-mapped registers offset from the MBAR address pointer defined in MBAR[BA]. This supervisor-level register is described in Section 6.2.2, “Module Base Address Register (MBAR).” Because SIM registers depend on the base address defined in MBAR[BA], MBAR must be programmed before SIM registers can be accessed.

NOTE:

Although external masters cannot access the MCF5407’s on-chip memories or MBAR, they can access any of the SIM memory map and peripheral registers, such as those belonging to the interrupt controller, chip-select module, UARTs, timers, DMA, and I^2C.

Table 6-1. SIM Registers

MBAR Offset	[31:24]	[23:16]	[15:8]	[7:0]
0x000	Reset status register (RSR) [p. 6-5]	System protection control register (SYPCR) [p. 6-8]	Software watchdog interrupt vector register (SWIVR) [p. 6-9]	Software watchdog service register (SWSR) [p. 6-9]
0x004	Pin assignment register (PAR) [p. 6-10]		Interrupt port assignment register (IRQPAR) [p. 9-7]	Reserved
0x008	PLL control (PLLCR) [p. 7-3]	Reserved		
0x00C	Default bus master park register (MPARK) [p. 6-11]	Reserved		
0x010–0x03C	Reserved			
Interrupt Controller Registers [p. 9-2]				
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]				

6.2.10 Bus Arbitration Control

This section describes the bus arbitration register and the four arbitration schemes.

6.2.10.1 Default Bus Master Park Register (MPARK)

The MPARK, shown in Figure 6-9, determines the default bus master arbitration between internal transfers (core and DMA module) and between internal and external transfers to internal resources. This arbitration is needed because external masters can access internal registers within the MCF5407 peripherals.

	7	6	5	4	3	2	0
Field	PARK		IARBCTRL	EARBCTRL	SHOWDATA	—	
Reset	0000_0000						
R/W	R/W						
Address	MBAR + 0x0C						

Figure 6-9. Default Bus Master Register (MPARK)

Table 6-6 describes MPARK bits.

Table 6-6. MPARK Field Descriptions

Bits	Name	Description
7–6	PARK	Park. Indicates the arbitration priority of internal transfers among MCF5407 resources. 00 Round-robin between DMA and ColdFire core 01 Park on master ColdFire core 10 Park on master DMA module 11 Park on current master Use of this field is described in detail in Section 6.2.10.1.1, “Arbitration for Internally Generated Transfers (MPARK[PARK]).”
5	IARBCTRL	Internal bus arbitration control. Controls external device access to the MCF5407 internal bus. 0 Arbitration disabled (single-master system) 1 Arbitration enabled. IARBCTRL must be set if external masters are using internal resources like the DRAM controller or chip selects. Use of this bit depends on whether the system has single or multiple masters, as follows: • In a single-master system, IARBCTRL should stay cleared, disabling internal arbitration by external masters. In this scenario, MPARK[PARK] applies only to priority of internal masters over one another. Note that the internal DMA (master 3) has priority over the ColdFire core (master 2), if internal DMA bandwidth is at its maximum (BWC = 000). • In multiple master systems that expect to use internal resources like the DRAM controller or chip selects, internal arbitration should be enabled. The external master defaults to the highest priority internal master anytime $\overline{BG}$ is negated.
4	EARBCTRL	External bus arbitration control. Enables internal register memory space to external bus arbitration. Internal registers are those accessed at offsets to the MBAR. These include the SIM, DMA, chip selects, timers, UARTs, I^2C, and parallel port registers. These registers do not include the MBAR; only the core can access the MBAR. 0 Arbitration disabled 1 Arbitration enabled The use of this field is described in detail in Section 6.2.10.1.2, “Arbitration between Internal and External Masters for Accessing Internal Resources.”

Table 11-13 describes DACR*n* fields.

Table 11-13. DACR0/DACR1 Field Descriptions (Synchronous Mode)

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>31–18</td><td>BA</td><td>Base address register. With DCMR[BAM], determines the address range in which the associated DRAM block is located. Each BA bit is compared with the corresponding address of the current bus cycle. If all unmasked bits match, the address hits in the associated DRAM block. BA functions the same as in asynchronous operation.</td></tr>
<tr><td>17–16</td><td>—</td><td>Reserved, should be cleared.</td></tr>
<tr><td>15</td><td>RE</td><td>Refresh enable. Determines when the DRAM controller generates a refresh cycle to the DRAM block.
0 Do not refresh associated DRAM block
1 Refresh associated DRAM block</td></tr>
<tr><td>14</td><td>—</td><td>Reserved, should be cleared.</td></tr>
<tr><td>13–12</td><td>CASL</td><td>$\overline{\text{CAS}}$ latency. Affects the following SDRAM timing specifications. Timing nomenclature varies with manufacturers. Refer to the SDRAM specification for the appropriate timing nomenclature:
<table>
<tr><th rowspan="2">Parameter</th><th colspan="4">Number of Bus Clocks</th></tr>
<tr><th>CASL= 00</th><th>CASL = 01</th><th>CASL= 10</th><th>CASL= 11</th></tr>
<tr><td>t_{RCD}—$\overline{\text{SRAS}}$ assertion to $\overline{\text{SCAS}}$ assertion</td><td>1</td><td>2</td><td>3</td><td>3</td></tr>
<tr><td>t_{CASL}—$\overline{\text{SCAS}}$ assertion to data out</td><td>1</td><td>2</td><td>3</td><td>3</td></tr>
<tr><td>t_{RAS}—ACTV command to precharge command</td><td>2</td><td>4</td><td>6</td><td>6</td></tr>
<tr><td>t_{RP}—Precharge command to ACTV command</td><td>1</td><td>2</td><td>3</td><td>3</td></tr>
<tr><td>t_{RWL},t_{RDL}—Last data input to precharge command</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>t_{EP}—Last data out to precharge command)</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table></td></tr>
<tr><td>11</td><td>—</td><td>Reserved, should be cleared.</td></tr>
<tr><td>10–8</td><td>CBM</td><td>Command and bank MUX [2:0]. Because different SDRAM configurations cause the command and bank select lines to correspond to different addresses, these resources are programmable. CBM determines the addresses onto which these functions are multiplexed.
CBM Command Bit Bank Select Bits
000 17 18 and up
001 18 19 and up
010 19 20 and up
011 20 21 and up
100 21 22 and up
101 22 23 and up
110 23 24 and up
111 24 25 and up
This encoding and the address multiplexing scheme handle common SDRAM organizations. Bank select bits include a base bit and all address bits above for SDRAMs with multiple bank select bits.</td></tr>
<tr><td>7</td><td>—</td><td>Reserved, should be cleared.</td></tr>
</table>

- Single-address transfers—An external device can initiate a single-address transfer by asserting $\overline{\text{DREQ}}$. The MCF5407 provides address and control signals for single-address transfers. The external device reads to or writes from the specified address, as Figure 12-4 shows. External logic is required.

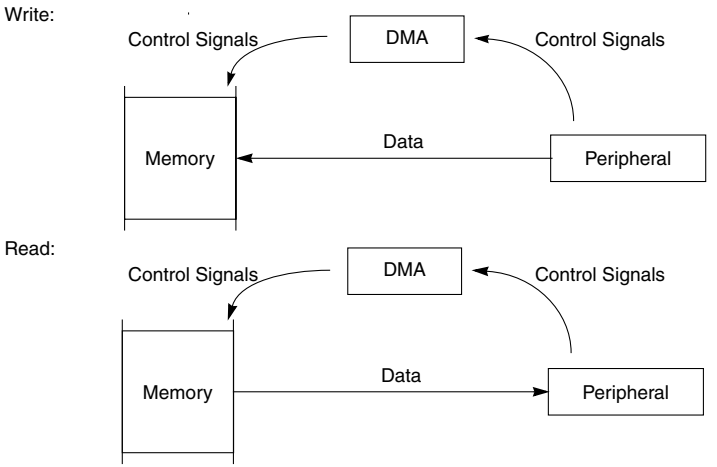


Figure 12-4. Single-Address Transfers

Any operation involving the DMA module follows the same three steps:

1. Channel initialization—Channel registers are loaded with control information, address pointers, and a byte-transfer count.
2. Data transfer—The DMA accepts requests for operand transfers and provides addressing and bus control for the transfers.
3. Channel termination—Occurs after the operation is finished, either successfully or due to an error. The channel indicates the operation status in the channel's DSR, described in Section 12.4.5, "DMA Status Registers (DSR0–DSR3)."

12.4 DMA Controller Module Programming Model

This section describes each internal register and its bit assignment. Note that there is no way to prevent a write to a control register during a DMA transfer. Table 12-3 shows the mapping of DMA controller registers.

12.4.1 Source Address Registers (SAR0–SAR3)

SAR*n*, Figure 12-5, contains the address from which the DMA controller requests data. In single-address mode, SAR*n* provides the address regardless of the direction.

	31	0
Field	SAR	
Reset	0000_0000_0000_0000_0000_0000_0000_0000	
R/W	R/W	
Address	MBAR + 0x300, 0x340, 0x380, 0x3C0	

Figure 12-5. Source Address Registers (SARn)

NOTE:

SAR/DAR address ranges cannot be programmed to on-chip SRAM because it cannot be accessed by on-chip DMA.

12.4.2 Destination Address Registers (DAR0–DAR3)

For dual-address transfers only, DAR*n*, Figure 12-6, holds the address to which the DMA controller sends data.

	31	0
Field	DAR	
Reset	0000_0000_0000_0000_0000_0000_0000_0000	
R/W	R/W	
Address	MBAR + 304, 0x344, 0x384, 0x3C4	

Figure 12-6. Destination Address Registers (DARn)

NOTE:

On-chip DMAs do not maintain coherency with MCF5407 caches and so must not transfer data to cacheable memory.

12.4.3 Byte Count Registers (BCR0–BCR3)

BCR*n*, Figure 12-7, holds the number of bytes yet to be transferred for a given block. BCR*n* decrements on the successful completion of the address transfer of either a write transfer in dual-address mode or any transfer in single-address mode. BCR*n* decrements by 1, 2, 4, or 16 for byte, word, longword, or line accesses, respectively.

Table 13-4. Time-Out Values (in Seconds)—TRR[REF] = 0xFFFF (162-MHz Processor Clock) (Continued)

TMR[PS] (Dec)	CLK = 10 (÷ 1)			CLK = 01 (÷ 16)		
	CLKIN (MHz)					
	54	40.5	32.4	54	40.5	32.4
6	0.136	0.181	0.227	0.008	0.011	0.014
7	0.155	0.207	0.259	0.010	0.013	0.016
8	0.175	0.233	0.291	0.011	0.015	0.018
9	0.194	0.259	0.324	0.012	0.016	0.020
10	0.214	0.285	0.356	0.013	0.018	0.022
11	0.233	0.311	0.388	0.015	0.019	0.024
12	0.252	0.337	0.421	0.016	0.021	0.026
13	0.272	0.362	0.453	0.017	0.023	0.028
14	0.291	0.388	0.485	0.018	0.024	0.030
15	0.311	0.414	0.518	0.019	0.026	0.032
16	0.330	0.440	0.550	0.021	0.028	0.034
17	0.350	0.466	0.583	0.022	0.029	0.036
18	0.369	0.492	0.615	0.023	0.031	0.038
19	0.388	0.518	0.647	0.024	0.032	0.040
20	0.408	0.544	0.680	0.025	0.034	0.042
21	0.427	0.570	0.712	0.027	0.036	0.044
22	0.447	0.595	0.744	0.028	0.037	0.047
23	0.466	0.621	0.777	0.029	0.039	0.049
24	0.485	0.647	0.809	0.030	0.040	0.051
25	0.505	0.673	0.841	0.032	0.042	0.053
26	0.524	0.699	0.874	0.033	0.044	0.055
27	0.544	0.725	0.906	0.034	0.045	0.057
28	0.563	0.751	0.939	0.035	0.047	0.059
29	0.583	0.777	0.971	0.036	0.049	0.061
30	0.602	0.803	1.003	0.038	0.050	0.063
31	0.621	0.829	1.036	0.039	0.052	0.065
32	0.641	0.854	1.068	0.040	0.053	0.067
33	0.660	0.880	1.100	0.041	0.055	0.069
34	0.680	0.906	1.133	0.042	0.057	0.071
35	0.699	0.932	1.165	0.044	0.058	0.073
36	0.718	0.958	1.197	0.045	0.060	0.075
37	0.738	0.984	1.230	0.046	0.061	0.077
38	0.757	1.010	1.262	0.047	0.063	0.079
39	0.777	1.036	1.295	0.049	0.065	0.081
40	0.796	1.062	1.327	0.050	0.066	0.083
41	0.816	1.087	1.359	0.051	0.068	0.085
42	0.835	1.113	1.392	0.052	0.070	0.087

TMR[PS] (Dec)	CLK = 10 (÷ 1)			CLK = 01 (÷ 16)		
	CLKIN (MHz)					
	54	40.5	32.4	54	40.5	32.4
134	2.621	3.495	4.369	0.164	0.218	0.273
135	2.641	3.521	4.401	0.165	0.220	0.275
136	2.660	3.547	4.434	0.166	0.222	0.277
137	2.680	3.573	4.466	0.167	0.223	0.279
138	2.699	3.599	4.499	0.169	0.225	0.281
139	2.719	3.625	4.531	0.170	0.227	0.283
140	2.738	3.651	4.563	0.171	0.228	0.285
141	2.757	3.676	4.596	0.172	0.230	0.287
142	2.777	3.702	4.628	0.174	0.231	0.289
143	2.796	3.728	4.660	0.175	0.233	0.291
144	2.816	3.754	4.693	0.176	0.235	0.293
145	2.835	3.780	4.725	0.177	0.236	0.295
146	2.854	3.806	4.757	0.178	0.238	0.297
147	2.874	3.832	4.790	0.180	0.239	0.299
148	2.893	3.858	4.822	0.181	0.241	0.301
149	2.913	3.884	4.855	0.182	0.243	0.303
150	2.932	3.910	4.887	0.183	0.244	0.305
151	2.952	3.935	4.919	0.184	0.246	0.307
152	2.971	3.961	4.952	0.186	0.248	0.309
153	2.990	3.987	4.984	0.187	0.249	0.311
154	3.010	4.013	5.016	0.188	0.251	0.314
155	3.029	4.039	5.049	0.189	0.252	0.316
156	3.049	4.065	5.081	0.191	0.254	0.318
157	3.068	4.091	5.113	0.192	0.256	0.320
158	3.087	4.117	5.146	0.193	0.257	0.322
159	3.107	4.143	5.178	0.194	0.259	0.324
160	3.126	4.168	5.211	0.195	0.261	0.326
161	3.146	4.194	5.243	0.197	0.262	0.328
162	3.165	4.220	5.275	0.198	0.264	0.330
163	3.185	4.246	5.308	0.199	0.265	0.332
164	3.204	4.272	5.340	0.200	0.267	0.334
165	3.223	4.298	5.372	0.201	0.269	0.336
166	3.243	4.324	5.405	0.203	0.270	0.338
167	3.262	4.350	5.437	0.204	0.272	0.340
168	3.282	4.376	5.469	0.205	0.273	0.342
169	3.301	4.401	5.502	0.206	0.275	0.344
170	3.320	4.427	5.534	0.208	0.277	0.346

$$\text{Baudrate} = \frac{54\text{MHz}}{[32 \times \text{divider}]}$$

Let baud rate = 9600; the divider can be calculated as follows:

$$\text{Divider} = \frac{54\text{MHz}}{[32 \times 9600]} = 176(\text{decimal}) = 0\text{x}00\text{B}0$$

Therefore UDU*n* = 0x00 and UDL*n* = 0xB0.

14.5.1.2.2 External Clock

An external source clock (TIN) can be used as is or divided by 16.

$$\text{Baudrate} = \frac{\text{Externalclockfrequency}}{16\text{or}1}$$

14.5.2 Transmitter and Receiver Operating Modes

Figure 14-29 is a functional block diagram of the transmitter and receiver showing the command and operating registers, which are described generally in the following sections and described in detail in Section 14.3, “Register Descriptions.”

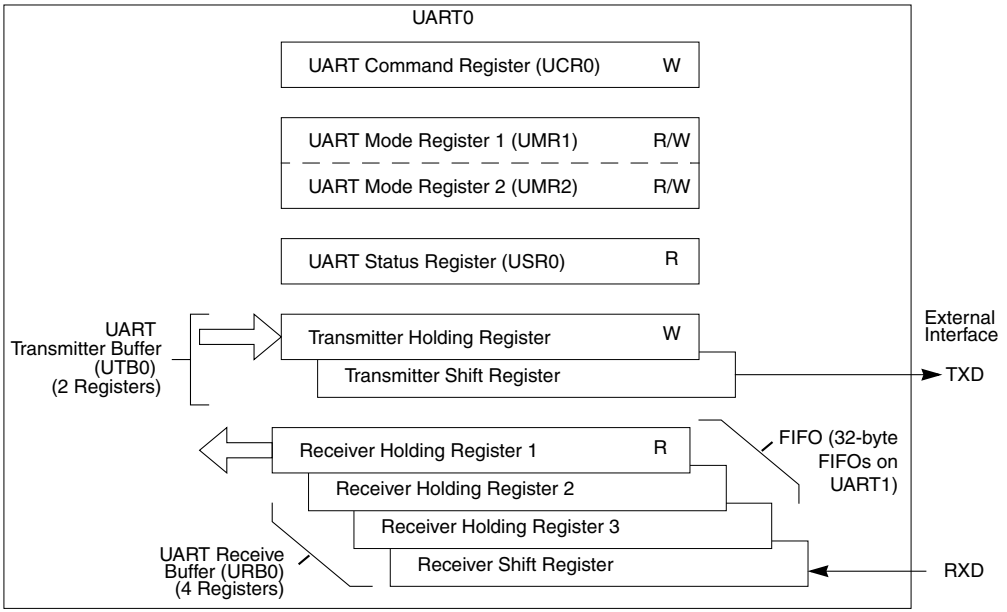


Figure 14-29. Transmitter and Receiver Functional Diagram

if a longword access occurs at a misaligned offset of 0x1, a byte is transferred first (SIZ[1:0] = 01), a word is next transferred at offset 0x2 (SIZ[1:0] = 10), then the final byte is transferred at offset 0x4 (SIZ[1:0] = 01).

For aligned transfers larger than the port size, SIZ[1:0] behaves as follows:

- If bursting is used, SIZ[1:0] stays at the size of transfer.
- If bursting is inhibited, SIZ[1:0] first shows the size of the transfer and then shows the port size.

Table 17-4. Bus Cycle Size Encoding

SIZ[1:0]	Port Size
00	Longword
01	Byte
10	Word
11	Line

For burst-inhibited transfers, SIZ[1:0] changes with each $\overline{TS}$ assertion to reflect the next transfer size. For transfers to port sizes smaller than the transfer size, SIZ[1:0] indicates the size of the entire transfer on the first access and the size of the current port transfer on subsequent transfers. For example, for a longword write to an 8-bit port, SIZ[1:0] = 00 for the first byte transfer and 01 for the next three.

17.2.5 Transfer Start ($\overline{TS}$)

The MCF5407 asserts $\overline{TS}$ during the first clock cycle when address and attributes (TM, TT, $\overline{TIP}$, R/$\overline{W}$, and SIZ) are valid. $\overline{TS}$ is negated in the following clock cycle. When the MCF5407 is not the bus master, $\overline{TS}$ is an input.

17.2.6 Address Strobe ($\overline{AS}$)

Address strobe ($\overline{AS}$) is asserted to indicate when the address is stable at the start of a bus cycle. The address and attributes are guaranteed to be valid during the entire period that $\overline{AS}$ is asserted. This signal is asserted and negated on the falling edge of the clock. When the MCF5407 is not the bus master, $\overline{AS}$ is an input.

17.2.7 Transfer Acknowledge ($\overline{TA}$)

When the MCF5407 is bus master, the external system drives this input to terminate the bus transfer. The bus continues to be driven until this synchronous signal is asserted. For write cycles, the processor continues to drive data one clock after $\overline{TA}$ is asserted. During read cycles, the peripheral must continue to drive data until $\overline{TA}$ is recognized.

If all bus cycles support fast termination, $\overline{TA}$ can be tied low. However, note that $\overline{TA}$ cannot be tied low if potential external bus masters are present. The MCF5407 drives $\overline{TA}$ for an

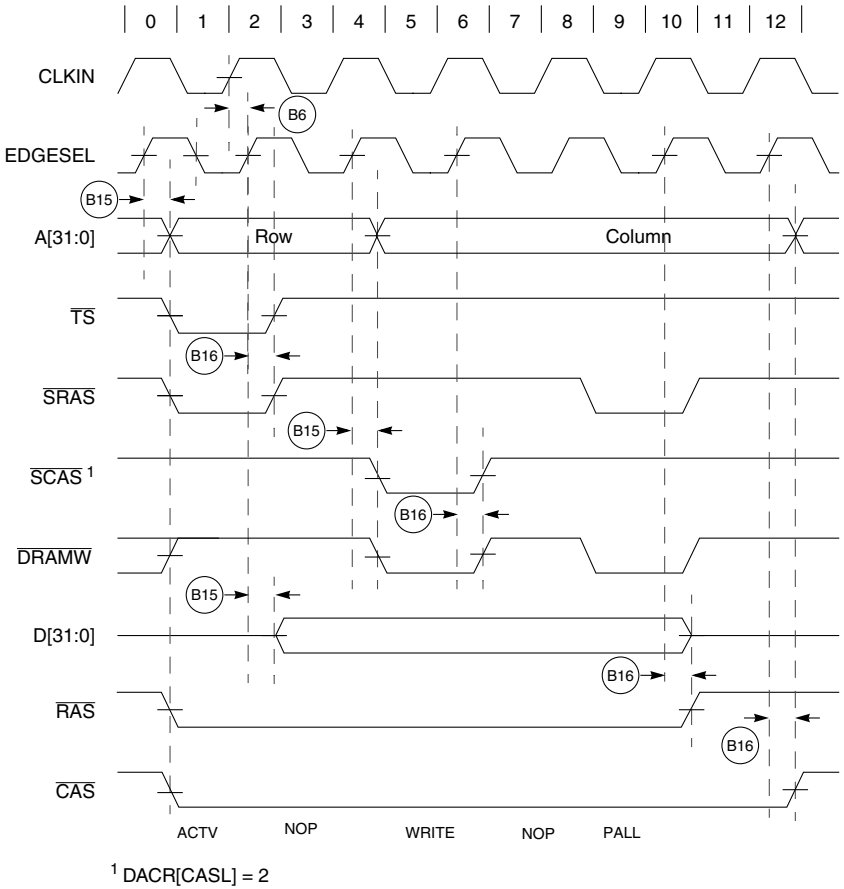


Figure 20-8. SDRAM Write Cycle with EDGESEL Tied to Buffered CLKIN

Figure 20-9 shows an SDRAM read cycle with EDGESEL tied high.