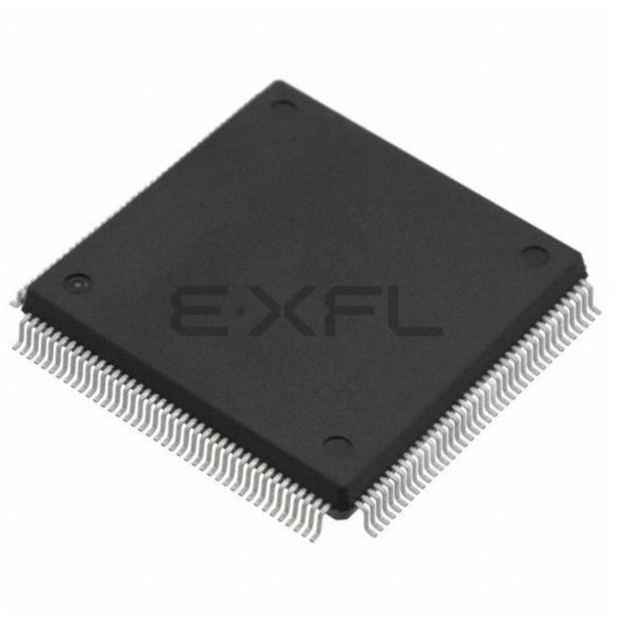


Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Obsolete
Core Processor	Coldfire V2
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	16MHz
Co-Processors/DSP	-
RAM Controllers	DRAM
Graphics Acceleration	-
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5V
Operating Temperature	0°C ~ 70°C (TA)
Security Features	-
Package / Case	160-BQFP
Supplier Device Package	160-QFP (28x28)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mcf5206ab16a

1.3.11 System Protection

The MCF5206 processor contains a 16-bit software watchdog timer with an 8-bit prescaler. The programmable software watchdog timer provides either a level 7 interrupt or a hardware reset on timeout. The MCF5206 processor also contains a reset status register that indicates the cause of the last reset.

1.3.12 JTAG

To help with system diagnostics and manufacturing testing, the MCF5206 processor includes dedicated user-accessible test logic that complies with the IEEE 1149.1 standard for boundary scan testability, often referred to as Joint Test Action Group, or JTAG. For more information, refer to the IEEE 1149.1 standard.

1.3.13 System Debug Interface

The ColdFire processor core debug interface supports real-time trace and Background-Debug Mode. A four-pin Background Debug Mode (BDM) interface provides system debug. The BDM is a proper subset of the BDM interface provided on Motorola's 683XX Family of parts.

In real-time trace, four status lines provide information on processor activity in real time (PST pins). A 4-bit wide debug data bus (DDATA) displays operand data, which helps track the machine's dynamic execution path as the change-of-flow instructions execute. These signals are multiplexed with an 8-bit parallel port for application development, which does not use real-time trace.

1.3.14 Pinout and Package

The MCF5206 device is supplied in a 160-pin plastic quad flat pack package.

Clock 3 (C3)

The MCF5206 increments A0 to address the next byte of the word transfer. The selected device(s) places the second byte of the addressed data onto D[31:24] and asserts the transfer acknowledge ($\overline{\text{TA}}$). At the end of C3, the MCF5206 samples the level of $\overline{\text{TA}}$ and if $\overline{\text{TA}}$ is asserted, latches the current value of D[31:24]. If $\overline{\text{TA}}$ is asserted, the transfer of the word read is complete and the transfer is terminated. If $\overline{\text{TA}}$ is negated, the MCF5206 continues to sample $\overline{\text{TA}}$ and inserts wait states instead of terminating the transfer. The MCF5206 continues to sample $\overline{\text{TA}}$ on successive rising edges of CLK until it is asserted.

6.5.3 Bursting Write Transfers: Word, Longword, and Line

The MCF5206 uses line-write transfers to access a 16-byte operand for a MOVEM instruction, when appropriate. A line write accesses a block of four longwords, aligned to a longword memory boundary, by supplying a starting address that points to one of the longwords and increments A3, A2, A1, and A0 of the supplied address for each transfer. A longword write accesses a single longword aligned to a longword boundary and increments A1 and A0 if the accessed port size is smaller than 32 bits. A word write accesses a single word of data, aligned to a word boundary and increments A0 if the accessed port size is smaller than 16 bits. Table 6-9 lists the encodings for the SIZx bits for each port size for transfers where bursting is both enabled and disabled.

Table 6-9. SIZx Encoding for Burst- and Bursting-Inhibited Ports

OPERAND SIZE	32-BIT PORT				16 -BIT PORT				8-BIT PORT			
	BURSTING ENABLED		BURSTING INHIBITED		BURSTING ENABLED		BURSTING INHIBITED		BURSTING ENABLED		BURSTING INHIBITED	
	SIZ1	SIZ0	SIZ1	SIZ0	SIZ1	SIZ0	SIZ1	SIZ0	SIZ1	SIZ0	SIZ1	SIZ0
BYTE	0	1	0	1	0	1	0	1	0	1	0	1
WORD	1	0	1	0	1	0	1	0	1	0	0	1
LONGWORD	0	0	0	0	0	0	1	0	0	0	0	1
LINE	1	1	0	0	1	1	1	0	1	1	0	1

Figure 6-10 is a flowchart for bursting write transfers to 8-, 16-, or 32-bit ports. Bus operations are similar for each case and vary only with the size indicated, the portion of the data bus used for the transfer and the specific number of cycles needed for each transfer. A bursted write transfer can be from two to sixteen transfers long. The flowchart in Figure 6-10 is for a bursting transfer of four transfers long.

Figure 6-11 shows a user data bursting line-write transfer to a 32-bit port.

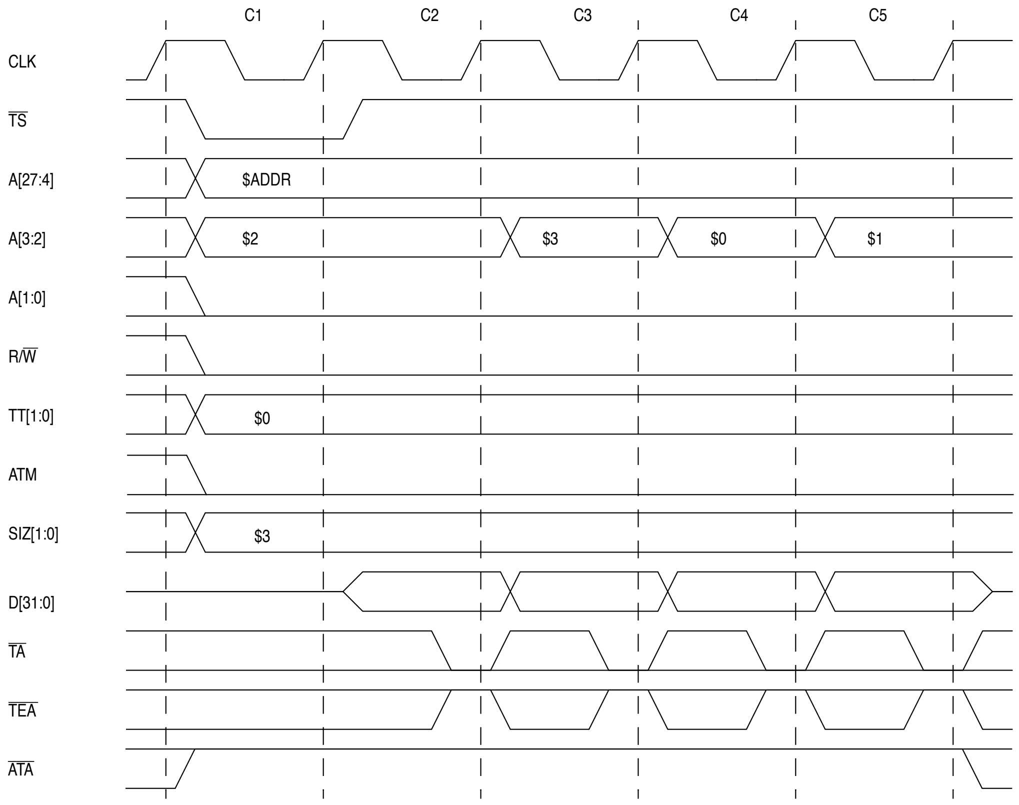


Figure 6-11. Line-Write Transfer to a 32-Bit Port (No Wait States)

Clock 1 (C1)

The write cycle starts in C1. During C1, the MCF5206 places valid values on the address bus (A[27:0]) and transfer control signals. The transfer type (TT[1:0]) signals identify the specific access type. Access type and mode (ATM) identifies the transfer as data. The read/write (R/W) signal is driven low for a write cycle, and the size signals (SIZ[1:0]) are driven to $3 to indicate a line transfer. The MCF5206 asserts transfer start (TS) to indicate the beginning of a bus cycle.

Clock 2 (C2)

During C2, the MCF5206 negates TS, drives ATM low to identify the transfer as user, and places the data on the data bus (D[31:0]). The selected device(s) asserts the transfer acknowledge (TA) if it is ready to latch the data. At the end of C2, the selected device latches the current value of D[31:0], and the MCF5206 samples the level of TA. If TA is asserted, the transfer of the first longword is complete. If TA is negated, the MCF5206

Table 6-9. MCF5206 Two-Wire Bus Arbitration Protocol Transition Conditions

AM OWN	D1	N	N	N	-	-	-	-	EM Own
	D2	N	N	A	A	-	-	-	Explicit Own
	D3	N	N	A	N	N	-	-	Implicit Own
	D4	N	N	A	N	A	-	-	Explictit Own

NOTES
1)“N” means negated; “A” means asserted; “EM” means external master.
2)End of Cycle: Whatever terminates a bus transaction whether it is normal or bus error. Note that bus cycles that result from a burst inhibited transfer are considered part of that original transfer.

Table 6-10. MCF5206 Two-Wire Arbitration Protocol State Diagram

STATE	OWN	BUS STATUS	BD
Reset	No	Not Driven	Negated
Implicit Own	Yes	Not Driven	Negated
Explicit Own	Yes	Driven	Asserted
Am Own	No	Not Driven	Negated

The MCF5206 can be in any one of four arbitration states during bus operation: reset, external master ownership, implicit ownership, or explicit ownership.

The MCF5206 enters the reset state whenever $\overline{\text{RSTI}}$ or software watchdog reset is asserted in any bus arbitration state. When $\overline{\text{RSTI}}$ and the software watchdog reset are negated, the MCF5206 proceeds to the implicit ownership state or external master ownership state, depending on $\overline{\text{BG}}$.

The external master ownership state denotes the MCF5206 does not have ownership ($\overline{\text{BG}}$ negated) of the bus and the MCF5206 does not drive the bus. The MCF5206 can assert memory control signals (i.e., $\overline{\text{CS}}$[7:0], $\overline{\text{WE}}$[3:0], $\overline{\text{RAS}}$[1:0] or $\overline{\text{CAS}}$[3:0]) and transfer acknowledge ($\overline{\text{TA}}$) during this state.

The implicit ownership state indicates that the MCF5206 owns the bus because $\overline{\text{BG}}$ is asserted to it. The MCF5206, however, is not ready to begin a bus cycle and the bus lock bit in the SIM Configuration Register (SIMR) is cleared. In this case, the MCF5206 keeps the bus three-stated until an internal bus request occurs or the bus lock bit in the SIMR is set to 1.

The MCF5206 explicitly owns the bus when the bus is granted to it ($\overline{\text{BG}}$ asserted) and at least one bus cycle has been initiated or the bus lock bit in the SIMR is set to 1. The MCF5206 asserts $\overline{\text{BD}}$ in this state to indicate the MCF5206 has explicit ownership of the bus. Until $\overline{\text{BG}}$ is negated, the MCF5206 retains explicit ownership of the bus whether or not active bus cycles are being executed. Once $\overline{\text{BG}}$ is negated and the bus lock bit in the SIMR is cleared, the MCF5206 relinquishes the bus at the end of the current bus cycle. When the MCF5206 is ready to relinquish the bus, it negates $\overline{\text{BD}}$ and three-states the bus signals.

(CSMRs), and DRAM Controller Mask Register (DCMRs), looking for a match.

The priority is listed in Table 8-3 (from highest priority to lowest priority):

Table 8-3. Chip select, DRAM and Default Memory Address Decoding Priority

Chip select 0	Highest Priority
Chip select 1	
Chip select 2	
Chip select 3	
Chip select 4	
Chip select 5	
Chip select 6	
Chip select 7	
DRAM Bank 0	
DRAM Bank 1	
Default Memory	Lowest Priority

The MCF5206 compares the address and mask in chip select 0 - 7 (chip select 0 is compared first), then the address and mask in DRAM 0 - 1. If the address does not match in either or these, the MCF5206 uses the control bits in the Default Memory Control Register (DMCR) to control the bus transfer. If the Default Memory Control Register (DMCR) control bits are used, no chip select or DRAM control signals are asserted during the transfer.

8.3.1.2 ACCESS PERMISSIONS. Chip select accesses can be restricted based on transfer direction and attributes. Each chip select can be enabled for read and/or write transfers using the WR and RD bits in the CSCRs. Each chip select can have supervisor data, supervisor code, user data, user code, and only chip select 1 can have CPU space (including interrupt acknowledge) transfers masked from their address space using the SD, SC, UD, UC, and C/I bits in the CSMRs. The transfer address must match, the transfer direction must be enabled, and transfer attributes must be unmasked for a chip select to assert.

8.3.1.3 CONTROL FEATURES. The chip select control registers and the default memory control register are used to program timing and assertion features of the chip select signals. The chip select control register provides the following programmable control features:

- Port size (8-, 16- or 32-bit)
- Number of internal wait states (0 - 15)
- Enable assertion of internal transfer acknowledge
- Enable assertion of transfer acknowledge for alternate master transfers

- Enable burst transfers
- Address setup
- Address hold
- Enable read and/or write transfers

8.3.1.3.1 8-, 16-, and 32-Bit Port Sizing. The general-purpose chip-selects support static bus sizing. You can program the size of the port controlled by a chip select. Defined 8 bit ports are connected to D[31:24]; defined 16-bit ports are connected to D[31:16]; and defined 32 bit ports are connected to D[31:0]. The port size is specified by the PS bits in the CSCR.

8.3.1.3.2 Termination. The general-purpose chip-selects support three methods of termination: internal termination, synchronous termination, and asynchronous termination. You can program the number of wait states required for each chip select and the default memory individually. You can also enable internal termination for MCF5206-initiated transfers for each chip select and default memory individually.

Transfer acknowledge ($\overline{\text{TA}}$) can synchronously terminate a transfer. If the MCF5206 initiates a bus transfer and internal termination is enabled (but $\overline{\text{TA}}$ is asserted before the specified number of wait states have been inserted), the transfer terminates on the CLK cycle where $\overline{\text{TA}}$ is asserted.

NOTE

If an alternate master initiates a bus transfer and internal termination is enabled, $\overline{\text{TA}}$ should not be driven by an external device. The MCF5206 drives $\overline{\text{TA}}$ throughout the alternate master access.

Asynchronous transfer acknowledge ($\overline{\text{ATA}}$) can asynchronously terminate a chip select or default memory transfer. If the MCF5206 initiates a bus transfer and internal termination is enabled but $\overline{\text{ATA}}$ is asserted before the specified number of wait states have been inserted, the transfer terminates on the CLK cycle where the internal asynchronous transfer acknowledge is asserted. If an alternate master initiates a bus transfer and internal termination is enabled, $\overline{\text{ATA}}$ can be driven by an external device to terminate the transfer before the specified number of wait states has been inserted. In this case, the transfer terminates when the internal asynchronous transfer acknowledge is asserted.

8.3.1.3.3 Bursting Control. The general-purpose chip-selects support burst and non-bursting peripherals and memory. If an external chip select device cannot be accessed using burst transfers, you can program the burst-enable bit in the appropriate Chip Select Control Register (CSCR) or in the Default Memory Control Register (DMCR) to a 0. If the burst-enable bit is set to 1, burst transfers are generated anytime the requested operand size is greater than the programmed chip select or default memory port size. If the burst enable bit is set to 0, nonburst transfers are always generated for the particular chip select

Table 8-4. Memory Map of Chip Select Registers (Continued)

ADDRESS	NAME	WIDTH	DESCRIPTION	RESET VALUE	ACCESS
MBAR + $6E	CSCR0	16	Chip Select Control Register - Bank 0	3C1F, 3C5F, 3C9F, 3CDF, 3D1F, 3D5F, 3D9F, or 3DDF AA set by $\overline{IRQ7}$ at reset PS1 set by $\overline{IRQ4}$ at reset PS0 set by $\overline{IRQ1}$ at reset	R/W
MBAR + $70	CSAR1	16	Chip Select Address Register - Bank 1	uninitialized	R/W
MBAR + $74	CSMR1	32	Chip Select Mask Register - Bank 1	uninitialized	R/W
MBAR + $7A	CSCR1	16	Chip Select Control Register - Bank 1	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W
MBAR + $7C	CSAR2	16	Chip Select Address Register - Bank 2	uninitialized	R/W
MBAR + $80	CSMR2	32	Chip Select Mask Register - Bank 2	uninitialized	R/W
MBAR + $86	CSCR2	16	Chip Select Control Register - Bank 2	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W
MBAR + $88	CSAR3	16	Chip Select Address Register - Bank 3	uninitialized	R/W
MBAR + $8C	CSMR3	32	Chip Select Mask Register - Bank 3	uninitialized	R/W
MBAR + $92	CSCR3	16	Chip Select Control Register - Bank 3	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W
MBAR + $94	CSAR4	16	Chip Select Address Register - Bank 4	uninitialized	R/W
MBAR +$ 98	CSMR4	32	Chip Select Mask Register - Bank 4	uninitialized	R/W
MBAR +$ 9E	CSCR4	16	Chip Select Control Register - Bank 4	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W
MBAR + $A0	CSAR5	16	Chip Select Address Register - Bank 5	uninitialized	R/W
MBAR + $A4	CSMR5	32	Chip Select Mask Register - Bank 5	uninitialized	R/W
MBAR + $AA	CSCR5	16	Chip Select Control Register - Bank 5	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W
MBAR + $AC	CSAR6	16	Chip Select Address Register - Bank 6	uninitialized	R/W
MBAR + $B0	CSMR6	32	Chip Select Mask Register - Bank 6	uninitialized	R/W
MBAR + $B6	CSCR6	16	Chip Select Control Register - Bank 6	uninitialized (except BRST=ASET=WRAH=RDAH=WR=RD=0)	R/W

ASET - Address Setup Enable

This field controls the assertion of chip select with respect to assertion of a valid address.

0 = Assert chip select on the rising edge of CLK that address is asserted. See Figure 8-11.

1 = Delay assertion of chip select for one CLK cycle after address is asserted. See Figure 8-12.

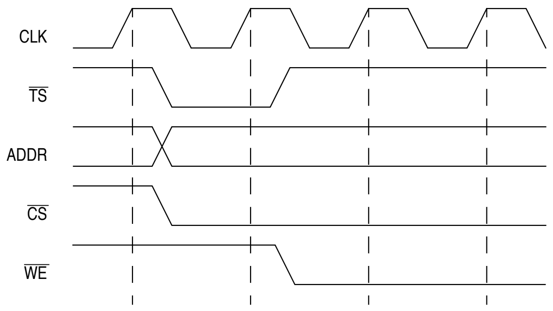


Figure 8-11. Chip Select and Write-Enable Assertion with ASET = 0 Timing

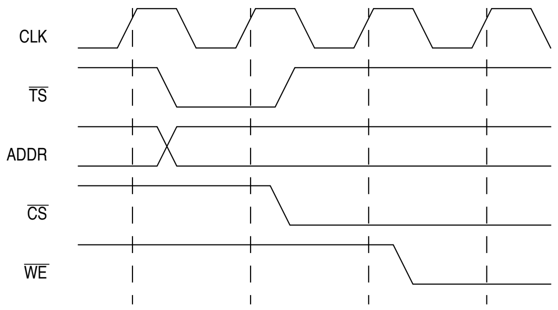


Figure 8-12. Chip Select and Write-Enable Assertion with ASET = 1Timing

NOTE

$\overline{\text{WE}}$ asserts one clock after the assertion of $\overline{\text{CS}}$. During write transfers, if ASET = 1, both $\overline{\text{CS}}$ and $\overline{\text{WE}}$ are delayed by one clock.

Clock H4

The MCF5206 registers the read data driven by the DRAM and negates the internal transfer acknowledge, $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$[0], ending the first byte-read transfer. This begins the $\overline{\text{RAS}}$ precharge. Once $\overline{\text{CAS}}$[0] is negated the DRAM disables its output drivers, and the data bus is three-stated.

Clock H6

Clock H6 is the earliest the next transfer initiated by the ColdFire core can start. The second DRAM byte-read transfer starts in H6. During H6, the MCF5206 drives the row address on the A[27:9], drives $\overline{\text{DRAMW}}$ high indicating a DRAM-read transfer, drives SIZ[1:0] to $1 indicating a byte transfer, and asserts $\overline{\text{TS}}$.

Clock L6

Clock L6 is the same as Clock L1.

Clock H7

Clock H7 is the same as Clock H2.

Clock L7

Clock L7 is the same as Clock L2.

Clock H8

Clock H8 is the same as Clock H3.

Clock H9

Clock H9 is the same as Clock H4.

10.3.3.2 BURST TRANSFER IN NORMAL MODE. A burst transfer to DRAM is generated when the operand size is larger than the DRAM bank port size (e.g., line transfer to a 32-bit port, longword transfer to an 8-bit port). On all DRAM transfers, the MCF5206 asserts $\overline{\text{TS}}$ only once. The start of the secondary transfers of a burst is delayed by the DRAMC until the programmed $\overline{\text{RAS}}$ precharge time is reached.

The timing of burst reads and burst writes is identical in normal page mode, with the exception of when the DRAM drives data on reads and when the MCF5206 drives data on writes.

The fastest possible burst transfer in normal mode requires 3 clocks for the first transfer of the burst and 4 clocks for the secondary transfers (including a 1.5 clock $\overline{\text{RAS}}$ precharge time). You can program the DCTR to generate slower normal mode transfers.

Clock L7

The $\overline{\text{RAS}}$ precharge time has been met, so the MCF5206 asserts $\overline{\text{RAS}}$ is to indicate the row address is valid on A[27:9].

Clock H8

Clock H8 is the same as Clock H2.

Clock L8

Clock L8 is the same as Clock L2.

Clock H9

Clock H9 is the same as Clock H3.

Clock H10

Clock H10 is the same as Clock H4.

10.3.4.5 BUS ARBITRATION. If the MCF5206 loses bus mastership while a page is open ($\overline{\text{RAS}}$ is asserted), $\overline{\text{RAS}}$ is precharged. The $\overline{\text{RAS}}$ precharge timing depends on whether an active fast page mode DRAM transfer is in progress, whether a nonDRAM transfer is in progress, or whether the external bus is idle.

If the BL bit in the SIMR is cleared and $\overline{\text{BG}}$ is negated while an active fast page mode DRAM transfer is in progress, $\overline{\text{BD}}$ remains asserted until the transfer is complete. Once the DRAM transfer completes, the MCF5206 negates $\overline{\text{BD}}$ and begins precharging $\overline{\text{RAS}}$.

In the case where the BL bit in the SIMR is cleared and $\overline{\text{BG}}$ is negated while a nonDRAM transfer is in progress and a page is open, the MCF5206 begins precharging $\overline{\text{RAS}}$ on the cycle following the negation of $\overline{\text{BG}}$, even though $\overline{\text{BD}}$ remains asserted until the completion of the nonDRAM transfer.

If the BL bit in the SIMR is cleared and $\overline{\text{BG}}$ is negated while the external bus is idle and a page is open, the MCF5206 negates $\overline{\text{BD}}$ and begins precharging $\overline{\text{RAS}}$ on the cycle following the negation of $\overline{\text{BG}}$.

When the BL bit in the SIMR is set to 1 and $\overline{\text{BG}}$ is asserted, the bus is locked with the MCF5206. If $\overline{\text{BG}}$ is negated while the bus is locked and a page is open, $\overline{\text{RAS}}$ and $\overline{\text{BD}}$ remains asserted, because the MCF5206 maintains bus mastership regardless of $\overline{\text{BG}}$ when the bus is locked.

NOTE

fast page mode is not supported for external master DRAM transfers. A DRAM bank programmed for fast page mode, BPS - Bank Page Size

This field defines the bank page size for each DRAM bank for fast page mode and burst page mode.

00 = 512 byte page size
01 = 1 KByte page size
10 = 2 KByte page size
11 = Reserved

PM - Page Mode Select

This field selects the type of DRAM transfers generated for each DRAM bank: normal mode, fast page mode, or burst-page-mode transfers.

00 = Normal Mode
01 = Burst Page Mode
10 = Reserved
11 = Fast Page Mode

WR - Write Enable

This field controls whether the DRAM bank can be accessed during write transfers.

0 = Do not activate DRAM control signals on write transfers
1 = Activate DRAM control signals on write transfers

RD - Read Enable

This field controls whether the DRAM bank can be accessed during read transfers.

0 = Do not activate DRAM control signals on read transfers
1 = Activate DRAM control signals on read transfers

10.5 DRAM INITIALIZATION EXAMPLE

The following sample of assembly program illustrates a DRAM initialization procedure. DRAM bank 0 is configured for a 4 MByte DRAM starting at address 0x00100000. The DRAM port size is programmed to 32-bit (1 MByte x 32), the page size to 512 byte, and fast page mode is enabled.

The Module Base Address Register (MBAR) is first written with the MODULE_BASE value. This locates all the MCF5206 internal modules at address 0x00004000. Then the DRAM Controller Timing Register (DCTR) is initialized to give the fastest possible DRAM transfer waveform timing. The DRAM Controller Refresh Register (DCRR) is then written causing DRAM refresh cycles to be generated once every 512 clocks (15.4 μsec for a 33 MHz system clock). Once the DCRR is written, a refresh cycle is immediately generated and refresh cycles start being generated at the newly programmed rate. Next, DRAM Controller Address Register 0 (DCAR0) is written, making the starting address of DRAM bank 0 0x00100000. DRAM Controller Mask Register 0 (DCMR0) is then written such that transfer address bits 18 - 16 are masked, making the DRAM bank 0 address space 1

11.3.5 Multidrop Mode

You can program the UART to operate in a wakeup mode for multidrop or multiprocessor applications. Functional timing information for the multidrop mode is shown in Figure 11-8. You select the mode by setting bits 3 and 4 in UART mode register 1 (UMR1). This mode of operation connects the master station to several slave stations (maximum of 256). In this mode, the master transmits an address character followed by a block of data characters targeted for one of the slave stations. The slave stations channel receivers are disabled; however, they continuously monitor the data stream sent out by the master station. When the master sends an address character, the slave receiver channel notifies its respective CPU by setting the RxRDY bit in the USR and generating an interrupt (if programmed to do so). Each slave station CPU then compares the received address to its station address and enables its receiver if it wants to receive the subsequent data characters or block of data from the master station. Slave stations not addressed continue to monitor the data stream for the next address character. Data fields in the data stream are separated by an address character. After a slave receives a block of data, the slave station CPU disables the receiver and reinitiates the process.

A transmitted character from the master station consists of a start bit, a programmed number of data bits, an address/data (A/D) bit flag, and a programmed number of stop bits. The A/D bit identifies the type of character being transmitted to the slave station. The character is interpreted as an address character if the A/D bit is set or as a data character if the A/D bit is cleared. You select the polarity of the A/D bit by programming bit 2 of UMR1. You should also program UMR1 before enabling the transmitter and loading the corresponding data bits into the transmit buffer.

In multidrop mode, the receiver continuously monitors the received data stream, regardless of whether it is enabled or disabled. If the receiver is disabled, it sets the RxRDY bit and loads the character into the receiver holding register FIFO stack, provided the received A/D bit is a one (address tag). The character is discarded if the received A/D bit is a zero (data tag). If the receiver is enabled, all received characters are transferred to the CPU via the receiver holding register stack during read operations.

In either case, the data bits are loaded into the data portion of the stack while the A/D bit is loaded into the status portion of the stack normally used for a parity error (USR bit 5). Framing error, overrun error, and break detection operate normally. The A/D bit takes the place of the parity bit; therefore, parity is neither calculated nor checked. Messages in this mode can still contain error detection and correction information. One way to provide error detection, if 8-bit characters are not required, is to use software to calculate parity and append it to the 5-, 6-, or 7-bit character.

12.5.3 M-Bus Control Register (MBCR)

M-Bus Control Register (MBCR) Address MBAR+$1E8

	7	6	5	4	3	2	1	0
	MEN	MIEN	MSTA	MTX	TXAK	RSTA	-	
RESET	0	0	0	0	0	0	0	0

Read/Write Supervisor or User Mode

MEN — M-Bus Enable

This bit controls the software reset of the entire M-Bus module.

1 = The M-Bus module is enabled. This bit must be set before any other MBCR bits have any effect.
0 = The module is reset and disabled. This is the power-on reset situation. When low, the interface is held in reset but registers can still be accessed.

If the M-Bus module is enabled in the middle of a byte transfer, the interface behaves as follows: the slave mode ignores the current transfer on the bus and starts operating whenever a subsequent start condition is detected. Master mode is not aware that the bus is busy; therefore, if a start cycle is initiated, the current bus cycle can become corrupt. This would ultimately result in either the current bus master or the M-Bus module losing arbitration, after which bus operation would return to normal.

MIEN — M-Bus Interrupt Enable

1 = Interrupts from the M-Bus module are enabled. An M-Bus interrupt occurs provided the MIF bit in the status register is also set.
0 = Interrupts from the M-Bus module are disabled. This does not clear any currently pending interrupt condition.

MSTA — Master/Slave Mode Select Bit

At reset, this bit is cleared. When this bit is changed from 0 to 1, a START signal is generated on the bus, and the master mode is selected. When this bit is changed from 1 to 0, a STOP signal is generated and the operation mode changes from master to slave.

MSTA is cleared without generating a STOP signal when the master loses arbitration.

1 = Master Mode
0 = Slave Mode

MTX — Transmit/Receive mode select bit

This bit selects the direction of master and slave transfers. When addressed as a slave this bit should be set by software according to the SRW bit in the status register. In master mode, this bit should be set according to the type of transfer required. Therefore, for address cycles, this bit is always high.

1 = Transmit
0 = Receive

```
ISRLEA.LMBSR,-(A7);          LOAD EFFECTIVE ADDRESS
BCLR.B#1,(A7)+;              CLEAR THE MIF FLAG
MOVE.BMBCR,-(A7);            PUSH ADDRESS ON STACK,
BTST.B#5,(A7)+;              CHECK THE MSTA FLAG
BEQ.SSLAVE;                  BRANCH IF SLAVE MODE
MOVE.BMBCR,-(A7);            PUSH ADDRESS ON STACK
BTST.B#4,(A7)+;              CHECK THE MODE FLAG
BEQ.SRECEIVE;                BRANCH IF IN RECEIVE MODE
MOVE.BMBSR,-(A7);            PUSH ADDRESS ON STACK,
BTST.B#0,(A7)+;              CHECK ACK FROM RECEIVER
BNE.B END;                   IF NO ACK, END OF TRANSMISSION
TRANSMITMOVE.BDATABUF,-(A7); STACK DATA BYTE
MOVE.B(A7)+, MBDR;           TRANSMIT NEXT BYTE OF DATA
```

12.6.4 Generation of STOP

A data transfer ends with a STOP signal generated by the "master" device. A master transmitter can generate a STOP signal after all the data has been transmitted. The following is an example showing how a master transmitter generates a stop condition.

```
MASTXMOVE.BMBSR, -(A7);  IF NO ACK, BRANCH TO END
BTST.B#0,(A7)+
BNE.B END
MOVE.BTXCNT,D0;          GET VALUE FROM THE TRANSMITTING COUNTER
BEQ.SEND;                IF NO MORE DATA, BRANCH TO END
MOVE.BDATABUF,-(A7);     TRANSMIT NEXT BYTE OF DATA
MOVE.B(A7)+,MBDR
MOVE.BTXCNT,D0;          DECREASE THE TXCNT
SUBQ.L#1,D0
MOVE.BD0,TXCNT
BRA.SEMASTX;             EXIT
ENDLEA.LMBCR,-(A7);      GENERATE A STOP CONDITION
BCLR.B#5,(A7)+
EMASTXRTE;               RETURN FROM INTERRUPT
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data, which can be done by setting the transmit acknowledge bit (TXAK) before reading the 2nd last byte of data. Before reading the last byte of data, a STOP signal must be generated first. The following is an example showing how a master receiver generates a STOP signal.

```
MASRMOVE.BRXCNT,D0;    DECREASE RXCNT
SUBQ.L#1,D0
MOVE.BD0,RXCNT
BEQ.SENMASR;           LAST BYTE TO BE READ
MOVE.BRXCNT,D1;        CHECK SECOND LAST BYTE TO BE READ (NOT LAST ONE OR SECOND
                       LAST
EXTB>LD1
SUBI.L#1,D1;
BNE.SNXMAR
LAMARBSET.B#3,MBCR;    SECOND LAST, DISABLE ACK TRANSMITTING
BRANXMAR
```

```
ENMASRBCLR.B#5,MBCR;   LAST ONE, GENERATE 'STOP' SIGNAL
NXMARMOVE.BMBDR,RXBUF; READ DATA AND STORE RTE
```

12.6.5 Generation of Repeated START

At the end of data transfer, if the master still wants to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. A program example is as shown.

```
RESTARTMOVE.BMBCR,-(A7); ANOTHER START (RESTART)
BSET.B#2, (A7)
MOVE.B(A7)+, MBCR
MOVE.BCALLING,-(A7);      TRANSMIT THE CALLING ADDRESS, D0=R/W-
MOVE.BCALLING,-(A7);
MOVE.B(A7)+, MBDR
```

12.6.6 Slave Mode

In the slave interrupt service routine, the module addressed as slave bit (MAAS) should be tested to check if a calling of its own address has just been received. If MAAS is set, software should set the transmit/receive mode select bit (MTX bit of MBCR) according to the R/$\overline{W}$ command bit (SRW). Writing to the MBCR clears the MAAS automatically. The only time MAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred; interrupts resulting from subsequent data transfers have MAAS cleared. A data transfer can now be initiated by writing information to MBDR, for slave transmits, or dummy reading from MBDR, in slave-receive mode. The slave drives SCL low in between byte transfers. SCL is released when the MBDR is accessed in the required mode.

In slave transmitter routine, the received acknowledge bit (RXAK) must be tested before transmitting the next byte of data. Setting RXAK means an "end-of-data" signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

12.6.7 Arbitration Lost

If several masters try to simultaneously engage the bus, only one master wins and the others lose arbitration. The devices that lost arbitration are immediately switched to slave receive mode by the hardware. Their data output to the SDA line is stopped, but SCL is still generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with MAL=1 and MSTA=0. If one master tries to transmit or do a START while the bus is being engaged by another master, the hardware : (1) inhibits the transmission, (2) switches the MSTA bit from 1 to 0 without generating STOP condition, (3) generates an interrupt to CPU and, (4) sets the MAL to indicate the failed attempt to engage the bus. When considering these cases, the slave service routine should test the MAL first and the software should clear the MAL bit if it is set.

Additionally, a WDDATA opcode is supported that lets the processor core write any operand (byte, word, longword) directly to the DDATA port, independent of any Debug module configuration. This opcode also generates the special PST = $4 encoding when executed.

14.2 BACKGROUND-DEBUG MODE (BDM)

ColdFire 52xx processors support a modified version of the Background-Debug mode (BDM) functionality found on Motorola's CPU32 Family of parts. BDM implements a low-level system debugger in the microprocessor hardware. Communication with the development system is handled via a dedicated, high-speed serial command interface (BDM port).

Unless noted otherwise, the BDM functionality provided by ColdFire 52xx processors is a proper subset of the CPU32 functionality. The main differences include the following:

- ColdFire implements the BDM controller in a dedicated hardware module. Although some BDM operations do require the CPU to be halted (e.g., CPU register accesses), other BDM commands such as memory accesses can be executed while the processor is running.
- DSCLK, DSI, and DSO are treated as synchronous signals, where the inputs (DSCLK and DSI) must meet the required input setup and hold timings, and the output (DSO) is specified as a delay relative to the rising edge of the processor clock.
- On CPU32 parts, DSO could signal hardware that a serial transfer can start. ColdFire clocking schemes restrict the use of this bit. Because DSO changes only when DSCLK is high, DSO cannot be used to indicate the start of a serial transfer. The development system should use either a free-running DSCLK or count the number of clocks in any given transfer.
- The Read/Write System Register commands (RSREG/WSREG) have been replaced by Read/Write Control Register commands (RCREG/WCREG). These commands use the register coding scheme from the MOVEC instruction.
- Read/Write Debug Module Register commands (RDMREG/WDMREG) have been added to support Debug module register accesses.
- CALL and RST commands are not supported.
- Illegal command responses can be returned using the FILL and DUMP commands.
- For any command performing a byte-sized memory read operation, the upper 8 bits of the response data are undefined. The referenced data is returned in the lower 8 bits of the response.
- The debug module forces alignment for memory-referencing operations: long accesses are forced to a 0-modulo-4 address; word accesses are forced to a 0-modulo-2 address. An address error response can no longer be returned.

14.2.1 CPU Halt

Although some BDM operations can occur in parallel with CPU operation, unrestricted BDM operation requires the CPU to be halted. A number of sources can cause the CPU to halt, including the following (as shown in order of priority):

1. The occurrence of the catastrophic fault-on-fault condition automatically halts the processor. The halt status is posted on the PST port ($F).
2. The occurrence of a hardware breakpoint (reference subsection **Section 14.3 Real-Time Debug Support**) can be configured to generate a pending halt condition in a manner similar to the assertion of the $\overline{\text{BKPT}}$ signal. In some cases, the occurrence of this type of breakpoint halts the processor in an imprecise manner. Once the hardware breakpoint is asserted, the processor halts at the next sample point. See **Section 14.3.2 Theory of Operation** for more detail.
3. The execution of the HALT (also known as BGND on the 683xx devices) instruction immediately suspends execution and posts the halt status ($F) on the PST outputs. By default, this is a supervisor instruction and attempted execution while in user mode generates a privilege-violation exception. A User Halt Enable (UHE) control bit is provided in the Configuration/Status Register (CSR) to allow execution of HALT in user mode.
4. The assertion of the $\overline{\text{BKPT}}$ input pin is treated as an pseudo-interrupt, i.e., the halt condition is made pending until the processor core samples for halts/interrupts. The processor samples for these conditions once during the execution of each instruction. If there is a pending halt condition at the sample time, the processor suspends execution and enters the halted state. The halt status ($F) is reflected in the PST outputs.

The halt source is indicated in CSR[27:24]; for simultaneous halt conditions, the highest priority source is indicated.

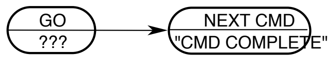
There are two special cases to be considered that involve the assertion of the $\overline{\text{BKPT}}$ pin.

After $\overline{\text{RSTI}}$ is negated, the processor waits for 16 clock cycles before beginning reset exception processing. If the $\overline{\text{BKPT}}$ input pin is asserted within the first eight cycles after $\overline{\text{RSTI}}$ is negated, the processor enters the halt state, signaling that status on the PST outputs ($F). While in this state, all resources accessible via the Debug module can be referenced. Once the system initialization is complete, the processor response to a BDM GO command depends on the set of BDM commands performed while “breakpointed.” Specifically, if the processor’s PC register was loaded, the GO command causes the processor to exit the halt state and pass control to the instruction address contained in the PC. In this case, the normal reset exception processing is bypassed. Conversely, if the PC register was not loaded, the GO BDM command causes the processor to exit the halt state and continue with reset exception processing.

ColdFire 52xx processors also handle a special case with the assertion of $\overline{\text{BKPT}}$ while the processor is stopped by execution of the STOP instruction. For this case when the $\overline{\text{BKPT}}$ is asserted, the processor exits the stopped mode and enters the halted state. Once halted, the standard BDM commands may be exercised. When the processor is restarted, it continues with the execution of the next sequential instruction, i.e., the instruction following the STOP opcode.

The debug module Configuration/Status Register (CSR) maintains status defining the condition that caused the CPU to halt.

Command Sequence:



Operand Data:

None

Result Data:

The “command complete” response ($0FFFF) is returned during the next shift operation.

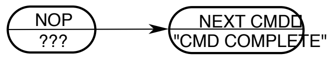
14.2.3.4.8 No Operation (NOP). NOP performs no operation and can be used as a null command, where required.

Formats:

15 — 12	11 — 8	7 — 4	3 — 0
$0	$0	$0	$0

NOP Command

Command Sequence:



Operand Data:

None

Result Data:

The “command complete” response ($0FFFF) is returned during the next shift operation.

14.2.3.4.9 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 in size, regardless of the implemented register width. The second and third words of the command effectively form a 32-bit address 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.

WCREG command, TT[1:0] = 11 and TM[2:0] = 000. For breakpoint-acknowledge transfers, the TM signals are low.

Table 14-10. Transfer Modifier Encodings for Normal Transfers

TM[2:0]	TRANSFER MODIFIER
000	Reserved
001	User Data Access
010	User Code Access
011 - 100	Reserved
101	Supervisor Data Access
110	Supervisor Code Access
111	Reserved

Table 14-11. Transfer Modifier Encodings for Alternate Access Transfers

TM[2:0]	TRANSFER MODIFIER
000 - 100, 111	Reserved
101	Emulator Mode Data Access
110	Emulator Mode Code Access

14.3.1.3 PROGRAM COUNTER BREAKPOINT REGISTER (PBR, PBMR). The PC Breakpoint Registers define a region in the instruction address space of the processor that can be used as part of the trigger. The PBR value is masked by the PBMR value, allowing only those bits in PBR that have a corresponding zero in PBMR to be compared with the processor's program counter register as defined in the TDR.

14.3.1.4 DATA BREAKPOINT REGISTER (DBR, DBMR). The Data Breakpoint Registers define a specific data pattern that can be used as part of the trigger into Debug mode.The DBR value is masked by the DBMR value, allowing only those bits in DBR that have a corresponding zero in DBMR to be compared with the ColdFire CPU core data signals, as defined in the TDR.

The data breakpoint register supports both aligned and misaligned operand references. The relationship between the processor core address, the access size, and the corresponding location within the 32-bit core data bus is shown in Table 14-12.

Table 14-12. Core Address, Access Size, and Operand Location

CORE ADDRESS[1:0]	ACCESS SIZE	OPERAND LOCATION
00	Byte	Data[31:24]
01	Byte	Data[23:16]

instructions. The various states of the TAP controller are shown in Figure 15-2 For more detail on each state, refer to the IEEE 1149.1 Standard JTAG document. Note that from any state the TAP controller is in, Test-Logic-Reset can be entered if TMS is held high for at least 5 rising edges of TCK.

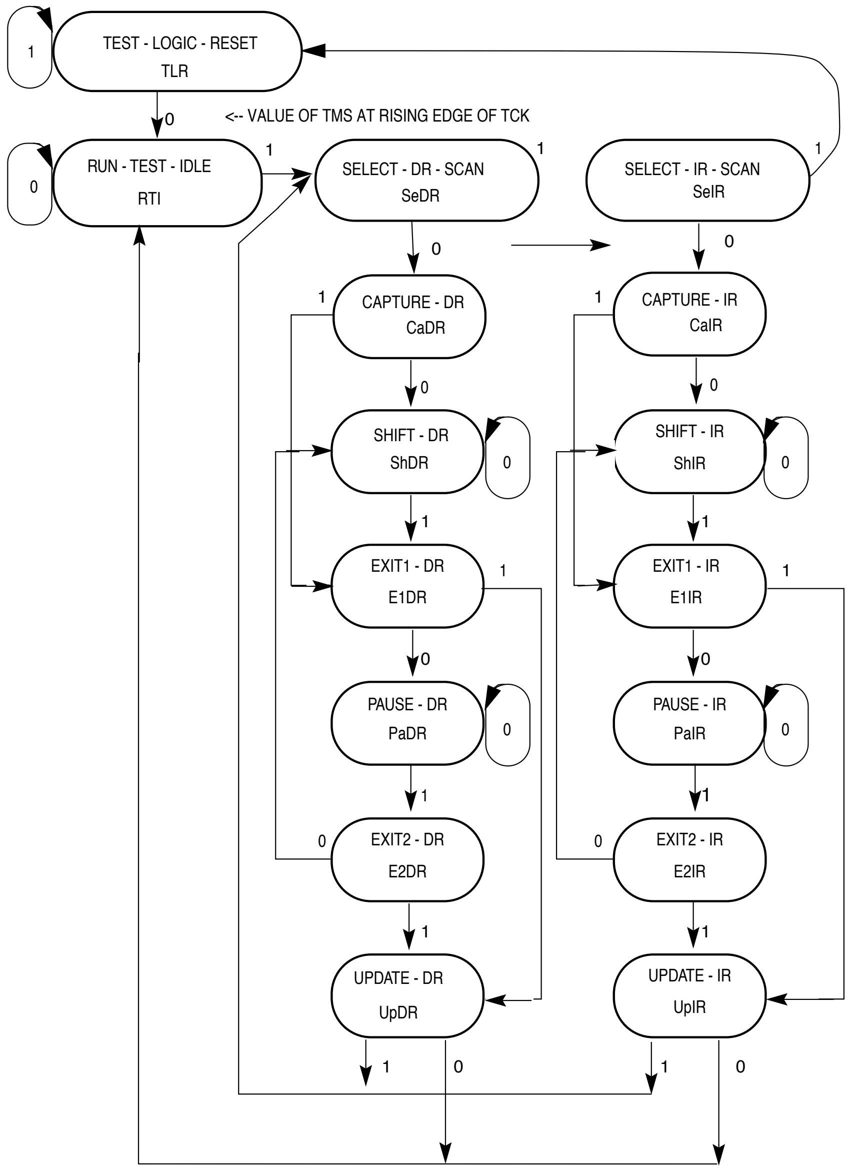


Figure 15-2. JTAG TAP Controller State Machine