Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

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

Applications of Embedded - Microprocessors

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

Details	
Product Status	Active
Core Processor	Coldfire V2
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	25MHz
Co-Processors/DSP	-
RAM Controllers	DRAM
Graphics Acceleration	-
Display & Interface Controllers	-
Ethernet	-
SATA	-
USB	-
Voltage - I/O	5V
Operating Temperature	-40°C ~ 85°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=mcf5206cab25a

Table 3-2. Format Field Encodings

ORIGINAL A7 @ TIME OF EXCEPTION, BITS 1:0	A7 @ 1ST INSTRUCTION OF HANDLER	FORMAT FIELD
00	Original A7 - 8	4
01	Original A7 - 9	5
10	Original A7 - 10	6
11	Original A7 - 11	7

- A 4-bit fault status field, FS[3:0], at the top of the system stack. This field is defined for access and address errors only and written as zeros for all other types of exceptions. See Table 3-3.

Table 3-3. Fault Status Encodings

FS[3:0]	DEFINITION
00xx	Reserved
0100	Error on instruction fetch
0101	Reserved
011x	Reserved
1000	Error on operand write
1001	Attempted write to write-protected space
101x	Reserved
1100	Error on operand read
1101	Reserved
111x	Reserved

- The 8-bit vector number, vector[7:0], defines the exception type and is calculated by the processor for all internal faults and represents the value supplied by the peripheral in the case of an interrupt. Refer to Table 3-1.

3.5 PROCESSOR EXCEPTIONS

3.5.1 Access Error Exception

An Access Error Exception vector number $2 occurs when a bus cycle terminates with an error condition. The exact processor response to an access error depends on the type of memory reference being performed.

For an instruction fetch, the processor postpones the error reporting until the faulted reference is needed by an instruction for execution. Therefore, faults that occur during instruction prefetches that are then followed by a change of instruction flow does not generate an exception. When the processor attempts to execute an instruction with a faulted opword and/or extension words, the access error is signaled and the instruction aborted. For this type of exception, the programming model has not been altered by the instruction generating the access error.

3.6.1 Timing Assumptions

For the timing data presented in this section, the following assumptions apply:

1. The operand execution pipeline (OEP) is loaded with the opword and all required extension words at the beginning of each instruction execution. This implies that the OEP does not wait for the instruction fetch pipeline (IFP) to supply opwords and/or extension words.
2. The OEP does not experience any sequence-related pipeline stalls. For ColdFire 5200 processors, the most common example of this type of stall involves consecutive store operations, excluding the MOVEM instruction. For all STORE operations (except MOVEM), certain hardware resources within the processor are marked as “busy” for two clock cycles after the final DSOC cycle of the store instruction. If a subsequent STORE instruction is encountered within this 2-cycle window, it is stalled until the resource again becomes available. Thus, the maximum pipeline stall involving consecutive STORE operations is 2 cycles. The MOVEM instruction uses a different set of resources and this stall does not apply.
3. The OEP completes all memory accesses without any stall conditions caused by the memory itself. Thus, the timing details provided in this section assume that an infinite zero-wait state memory is attached to the processor core.
4. All operand data accesses are aligned on the same byte boundary as the operand size, i.e., 16-bit operands aligned on 0-modulo-2 addresses, 32-bit operands aligned on 0-modulo-4 addresses.

If the operand alignment fails these guidelines, it is misaligned. The processor core decomposes the misaligned operand reference into a series of aligned accesses as shown in Table 3-4.

Table 3-4. Misaligned Operand References

ADDRESS[1:0]	SIZE	KBUS OPERATIONS	ADDITIONAL C(R/W)
X1	Word	Byte, Byte	2(1/0) if read 1(0/1) if write
X1	Long	Byte, Word, Byte	3(2/0) if read 2(0/2) if write
10	Long	Word, Word	2(1/0) if read 1(0/1) if write

3.6.2 MOVE Instruction Execution Times

The execution times for the MOVE.{B,W} instructions are shown in Table 3-5, while Table 3-6 provides the timing for MOVE.L.

For all tables in this section, the execution time of any instruction using the PC-relative effective addressing modes is the same for the comparable An-relative mode.

The nomenclature “xxx.wl” refers to both forms of absolute addressing, xxx.w and xxx.l.

contents of the fill buffer versus its corresponding cache location. At the time of the miss, the hardware indicator is set, marking the fill buffer as “most recently used.” If a subsequent access occurs to the cache location defined by bits [8:4] of the fill buffer address, the data in the cache memory array is now most recently used, so the hardware indicator is cleared. In all cases, the indicator defines whether the contents of the line fill buffer or the memory data array are most recently used. At the time of the next cache miss, the contents of the line-fill buffer are written into the memory array if the entire line is present, and the fill buffer data is still most recently used compared to the memory array.

The fill buffer can also be used as temporary storage for line-sized bursts of non-cacheable references under control of CACR[10]. With this bit set, a noncacheable instruction fetch is processed as defined by Table 4-2. For this condition, the fill buffer is loaded and subsequent references can hit in the buffer, but the data is never loaded into the memory array.

Table 4-2 shows the relationship between CACR bits 31 and 10 and the type of instruction fetch.

Table 4-2. Instruction Cache Operation as Defined by CACR[31, 10]

CACR[31]	CACR[10]	TYPE OF INSTR. FETCH	DESCRIPTION
0	0	N/A	Instruction cache is completely disabled; all fetches are word, longword in size.
0	1	N/A	All fetches are word, longword in size
1	X	Cacheable	Fetch size is defined by Table 5-1 and contents of the line-fill buffer can be written into the memory array
1	0	Noncacheable	All fetches are longword in size, and not loaded into the line-fill buffer
1	1	Noncacheable	Fetch size is defined by Table 5-1 and loaded into the line-fill buffer, but are never written into the memory array.

4.4 INSTRUCTION CACHE PROGRAMMING MODEL

Three supervisor registers define the operation of the instruction cache and local bus controller: the Cache Control Register (CACR) and two Access Control Registers (ACR0, ACR1).

4.4.1 Instruction Cache Registers Memory Map

Table 4-3 below shows the memory map of the Instruction cache and access control registers.

The following lists several keynotes regarding the programming model table:

- The Cache Control Register and Access Control Registers can only be accessed in supervisor mode using the MOVEC instruction with an Rc value of $002, $004 and $005, respectively.

The ACRs are 32-bit write-only supervisor control register. They are accessed in the CPU address space via the MOVEC instruction with an Rc encoding of $004 and $005. The ACRs can be read when in background debug mode (BDM). At system reset, the entire registers are cleared.

ACR Programming Model

Access Control Registers (ACR0, ACR1)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AB31	AB30	AB29	AB28	AB27	AB26	AB25	AB24	AM31	AM30	AM29	AM28	AM27	AM26	AM25	AM24
RESET: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	SM1	SM0	-	-	-	-	-	-	CM	BUFW	-	-	WP	-	-
RESET: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

AB[31:24] - Address Base [31:24]

This 8-bit field is compared to address bits [31:24] from the processor's local bus under control of the ACR address mask. If the address matches, the attributes for the memory reference are sourced from the given ACR.

AM[31:24] - Address Mask [31:24]

This 8-bit field can mask any bit of the AB field comparison. If a bit in the AM field is set, then the corresponding bit of the address field comparison is ignored.

EN - Enable: ACR [15]

The EN bit defines the ACR enable. Hardware reset clears this bit, disabling the ACR.

0 = ACR disabled
1 = ACR enabled

SM[1:0] - Supervisor mode: ACR [14:13]

This two-bit field allows the given ACR to be applied to references based on operating privilege mode of the ColdFire processor. The field uses the ACR for user-references only, supervisor-references only, or all accesses.

00 = Match if user mode
01 = Match if supervisor mode
1x = Match always - ignore user/supervisor mode

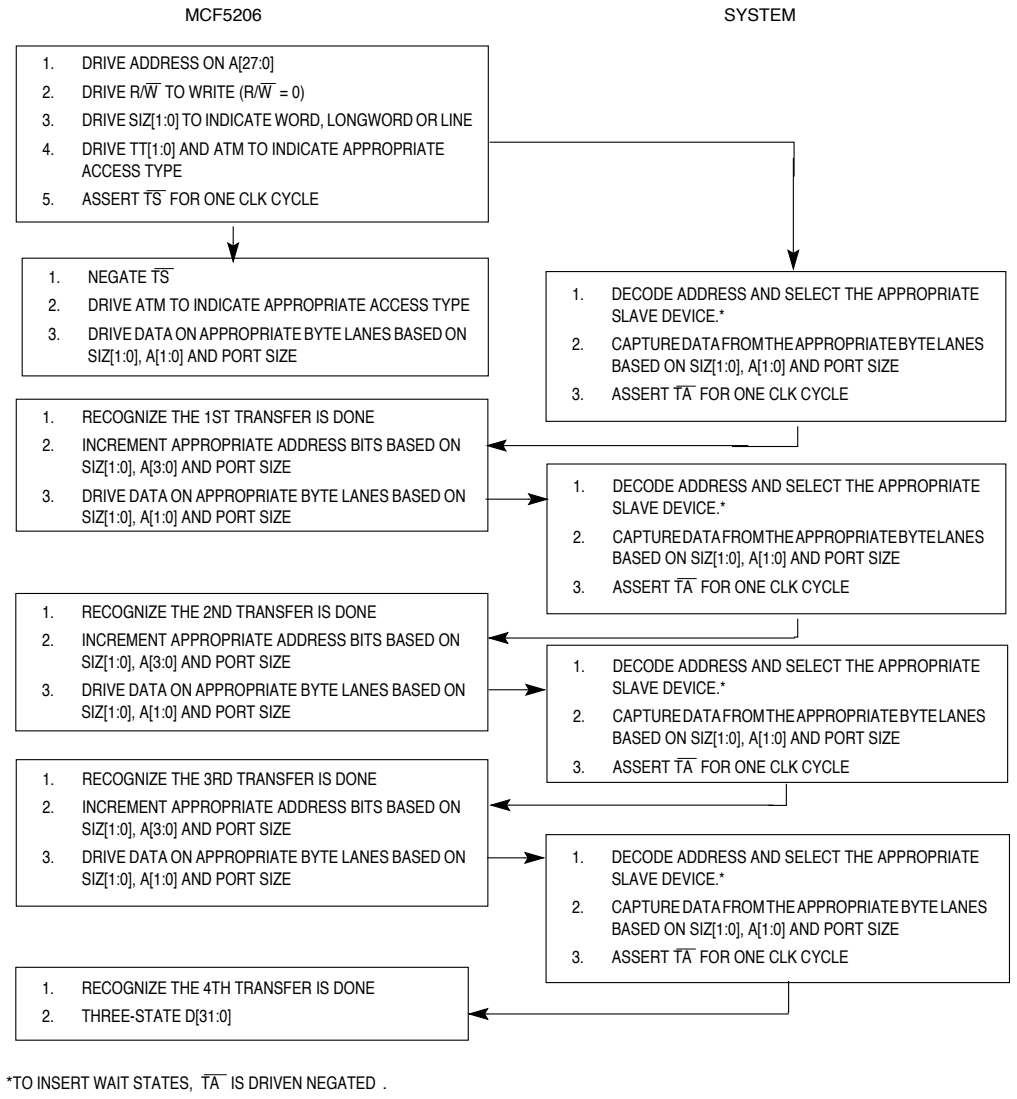


Figure 6-10. Word-, Longword-, and Line-Write Transfer Flowchart

wire mode. In this mode, the active-low bus grant ($\overline{BG}$) input of the MCF5206 is connected to the active-high HOLDREQ output of the external bus master and the active-low bus-driven ($\overline{BD}$) output of the MCF5206 is connected to the active-high HOLDACK input of the external bus master. Because the external bus master controls the assertion/negation of HOLDREQ, it controls when the MCF5206 is granted the bus, making the MCF5206 the lower priority master. You can program the bus lock (BL) bit in the SIM Configuration Register (SIMR) to a 1, instructing the MCF5206 to retain control of the external bus, even when bus grant ($\overline{BG}$) is negated. This lets you control the priority of the MCF5206 with respect to the external master when in two-wire mode.

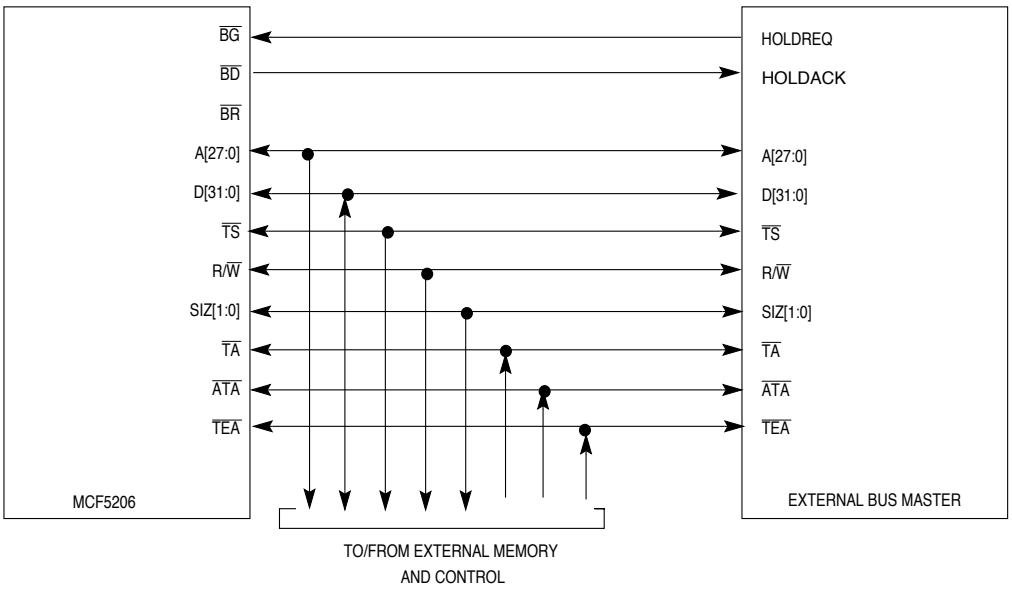


Figure 6-33. MCF5206 Two-Wire Mode Bus Arbitration Interface

When the external master is not using the bus, it negates HOLDREQ driving bus grant ($\overline{BG}$) low, granting the bus to the MCF5206. When the MCF5206 has an internal bus request pending and bus grant ($\overline{BG}$) is low, the MCF5206 drives $\overline{BD}$ low, negating HOLDACK to the external bus master. When the external bus master requires use of the external bus, it asserts HOLDREQ, driving bus grant ($\overline{BG}$) high, requesting the MCF5206 to relinquish the bus. If $\overline{BG}$ is negated while a bus cycle is in progress and if the bus lock bit is cleared, the MCF5206 relinquishes the bus at the completion of the bus cycle. Note that the MCF5206 considers the individual transfers of a burst or burst-inhibited access to be a single bus cycle and does not relinquish the bus until the completion of the last transfer of the series.

When the bus has been granted to the MCF5206, one of two situations can occur. In the first case, the MCF5206 has an internal bus request pending, the MCF5206 asserts $\overline{BD}$ to indicate explicit bus ownership and begins the pending bus cycle by asserting $\overline{TS}$. The

PAR6 - Pin Assignment Bit 6

This bit lets you select how the external interrupt inputs are used by the MCF5206 as follows:

0 = Set $\overline{\text{IRQ7}}$/$\overline{\text{IPL2}}$, $\overline{\text{IRQ4}}$/$\overline{\text{IPL1}}$ and $\overline{\text{IRQ1}}$/$\overline{\text{IPL0}}$ to be used as individual interrupt inputs at level 7, 4 and 1, respectively.
1 = Set $\overline{\text{IRQ7}}$/$\overline{\text{IPL2}}$, $\overline{\text{IRQ4}}$/$\overline{\text{IPL1}}$ and $\overline{\text{IRQ1}}$/$\overline{\text{IPL0}}$ to be used as encoded interrupt priority level inputs

PAR5 - Pin Assignment Bit 5

This bit lets you select the signals output on the pins PP[7:4]/PST[3:0] as follows:

0 = Output general purpose I/O signals PP7 - PP4 on PP[7:4]/PST[3:0] pins
1 = Output background debug mode signals PST3 - PST0 on PP[7:4]/PST[3:0] pins

PAR4 - Pin Assignment Bit 4

This bit lets you select the signals output on the pins PP[3:0]/DDATA[3:0] as follows:

0 = Output Parallel Port output signals PP3 - PP0 on PP[3:0]/DDATA[3:0] pins
1 = Output background debug mode signals DDATA3 - DDATA0 on PP[3:0]/DDATA[3:0] pins

PAR3 - PAR0 - Pin Assignment Bits 3 - 0

These bits let you select the signal output on the pins $\overline{\text{CS}}$[7:4]/$\overline{\text{A}}$[27:24]/$\overline{\text{WE}}$[3:0]. You can select the signals as shown in Table 7-9.

Table 7-9. PAR3 - PAR0 Pin Assignment

PAR[3:0]	A27/CS7/WE0	A26/CS6/WE1	A25/CS5/WE2	A24/CS4/WE3
0000	WE0	WE1	WE2	WE3
0001	WE0	WE1	CS5	CS4
0010	WE0	WE1	CS5	A24
0011	WE0	WE1	A25	A24
0100	WE0	CS6	CS5	CS4
0101	WE0	CS6	CS5	A24
0110	WE0	CS6	A25	A24
0111	WE0	A26	A25	A24
1000	CS7	CS6	CS5	CS4
1001	CS7	CS6	CS5	A24
1010	CS7	CS6	A25	A24
1011	CS7	A26	A25	A24
1100	A27	A26	A25	A24
1101	Reserved			
1110	Reserved			
1111	Reserved			

transfer of the first word of the longword is complete. If $\overline{TA}$ is negated, the alternate master continues to sample $\overline{TA}$ and inserts wait states instead of terminating the transfer.

Clock 4 (C4)

At the start of clock 4, the alternate master increments the address to indicate the second word of the longword transfer. The MCF5206 continues to assert $\overline{CS}$ and $\overline{TA}$ and the selected slave outputs the data indicated by the new address on D[31:16]. At the end of clock 4, the alternate master samples the level of $\overline{TA}$. If $\overline{TA}$ is asserted, the transfer of the second word of the longword is complete. If $\overline{TA}$ is negated, the alternate master continues to sample $\overline{TA}$ and inserts wait states instead of terminating the transfer.

Clock 5 (C5)

At the start of clock 5, the MCF5206 negates $\overline{CS}$ and $\overline{TA}$, completing the alternate master transfer. After the next rising edge of CLK, the MCF5206 three states $\overline{TA}$. The alternate master can assert $\overline{TS}$ starting another transfer.

NOTE

Write enables ($\overline{WE}$[3:0]) does not assert on zero wait state alternate master write transfers.

8.3.4.3 ALTERNATE MASTER BURST TRANSFER WITH ADDRESS SETUP AND HOLD. Figure 8-10 illustrates a longword bursting read transfer from a 16-bit port with either address setup or read address hold enabled.

Chip Select Mask Register(CSMR2 - CSMR7)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BAM31	BAM30	BAM29	BAM28	BAM27	BAM26	BAM25	BAM24	BAM23	BAM22	BAM21	BAM20	BAM19	BAM18	BAM17	BAM16
RESET: -	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	SC	SD	UC	UD	-
RESET: 0	0	0	0	0	0	0	0	0	0	0	-	-	-	-	0

BAM31-BAM16 - Base Address Mask

This field defines the chip select block size through the use of address mask bits. Any bit set to 1 masks the corresponding base address register (CSAR) bit (the base address bit becomes a “don’t care” in the decode).

0 = Corresponding address bit is used in chip select address decode.
1 = Corresponding address bit is not used in chip select address decode.

C/I, SC, SD, UC, UD - CPU Space, Supervisor Code, Supervisor Data, User Code, User Data Transfer Mask

These fields allows specific types of transfers to be inhibited from accessing a chip select. If a transfer mask bit is cleared, a transfer of that type can access the corresponding chip select. If a transfer mask bit is set to 1, an transfer of that type can not access the corresponding chip select. The transfer mask bits are:

C/I = CPU space and Interrupt Acknowledge Cycle mask ($\overline{CS}$[1] only)
SC = Supervisor Code mask
SD = Supervisor Data mask
UC = User Code mask
UD = User Data mask

For each transfer mask bit:

0 = Do not mask this type of transfer for the chip select. A transfer of this type can occur for this chip select.
1 = Mask this type of transfer from the chip select. If this type of transfer is generated, this chip select activation is not activated.

NOTE

The C/I, SC, SD, UC, and UD bits are ignored during alternate master transfers. Therefore, an alternate master transfer can activate a chip select regardless of the transfer masks.

RDAH - Read Address Hold Enable

This field controls the address and attribute hold time after the termination ($\overline{TA}$, $\overline{ATA}$, $\overline{TEA}$ or internal transfer acknowledge) during a read cycle that hits in the chip select address space.

0 = Do not hold address and attributes an extra cycle after $\overline{CS}$ negates on reads. See Figure 8-15.
1 = Hold address and attributes one cycle after $\overline{CS}$ negates on reads. See Figure 8-16.

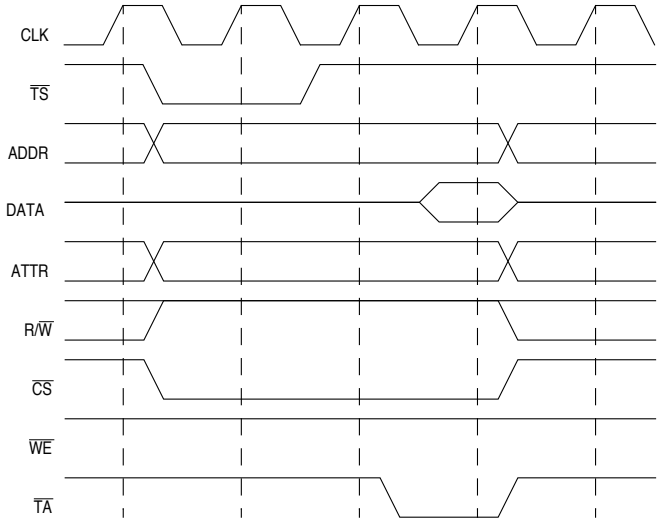


Figure 8-15. Address Hold Timing with RDAH = 0

Figure 10-13 shows the timing of a word read in Fast Page Mode followed by a page miss word read using 8-bit wide EDO DRAM (the EDO bit in the DCTR is set).

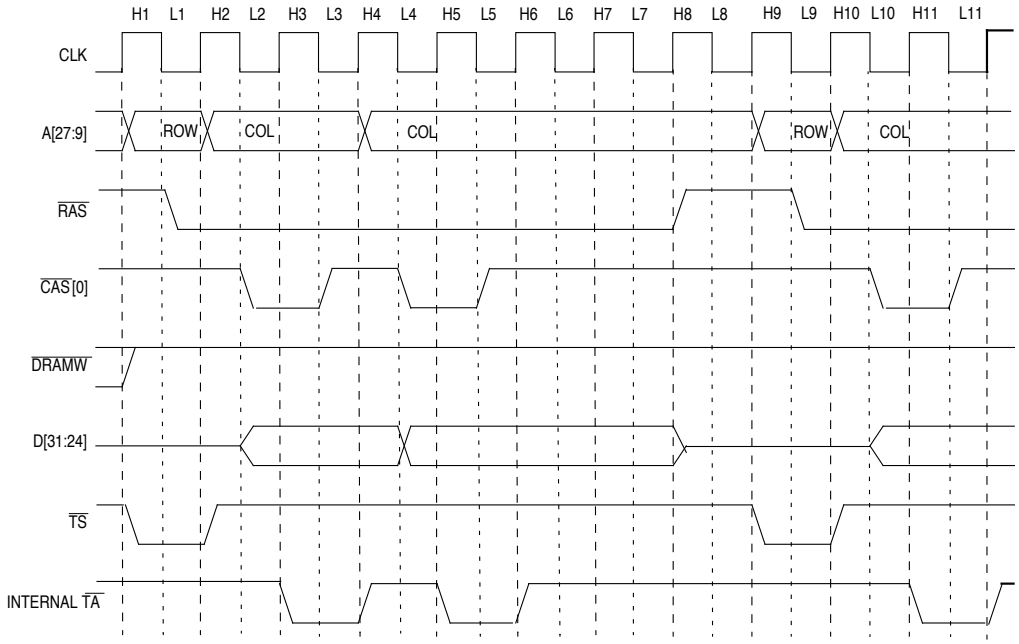


Figure 10-13. Word Read Transfer Followed by a Page Miss Byte Read Transfer in Fast Page Mode with 8-Bit EDO DRAM

Clock H1

The first byte read transfer of the burst word transfer starts in H1. During H1, the MCF5206 drives the row address on A[27:9], drives $\overline{\text{DRAMW}}$ high indicating a DRAM write transfer, drives SIZ[1:0] to $2 indicating a byte transfer, and asserts $\overline{\text{TS}}$.

Clock L1

The MCF5206 asserts $\overline{\text{RAS}}$ to indicate the row address is valid on A[27:9].

Clock H2

The MCF5206 negates $\overline{\text{TS}}$, drives the column address on A[27:9].

Clock L2

The MCF5206 asserts $\overline{\text{CAS}}$[0] to indicate the column address is valid on A[27:9]. At this point the EDO DRAM turns on it's output drivers and begins driving data on D[31:24].

For each transfer mask bit:

0 = Do not mask this type of transfer for the DRAM bank. A transfer of this type can access the DRAM bank
1 = Mask this type of transfer for the DRAM bank. A transfer of this type cannot access the DRAM bank

NOTE

The SC, SD, UC, and UD bits are ignored during external master transfers. Therefore, an external master transfer can access the DRAM banks regardless of the transfer masks.

NOTE

In determining whether an external master transfer address hits in a DRAM bank, the portion of the address bus that is unavailable externally are regarded as “0's.” That is, the external master transfer address always have A[31:28] as 0’s and those bits of A[27:24] that are not programmed to be external address bits as 0’s. In order for a bank to be accessible to an external master, the address bits that are unavailable to the external master must either be set to 0 in the DCAR or be masked in the DCMR.

10.4.2.5 DRAM CONTROLLER CONTROL REGISTER (DCCR0 - DCCR1). Each DCCR specifies the port size, page size, page mode, and activation of each of the DRAM banks. Each DCCR is an 8-bit read/write control register. Master reset and normal reset set all bits to zero.

DRAM Controller Control Register(DCCR) — Address MBAR + $57 (Bank0), Address MBAR + $63 (Bank1)

7	6	5	4	3	2	1	0
PS1	PS0	BPS1	BPS0	PM1	PM0	WR	RD
NORMAL OR MASTER RESET: 0	0	0	0	0	0	0	0

PS - Port Size

This field specifies the data width of the DRAM bank. PS determines the byte lanes that data is driven on during write cycles and the byte lanes that data is sampled from during read cycles.

00 = 32-bit port size - Data sampled and driven on D[31:0]
01 = 8-bit port size - Data sampled and driven on D[31:24] only
10 = 16-bit port size - Data sampled and driven on D[31:16] only
11 = 16-bit port size - Data sampled and driven on D[31:16] only

NOTE

For further information on M-Bus system configuration, protocol, and restrictions please refer to the Philip's I^2C standard

12.4 M-BUS PROTOCOL

Normally, a standard communication is composed of four parts: (1) START signal, (2) slave address transmission, (3) data transfer, and (4) STOP signal. They are described briefly in the following sections and illustrated in Figure 12-2.

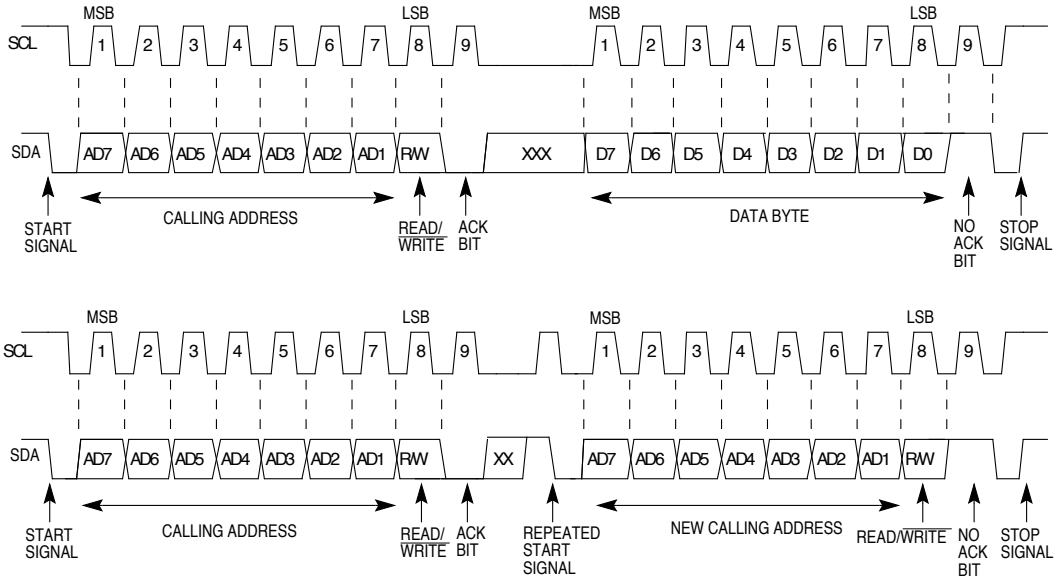


Figure 12-2. M-Bus Standard Communication Protocol

12.4.1 START Signal

When the bus is free, i.e., no master device is engaging the bus (both SCL and SDA lines are at logic high), a master can initiate communication by sending a START signal. As shown in Figure 12-2, a START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer (each data transfer can contain several bytes of data) and awakens all slaves.

12.4.2 Slave Address Transmission

The first byte of data transfered by the master immediately after the START signal is the slave address. This is a seven-bit calling address followed by a R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave data transfer direction. No two slaves in the system can have the same address.

immediately switch over to slave-receive mode and stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate loss of arbitration.

12.4.7 Clock Synchronization

Because wire-AND logic is performed on SCL line, a high-to-low transition on SCL line affects all the devices connected on the bus. The devices start counting their low period when the master drives the SCL line low. Once a device clock has gone low, it holds the SCL line low until the clock high state is reached. However, the change of low to high in this device clock may not change the state of the SCL line if another device clock is still within its low period. Therefore, synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see Figure 12-3). When all devices concerned have counted off their low period, the synchronized clock SCL line is released and pulled high. There is then no difference between the device clocks and the state of the SCL line and all the devices start counting their high periods. The first device to complete its high period pulls the SCL line low again.

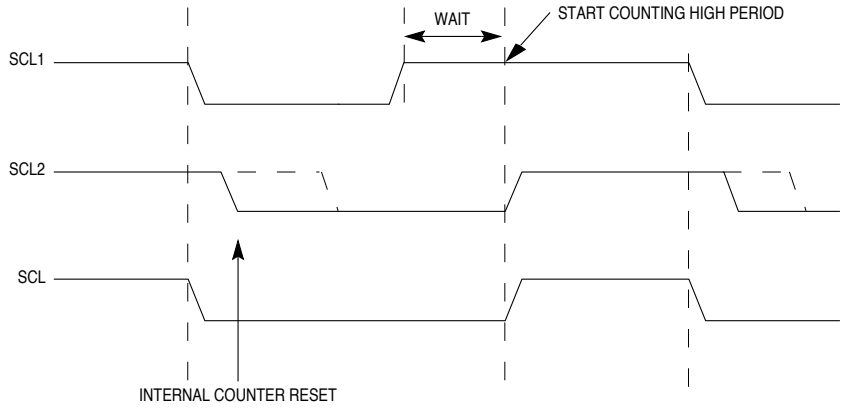


Figure 12-3. Synchronized Clock SCL

12.4.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices can hold the SCL low after completion of one byte transfer (9 bits). In such cases, it halts the bus clock and forces the master clock into wait states until the slave releases the SCL line.

12.4.9 Clock Stretching

Slaves can use the clock synchronization mechanism to slow down the transfer bit rate. After the master has driven SCL low the slave can drive SCL low for the required period and

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

CORE ADDRESS[1:0]	ACCESS SIZE	OPERAND LOCATION
10	Byte	Data[15:8]
11	Byte	Data[7:0]
0-	Word	Data[31:16]
1-	Word	Data[15:0]
--	Long	Data[31:0]

14.3.1.5 TRIGGER DEFINITION REGISTER (TDR). The TDR configures the operation of the hardware breakpoint logic within the Debug module and controls the actions taken under the defined conditions. The breakpoint logic can be configured as a one- or two-level trigger, where bits [29:16] of the TDR define the 2nd level trigger, bits [13:0] define the first level trigger, and bits [31:30] define the trigger response.

Reset clears the TDR.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRC		EBL	EDLW	EDWL	EDWU	EDLL	EDLM	EDUM	EDUU	DI	EAI	EAR	EAL	EPC	PCI

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
00		EBL	EDLW	EDWL	EDWU	EDLL	EDLM	EDUM	EDUU	DI	EAI	EAR	EAL	EPC	PCI

TDR Bit Definitions

TRC–Trigger Response Control

The trigger response control determines how the processor is to respond to a completed trigger condition. The trigger response is always displayed on the DDATA pins.

00=displayed on DDATA pins only
01=processor halt
10=debug interrupt
11=reserved

EBL–Enable Breakpoint Level

If set, this bit serves as the global enable for the breakpoint trigger. If cleared, all breakpoints are disabled.

EDLW–Enable Data Breakpoint for the Data Longword

If set, this bit enables the data breakpoint based on the core data bus (KD) KD[31:0] longword. The assertion of any of the ED bits enables the data breakpoint. If all bits are cleared, the data breakpoint is disabled.

EDWL–Enable Data Breakpoint for the Lower Data Word

If set, this bit enables the data breakpoint based on the KD[15:0] word.

EDWU–Enable Data Breakpoint for the Upper Data Word

If set, this bit enables the data breakpoint trigger based on the KD[31:16] word.

EDLL–Enable Data Breakpoint for the Lower Lower Data Byte

If set, this bit enables the data breakpoint trigger based on the KD[7:0] byte.

EDLM–Enable Data Breakpoint for the Lower Middle Data Byte

If set, this bit enables the data breakpoint trigger based on the KD[15:8] byte.

EDUM–Enable Data Breakpoint for the Upper Middle Data Byte

If set, this bit enables the data breakpoint trigger based on the KD[23:16] byte.

EDUU–Enable Data Breakpoint for the Upper Upper Data Byte

If set, this bit enables the data breakpoint trigger based on the KD[31:24] byte.

DI–Data Breakpoint Invert

This bit provides a mechanism to invert the logical sense of all the data breakpoint comparators. This can develop a trigger based on the occurrence of a data value not equal to the one programmed into the DBR.

The assertion of any of the EA bits enables the address breakpoint. If all three bits are cleared, this breakpoint is disabled.

EAI–Enable Address Breakpoint Inverted

If set, this bit enables the address breakpoint based outside the range defined by ABLR and ABHR.

EAR–Enable Address Breakpoint Range

If set, this bit enables the address breakpoint based on the inclusive range defined by ABLR and ABHR.

EAL–Enable Address Breakpoint Low

If set, this bit enables the address breakpoint based on the address contained in the ABLR.

EPC–Enable PC Breakpoint

If set, this bit enables the PC breakpoint. Clearing this bit disables the PC breakpoint.

PCI–PC Breakpoint Invert

If set, this bit allows execution outside a given region as defined by PBR and PBMR to enable a trigger. If cleared, the PC breakpoint is defined within the region defined by PBR and PBMR.

14.3.1.6 CONFIGURATION/STATUS REGISTER (CSR). The Configuration/Status Register defines the operating configuration for the processor and memory subsystem. In

15.5 RESTRICTIONS

The test logic is implemented using static logic design, and TCK can be stopped in either a high or low state without loss of data. The system logic, however, operates on a different system clock which is not synchronized to TCK internally. Any mixed operation requiring the use of 1149.1 test logic in conjunction with system functional logic that uses both clocks, must have coordination and synchronization of these clocks done externally to the MCF5206.

15.6 DISABLING THE IEEE 1149.1 STANDARD OPERATION

There are two methods by which the MCF5206 can be used without the IEEE 1149.1 test logic being active: (1) Nonuse of the JTAG test logic by either nontermination (disconnection) or intentional fixing of TAP logic values, and (2) Intentional disabling of the JTAG test logic by assertion of the $\overline{\text{JTAG}}$ signal (entering Debug mode).

There are several considerations that must be addressed if the IEEE 1149.1 logic is not going to be used once the MCF5206 is assembled onto a board. The prime consideration is to ensure that the IEEE 1149.1 test logic remains transparent and benign to the system logic during functional operation. This requires the minimum of either connecting the $\overline{\text{TRST}}$ pin to logic 0, or connecting the TCK clock pin to a clock source that supplies five rising edges and the falling edge after the fifth rising edge, to ensure that the part enters the test-logic-reset state. The recommended solution is to connect $\overline{\text{TRST}}$ to logic 0. Another consideration is that the TCK pin does not have an internal pullup as is required on the TMS, TDI, and $\overline{\text{TRST}}$ pins; therefore, it should not be left unterminated to preclude mid-level input values. Figure 15-3 shows pin values recommended for disabling JTAG with the MCF5206 in JTAG mode ($\overline{\text{JTAG}}$=0).

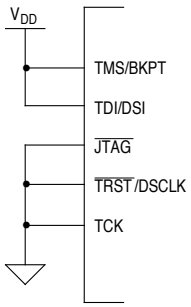
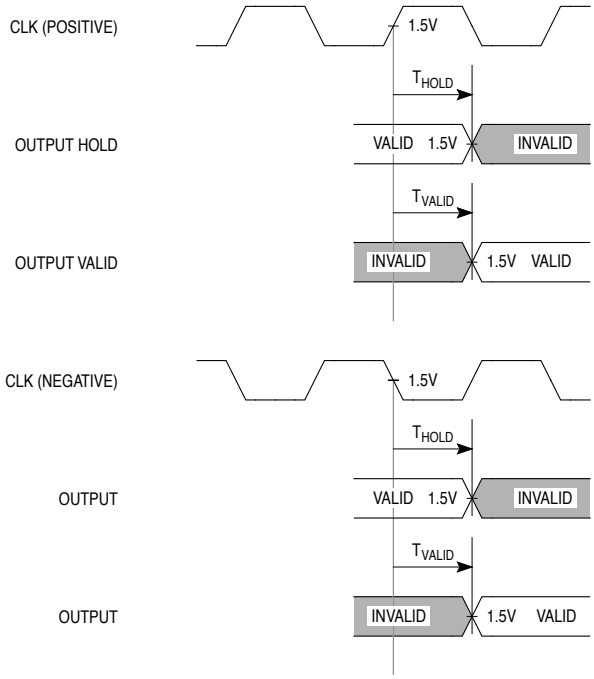


Figure 15-3. Disabling JTAG in JTAG Mode

A second method of using the MCF5206 without the IEEE 1149.1 logic being active is to select Debug mode by placing a logic 1 on the defined compliance enable pin, $\overline{\text{JTAG}}$. When $\overline{\text{JTAG}}$ is a logic 1, then the IEEE 1149.1 test controller is placed in the test-logic-reset state by the internal assertion of the $\overline{\text{TRST}}$ signal to the controller, and, the TAP pins function as debug mode pins. While in JTAG mode, input pins TDI/DSI, TMS/$\overline{\text{BKPT}}$, and

16.3.6 Output Timing Waveform Diagram



Output Timing Waveform

APPENDIX A
MCF5206 MEMORY MAP SUMMARY

This section is a summary chart of the entire memory map for the MCF5206.

Table A-1. MCF5206 User Programming Model

ADDRESS	NAME	WIDTH	DESCRIPTION	RESET VALUE	ACCESS
MOVEC with $C0F	MBAR	32	Module Base Address Register	uninitialized (except V=0)	W
MBAR + $003	SIMR	8	SIM Configuration Register	$C0	R/W
MBAR + $014	ICR1	8	Interrupt Control Register 1 - External IRQ1/IPL1	$04	R/W
MBAR + $015	ICR2	8	Interrupt Control Register 2 - External IPL2	$08	R/W
MBAR + $016	ICR3	8	Interrupt Control Register 3 - External IPL3	$0C	R/W
MBAR + $017	ICR4	8	Interrupt Control Register 4 - External IRQ4/IPL4	$10	R/W
MBAR + $018	ICR5	8	Interrupt Control Register 5 - External IPL5	$14	R/W
MBAR + $019	ICR6	8	Interrupt Control Register 6 - External IPL6	$18	R/W
MBAR + $01A	ICR7	8	Interrupt Control Register 7 - External IRQ7/IPL7	$1C	R/W
MBAR + $01B	ICR8	8	Interrupt Control Register 8 - SWT	$1C	R/W
MBAR + $01C	ICR9	8	Interrupt Control Register 9 - Timer 1 Interrupt	$80	R/W
MBAR + $01D	ICR10	8	Interrupt Control Register 10 - Timer 2 Interrupt	$80	R/W
MBAR + $01E	ICR11	8	Interrupt Control Register 11 - MBUS Interrupt	$80	R/W
MBAR + $01F	ICR12	8	Interrupt Control Register 12 - UART 1 Interrupt	$00	R/W
MBAR + $020	ICR13	8	Interrupt Control Register 13 - UART2 Interrupt	$00	R/W
MBAR + $036	IMR	16	Interrupt Mask Register	$3FFE	R/W
MBAR + $03A	IPR	16	Interrupt Pending Register	$0000	R
MBAR + $040	RSR	8	Reset Status Register	$80 or $20	R/W
MBAR + $041	SYPCR	8	System Protection Control Register	$00	R/W
MBAR + $042	SWIVR	8	Software Watchdog Interrupt Vector Register	$0F	R/W
MBAR + $043	SWSR	8	Software Watchdog Service Register	uninitialized	W
MBAR + $046	DCRR	16	DRAM Controller Refresh	Master Reset: $0000 Normal Reset: uninitialized	R/W
MBAR + $04A	DCTR	16	DRAM Controller Timing Register	Master Reset: $0000 Normal Reset: uninitialized	R/W
MBAR + $04C	DCAR0	16	DRAM Controller Address Register - Bank 0	Master Reset: uninitialized Normal Reset: uninitialized	R/W
MBAR + $050	DCMR0	32	DRAM Controller Mask Register - Bank 0	Master Reset: uninitialized Normal Reset: uninitialized	R/W
MBAR + $057	DCCR0	8	DRAM Controller Control Register- Bank 0	Master Reset: $00 Normal Reset: $00	R/W
MBAR + $058	DCAR1	16	DRAM Controller Address Register - Bank 1	Master Reset: uninitialized Normal Reset: uninitialized	R/W

Appendix A

ADDRESS	NAME	WIDTH	DESCRIPTION	RESET VALUE	ACCESS
MBAR+$1E4	MFDR	8	M-Bus Frequency Divider Register	$00	R/W
MBAR+$1E8	MBCR	8	M-Bus Control Register	$00	R/W
MBAR+$1EC	MBSR	8	M-Bus Status Register	$00	R/W
MBAR+$1F0	MBDR	8	M-Bus Data I/O Register	$00	R/W