Details	
Product Status	Not For New Designs
Core Processor	CPU12
Core Size	16-Bit
Speed	8MHz
Connectivity	SCI, SPI
Peripherals	POR, PWM, WDT
Number of I/O	63
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	768 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/product-detail/nxp-semiconductors/mchc912b32cfue8

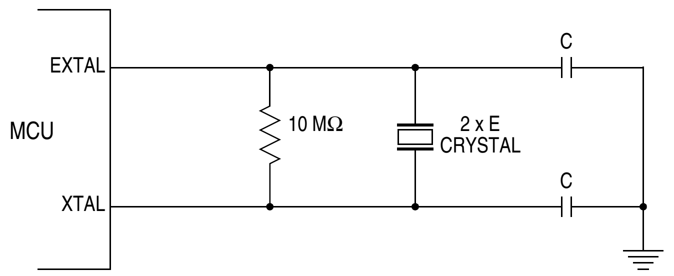

Figure 1-5. Common Crystal Connections

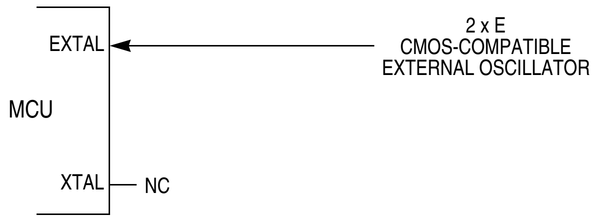

Figure 1-6. External Oscillator Connections

1.6.3.2 ECLK

ECLK is the output connection for the internal bus clock and is used to demultiplex the address and data and is used as a timing reference. ECLK frequency is equal to one half the crystal frequency out of reset.

In normal single-chip mode, the E-clock output is off at reset to reduce the effects of radio frequency interference (RFI), but it can be turned on if necessary.

In special single-chip mode, the E-clock output is on at reset but can be turned off.

In special peripheral mode, the E clock is an input to the MCU.

All clocks, including the E clock, are halted when the MCU is in stop mode. It is possible to configure the MCU to interface to slow external memory. ECLK can be stretched for such accesses.

1.6.3.3 $\overline{\text{RESET}}$

An active-low, bidirectional control signal, $\overline{\text{RESET}}$ is an input to initialize the MCU to a known startup state. It also acts as an open-drain output to indicate that an internal failure has been detected in either the clock monitor or COP watchdog circuit. The MCU goes into reset asynchronously and comes out of reset synchronously. This allows the part to reach a proper reset state even if the clocks have failed, while allowing synchronized operation when starting out of reset.

It is possible to determine whether a reset was caused by an internal source or an external source. An internal source drives the pin low for 16 cycles; eight cycles later, the pin is sampled. If the pin has returned high, either the COP watchdog vector or clock monitor vector is taken. If the pin is still low, the external reset is determined to be active and the reset vector is taken. Hold reset low for at least 32 cycles to assure that the reset vector is taken in the event that an internal COP watchdog timeout or clock monitor fail occurs.

1.6.3.4 $\overline{\text{IRQ}}$

$\overline{\text{IRQ}}$ is the maskable external interrupt request pin. It provides a means of applying asynchronous interrupt requests to the MCU. Either falling edge-sensitive triggering or level-sensitive triggering is program

4.4 Latching of Interrupts

$\overline{\text{XIRQ}}$ is always level triggered and $\overline{\text{IRQ}}$ can be selected as a level-triggered interrupt. These level-triggered interrupt pins should be released only during the appropriate interrupt service routine. Generally, the interrupt service routine will handshake with the interrupting logic to release the pin. In this way, the MCU will never start the interrupt service sequence only to determine that there is no longer an interrupt source. In the event that this does occur, the trap vector will be taken.

If $\overline{\text{IRQ}}$ is selected as an edge-triggered interrupt, the hold time of the level after the active edge is independent of when the interrupt is serviced. As long as the minimum hold time is met, the interrupt will be latched inside the MCU. In this case, the IRQ edge interrupt latch is cleared automatically when the interrupt is serviced.

All of the remaining interrupts are latched by the MCU with a flag bit. These interrupt flags should be cleared during an interrupt service routine or when the interrupts are masked by the I bit. By doing this, the MCU will never get an unknown interrupt source and take the trap vector.

4.5 Interrupt Control and Priority Registers

This section describes the interrupt control and priority registers.

4.5.1 Interrupt Control Register

Address: $001E

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	IRQE	IRQEN	DLY	0	0	0	0	0
Reset:	0	1	1	0	0	0	0	0

Figure 4-1. Interrupt Control Register (INTCR)

Read: Anytime

Write: Varies from bit to bit

IRQE — $\overline{\text{IRQ}}$ Edge-Sensitive Only Bit

IRQE can be written once in normal modes. In special modes, IRQE can be written anytime, but the first write is ignored.

1 = $\overline{\text{IRQ}}$ pin responds only to falling edges.

0 = $\overline{\text{IRQ}}$ pin responds to low levels.

IRQEN — External $\overline{\text{IRQ}}$ Enable Bit

IRQEN can be written anytime in all modes. The $\overline{\text{IRQ}}$ pin has an internal pullup.

1 = $\overline{\text{IRQ}}$ pin connected to interrupt logic

0 = $\overline{\text{IRQ}}$ pin disconnected from interrupt logic

DLY — Oscillator Startup Delay on Exit from Stop Mode Bit

DLY can be written once in normal modes. In special modes, DLY can be written anytime. The delay time of about 4096 cycles is based on the E-clock rate.

1 = Stabilization delay on exit from stop mode

0 = No stabilization delay on exit from stop mode

Chapter 6
Bus Control and Input/Output (I/O)

6.1 Introduction

Internally, the MCU has full 16-bit data paths, but depending upon the operating mode and control registers, the external bus may be eight or 16 bits. There are cases where 8-bit and 16-bit accesses can appear on adjacent cycles using the $\overline{\text{LSTRB}}$ signal to indicate 8-bit or 16-bit data.

6.2 Detecting Access Type from External Signals

The external signals $\overline{\text{LSTRB}}$, R/$\overline{\text{W}}$, and A0 can be used to determine the type of bus access that is taking place. Accesses to the internal RAM module are the only accesses that produce $\overline{\text{LSTRB}}$ = A0 = 1, because the internal RAM is specifically designed to allow misaligned 16-bit accesses in a single cycle. In these cases, the data for the address that was accessed is on the low half of the data bus and the data for address +1 is on the high half of the data bus (data order is swapped).

Table 6-1. Detecting Access Type

$\overline{\text{LSTRB}}$	A0	R/$\overline{\text{W}}$	Type of Access
1	0	1	8-bit read of an even address
0	1	1	8-bit read of an odd address
1	0	0	8-bit write to an even address
0	1	0	8-bit write to an odd address
0	0	1	16-bit read of an even address
1	1	1	16-bit read of an odd address (low/high data swapped)
0	0	0	16-bit write to an even address
1	1	0	16-bit write to an odd address (low/high data swapped)

6.3 Registers

Under certain conditions, not all registers are visible in the memory map. In special peripheral mode, the first 16 registers associated with bus expansion are removed from the memory map.

In expanded modes, some or all of port A, port B, and port E are used for expansion buses and control signals. To allow emulation of the single-chip functions of these ports, some of these registers must be rebuilt in an external port replacement unit. In any expanded mode,
port A and port B are used for address and data lines so registers for these ports, as well as the data direction registers for these ports, are removed from the on-chip memory map and become external accesses.

In any expanded mode, port E pins may be needed for bus control (for example, ECLK and R/$\overline{\text{W}}$). To regain the single-chip functions of port E, the emulate port E (EME) control bit in the MODE register may

6.3.3 Port B Data Register

Address: $0001

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Reset:	Unaffected by reset							
Alternate functions:								
Expanded wide and peripheral:	ADDR7 DATA7	ADDR6 DATA6	ADDR5 DATA5	ADDR4 DATA4	ADDR3 DATA3	ADDR2 DATA2	ADDR1 DATA1	ADDR0 DATA0
Expanded narrow:	ADDR7	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0

Figure 6-3. Port B Data Register (PORTB)

Read: Anytime, if register is in the map

Write: Anytime, if register is in the map

Bits PB7–PB0 are associated with addresses ADDR7–ADDR0 and DATA7–DATA0. When port B is not used for external addresses and data such as in single-chip mode, these pins can be used as general-purpose I/O. DDRB determines the primary direction of each pin. This register is not in the on-chip map in expanded and peripheral modes.

6.3.4 Port B Data Direction Register

Address: $0003

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0
Reset:	0	0	0	0	0	0	0	0

Figure 6-4. Port B Data Direction Register (DDRB)

Read: Anytime, if register is in the map

Write: Anytime, if register is in the map

This register determines the primary direction for each port B pin when functioning as a general-purpose I/O port. DDRB is not in the on-chip map in expanded and peripheral modes.

1 = Associated pin is an output.

0 = Associated pin is a high-impedance input.

6.3.8 Pullup Control Register

Address: $000C

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	PUPE	0	0	PUPB	PUPA
Write:	▨	▨	▨		▨	▨		
Reset:	0	0	0	1	0	0	0	0

▨ = Unimplemented

Figure 6-8. Pullup Control Register (PUCR)

Read: Anytime, if register is in the map

Write: Anytime, if register is in the map

These bits select pullup resistors for any pin in the corresponding port that is currently configured as an input. This register is not in the map in peripheral mode.

PUPE — Pullup Port E Enable Bit

Pin PE1 always has a pullup. Pins PE6, PE5, and PE4 never have pullups.

1 = Enable port E pullups on PE7, PE3, PE2, PE1, and PE0
0 = Disable port E pullups on PE7, PE3, PE2, PE1, and PE0

PUPB — Pullup Port B Enable Bit

1 = Enable pullups for all port B input pins
0 = Disable port B pullups

This bit has no effect if port B is used as part of the address/data bus (pullups are inactive).

PUPA — Pullup Port A Enable Bit

0 = Disable port A pullups
1 = Enable pullups for all port A input pins

This bit has no effect, if port A is used as part of the address/data bus (pullups are inactive).

FCOP — Force COP Watchdog Reset Bit

Writes are not allowed in normal modes; can be written anytime in special modes.
If DISR is set, this bit has no effect.

0 = Normal operation
1 = Force a COP reset, if COP is enabled.

DISR — Disable Resets from COP Watchdog and Clock Monitor Bit

Writes are not allowed in normal modes, anytime in special modes.

0 = Normal operation
1 = Regardless of other control bit states, COP and clock monitor do not generate a system reset.

CR2, CR1, and CR0 — COP Watchdog Timer Rate Select Bit

The COP system is driven by a constant frequency of E/2^{13}. (RTBYP in the RTICTL register allows all but two stages of this divider to be bypassed for testing in special modes only.) These bits specify an additional division factor to arrive at the COP timeout rate. The clock used for this module is the E clock.
Write once in normal modes, anytime in special modes.

Table 10-4. COP Watchdog Rates (RTBYP = 0)

CR2	CR1	CR0	Divide E By:	At E = 4.0-MHz Timeout 0 to +2.048 ms	At E = 8.0-MHz Timeout 0 to +1.024 ms
0	0	0	OFF	OFF	OFF
0	0	1	2^{13}	2.048 ms	1.024 ms
0	1	0	2^{15}	8.1920 ms	4.096 ms
0	1	1	2^{17}	32.768 ms	16.384 ms
1	0	0	2^{19}	131.072 ms	65.536 ms
1	0	1	2^{21}	524.288 ms	262.144 ms
1	1	0	2^{22}	1.048 s	524.288 ms
1	1	1	2^{23}	2.097 s	1.048576 s

10.7.5 Arm/Reset COP Timer Register

Address: $0017

	Bit 7	6	5	4	3	2	1	Bit 0
Read: / Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 10-9. Arm/Reset COP Timer Register (COPRST)

Always reads $00.

Writing $55 to this address is the first step of the COP watchdog sequence.

Writing $AA to this address is the second step of the COP watchdog sequence. Other instructions may be executed between these writes but both must be completed in the correct order prior to timeout to avoid a watchdog reset. Writing anything other than $55 or $AA causes a COP reset to occur.

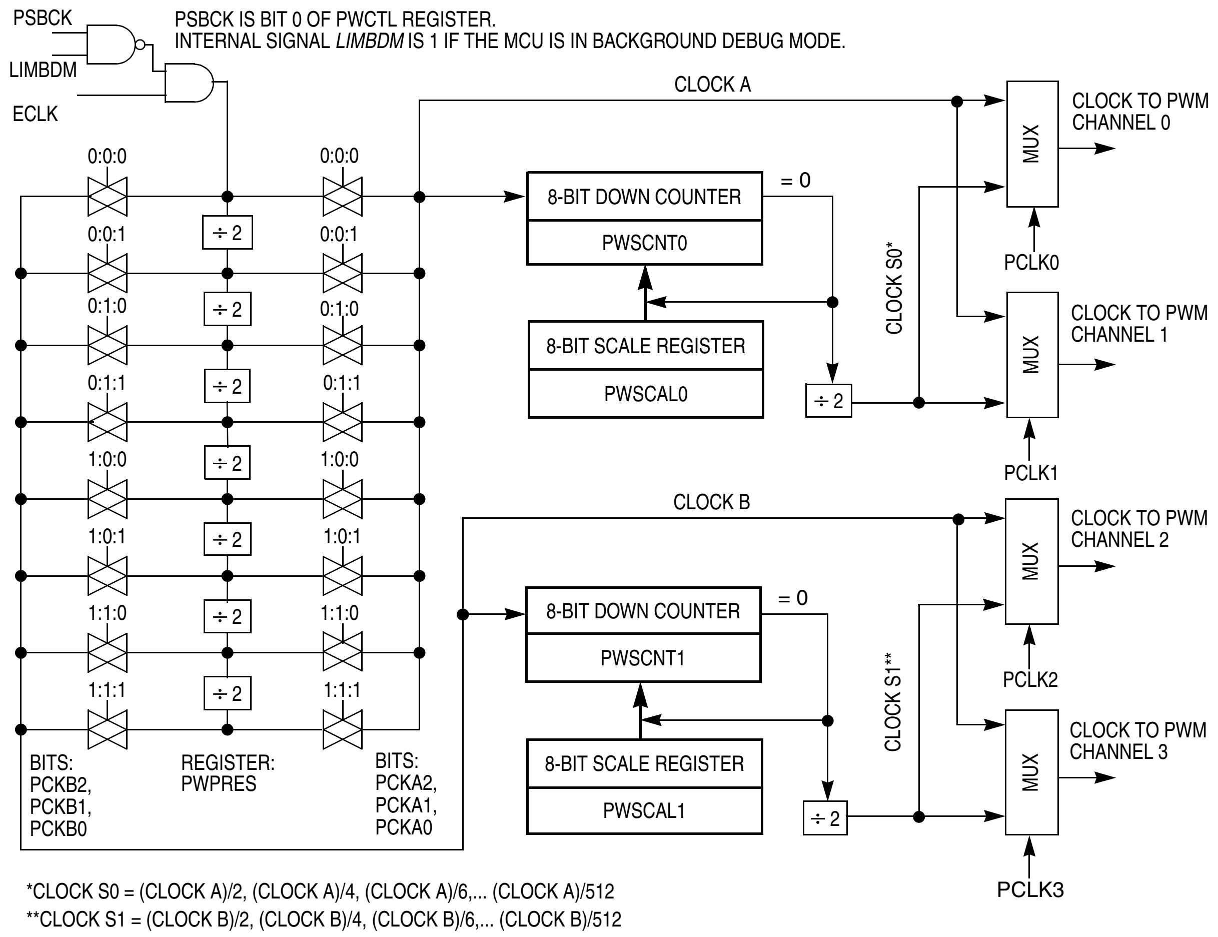

Figure 11-3. PWM Clock Sources

12.3.4 Timer Count Register

Address: $0084

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

Address: $0085

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-6. Timer Count Register (TCNT)

Read: Anytime

Write: Has no meaning or effect in normal mode; only writable in special modes

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give a different result than accessing them as a word.

The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

12.3.5 Timer System Control Register

Address: $0086

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TEN	TSWAI	TSBCK	TFFCA	0	0	0	0
Write:					▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 12-7. Timer System Control Register (TSCR)

Read: Anytime

Write: Anytime

TEN — Timer Enable Bit

If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator since the E ÷ 64 is generated by the timer prescaler.

0 = Disables timer, including the counter; can be used for reducing power consumption
1 = Allows timer to function normally

TSWAI — Timer Stops While in Wait Bit

Timer interrupts cannot be used to get the MCU out of wait.

0 = Allows timer to continue running during wait
1 = Disables timer when MCU is in wait mode

PAEN — Pulse Accumulator System Enable Bit

0 = Pulse accumulator system disabled
1 = Pulse accumulator system enabled

PAEN is independent from TEN.

PAMOD — Pulse Accumulator Mode Bit

0 = Event counter mode
1 = Gated time accumulation mode

PEDGE — Pulse Accumulator Edge Control Bit

For PAMOD = 0 (event counter mode)

0 = Falling edges on the pulse accumulator input pin (PT7/PAI) cause the count to be incremented.
1 = Rising edges on the pulse accumulator input pin cause the count to be incremented.

For PAMOD = 1 (gated time accumulation mode)

0 = Pulse accumulator input pin high enables E ÷ 64 clock to pulse accumulator and the trailing falling edge on the pulse accumulator input pin sets the PAIF flag.
1 = Pulse accumulator input pin low enables E ÷ 64 clock to pulse accumulator and the trailing rising edge on the pulse accumulator input pin sets the PAIF flag.

If the timer is not active (TEN = 0 in TSCR), there is no ÷64 clock since the E ÷ 64 clock is generated by the timer prescaler.

CLK1 and CLK0 — Clock Select Bits

Table 12-4. Clock Selection

CLK1	CLK0	Selected Clock
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written.

PAOVI — Pulse Accumulator Overflow Interrupt Enable Bit

0 = Interrupt inhibited
1 = Interrupt requested if PAOVF is set

PAI — Pulse Accumulator Input Interrupt Enable Bit

0 = Interrupt inhibited
1 = Interrupt requested if PAIF is set

MCEN — Modulus Down-Counter Enable Bit

When MCEN = 0, the counter is preset to $FFFF. This will prevent an early interrupt flag when the modulus down-counter is enabled.

0 = Modulus counter disabled.
1 = Modulus counter is enabled.

MCPR1 and MCPR0 — Modulus Counter Prescaler Select Bits

These two bits specify the division rate of the modulus counter prescaler. The newly selected prescaler division rate will not be effective until a load of the load register into the modulus counter count register occurs.

MCPR1	MCPR0	Prescalar Division Rate
0	0	1
0	1	4
1	0	8
1	1	16

13.4.15 16-Bit Modulus Down-Counter Flag Register

Address: $00A7

	Bit 7	6	5	4	3	2	1	Bit 0
Read: / Write:	MCZF	0	0	0	POLF3	POLF2	POLF1	POLF0
Reset:	0	0	0	0	0	0	0	0

Figure 13-36. 16-Bit Modulus Down-Counter Flag Register (MCFLG)

Read: Anytime

Write: Only for clearing bit 7

MCZF — Modulus Counter Underflow Interrupt Flag

The flag is set when the modulus down-counter reaches $0000. Writing 1 to this bit clears the flag. Writing 0 has no effect. Any access to the MCCNT register will clear the MCZF flag in this register when TFFCA bit in register TSCR ($86) is set.

POLF3–POLF0 — First Input Capture Polarity Status Bits

This are read-only bits. Writing to these bits has no effect. Each status bit gives the polarity of the first edge which has caused an input capture to occur after capture latch has been read. Each POLFx corresponds to a timer PORTx input.

0 = The first input capture has been caused by a falling edge.
1 = The first input capture has been caused by a rising edge.

13.4.26 Modulus Down-Counter Count Registers

Address: $00B6

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	1	1	1	1	1	1	1	1

Address: $00B7

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	1	1	1	1	1	1	1	1

Figure 13-50. Modulus Down-Counter Count Registers (MCCNT)

Read: Anytime

Write: Anytime

A full access for the counter register should take place in one clock cycle. A separate read/write for high byte and low byte will give different results than accessing them as a word. If the RDMCL bit in MCCTL register is cleared, reads of the MCCNT register will return the present value of the count register. If the RDMCL bit is set, reads of the MCCNT will return the contents of the load register.

If a $0000 is written into MCCNT and modulus counter while LATQ and BUFEN in ICSYS ($AB) register are set, the input capture and pulse accumulator registers will be latched. With a $0000 write to the MCCNT, the modulus counter will stay at 0 and does not set the MCZF flag in MCFLG register.

If modulus mode is enabled (MODMC = 1), a write to this address will update the load register with the value written to it. The count register will not be updated with the new value until the next counter underflow. The FLMC bit in MCCTL ($A6) can be used to immediately update the count register with the new value if an immediate load is desired.

If modulus mode is not enabled (MODMC = 0), a write to this address will clear the prescaler and will immediately update the counter register with the value written to it and down-counts once to $0000.

13.4.27 Timer Input Capture Holding Registers

Address: $00B8

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset:	0	0	0	0	0	0	0	0

Address: $00B9

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset:	0	0	0	0	0	0	0	0

Figure 13-51. Timer Input Capture Holding Register 0 (TC0H)

TMIFR0 — Transmit Multiple Byte IFR without CRC (Type 3)

The TMIFR0 bit is used to request the BDLC to transmit the byte in the BDLC data register (BDR) as the first byte of a multiple byte IFR without CRC. Response IFR bytes are still subject to J1850 message length maximums (see 15.7.2 J1850 Frame Format and Figure 15-14).

1 = If set prior to a valid EOD being received with no CRC error, once the EOD symbol has been received, the BDLC attempts to transmit the appropriate normalization bit followed by IFR bytes. The programmer should set TEOD after the last IFR byte has been written into the BDR. After TEOD has been set, the last IFR byte to be transmitted is the last byte which was written into the BDR.

0 = Bit is cleared automatically once the BDLC has successfully transmitted the EOD symbol, by the detection of an error on the multiplex bus or by a transmitter underrun caused when the programmer does not write another byte to the BDR after the TDRE interrupt.

If the TMIFR0 bit is set, the BDLC attempts to transmit the normalization symbol followed by the byte in the BDR. After the byte in the BDR has been loaded into the transmit shift register, a TDRE interrupt (see 15.9.3 BDLC State Vector Register) occurs similar to the main message transmit sequence. The programmer should then load the next byte of the IFR into the BDR for transmission. When the last byte of the IFR has been loaded into the BDR, the programmer should set the TEOD bit in the BCR2. This instructs the BDLC to transmit an EOD symbol once the byte in the BDR is transmitted, indicating the end of the IFR portion of the message frame. The BDLC does not append a CRC when the TMIFR0 is set.

If the programmer attempts to set the TMIFR0 bit after the EOD symbol has been received from the bus, the TMIFR0 bit remains in the reset state, and no attempt is made to transmit an IFR byte.

If a loss of arbitration occurs when the BDLC is transmitting, the TMIFR0 bit is cleared, and no attempt is made to retransmit the byte in the BDR. If loss of arbitration occurs in the last bit of the IFR byte, two additional 1 bits are sent out.

NOTE

The extra logic 1 bits are an enhancement to the J1850 protocol which forces a byte boundary condition fault. This is helpful in preventing noise onto the J1850 bus from a corrupted message.

TMIFR1 — Transmit Multiple Byte IFR with CRC Bit (Type 3)

The TMIFR1 bit requests the BDLC to transmit the byte in the BDLC data register (BDR) as the first byte of a multiple byte IFR with CRC or as a single byte IFR with CRC. Response IFR bytes are still subject to J1850 message length maximums (see 15.7.2 J1850 Frame Format and Figure 15-14).

1 = If this bit is set prior to a valid EOD being received with no CRC error, once the EOD symbol has been received, the BDLC attempts to transmit the appropriate normalization bit followed by IFR bytes. The programmer should set TEOD after the last IFR byte has been written into the BDR. After TEOD has been set and the last IFR byte has been transmitted, the CRC byte is transmitted.

0 = The bit is cleared automatically, once the BDLC has successfully transmitted the CRC byte and EOD symbol, by the detection of an error on the multiplex bus or by a transmitter underrun caused when the programmer does not write another byte to the BDR after the TDRE interrupt.

If the TMIFR1 bit is set, the BDLC attempts to transmit the normalization symbol followed by the byte in the BDR. After the byte in the BDR has been loaded into the transmit shift register, a TDRE interrupt (see 15.9.3 BDLC State Vector Register) occurs similar to the main message transmit sequence. The programmer should then load the next byte of the IFR into the BDR for transmission. When the last byte of the IFR has been loaded into the BDR, the programmer should set the TEOD bit in the BDLC control register 2 (BCR2). This instructs the BDLC to transmit a CRC byte once the byte in the BDR is transmitted, and then transmit an EOD symbol, indicating the end of the IFR portion of the message frame.

However, to transmit a single byte followed by a CRC byte, the programmer should load the byte into the BDR before the EOD symbol has been received, and then set the TMIFR1 bit. Once the TDRE interrupt occurs, the programmer sets the TEOD bit in the BCR2. This results in the byte in the BDR being the only byte transmitted before the IFR CRC byte, and no TDRE interrupt is generated.

If the programmer attempts to set the TMIFR1 bit immediately after the EOD symbol has been received from the bus, the TMIFR1 bit remains in the reset state, and no attempt is made to transmit an IFR byte.

If a loss of arbitration occurs when the BDLC is transmitting any byte of a multiple byte IFR, the BDLC goes to the loss of arbitration state, sets the appropriate flag, and ceases transmission.

If the BDLC loses arbitration during the IFR, the TMIFR1 bit is cleared and no attempt is made to retransmit the byte in the BDR. If loss of arbitration occurs in the last bit of the IFR byte, two additional 1 bits are sent out.

NOTE

The extra logic 1 bits are an enhancement to the J1850 protocol which forces a byte boundary condition fault. This is helpful in preventing noise on the J1850 bus from corrupting a message.

15.9.3 BDLC State Vector Register

Address: $00F9

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	I3	I2	I1	I0	0	0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 15-15. BDLC State Vector Register (BSVR)

This register is provided to substantially decrease the CPU overhead associated with servicing interrupts while under operation of a multiplex protocol. It provides an index offset that is directly related to the BDLC's current state, which can be used with a user-supplied jump table to rapidly enter an interrupt service routine. This eliminates the need for the user to maintain a duplicate state machine in software.

I0, I1, I2, I3 — Interrupt Source Bits

These bits indicate the source of the pending interrupt request. Bits are encoded according to Table 15-4.

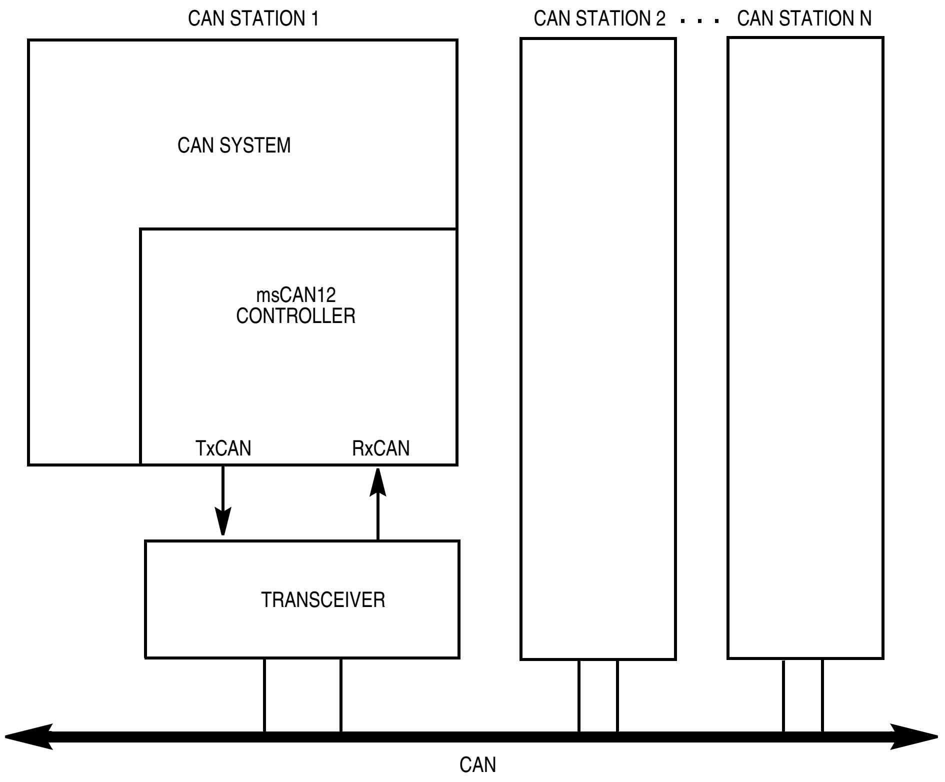

Figure 16-1. Typical CAN System with msCAN12

16.3.1 Background

Modern application layer software is built on two fundamental assumptions:

1. Any CAN node is able to send out a stream of scheduled messages without releasing the bus between two messages. Such nodes will arbitrate for the bus right after sending the previous message and will only release the bus in case of lost arbitration.
2. The internal message queue within any CAN node is organized so that the highest priority message will be sent out first if more than one message is ready to be sent.

This behavior cannot be achieved with a single transmit buffer. That buffer must be reloaded right after the previous message has been sent. This loading process lasts a definite amount of time and has to be completed within the inter-frame sequence (IFS) to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the central processor unit (CPU) reacts with short latencies to the transmit interrupt.

A double buffer scheme would decouple the reloading of the transmit buffers from the actual message being sent and as such reduces the reaction requirements on the CPU. Problems may arise if the sending of a message would be finished just while the CPU reloads the second buffer. In that case, no buffer would then be ready for transmission and the bus would be released.

At least three transmit buffers are required to meet the first requirement under all circumstances. The msCAN12 has three transmit buffers.

The second requirement calls for some sort of internal priorization which the msCAN12 implements with the “local priority” concept described here.

Table 16-1. msCAN12 Interrupt Vectors

Function	Source	Local Mask	Global Mask
Wakeup	WUPIF	WUPIE	I bit
Error interrupts	RWRNIF	RWRNIE	
	TWRNIF	TWRNIE	
	RERRIF	RERRIE	
	TERRIF	TERRIE	
	BOFFIF	BOFFIE	
	OVRIF	OVRIE	
Receive	RXF	RXFIE	
Transmit	TXE0	TXEIE0	
	TXE1	TXEIE1	
	TXE2	TXEIE2	

16.6 Protocol Violation Protection

The msCAN12 will protect the user from accidentally violating the CAN protocol through programming errors. The protection logic implements these features:

- The receive and transmit error counters cannot be written or otherwise manipulated.
- All registers which control the configuration of the msCAN12 cannot be modified while the msCAN12 is online. The SFTRES bit in CMCR0 (see **16.12.1 msCAN12 Module Control Register 0**) serves as a lock to protect these registers:
  - msCAN12 module control register 1 (CMCR1)
  - msCAN12 bus timing register 0 and 1 (CBTR0 and CBTR1)
  - msCAN12 identifier acceptance control register (CIDAC)
  - msCAN12 identifier acceptance registers (CIDAR0–CIDAR7)
  - msCAN12 identifier mask registers (CIDMR0–CIDMR7)
- The TxCAN pin is forced to recessive if the CPU goes into stop mode.

16.7 Low-Power Modes

In addition to normal mode, the msCAN12 has three modes with reduced power consumption compared to normal mode. In sleep and soft-reset mode, power consumption is reduced by stopping all clocks except those to access the registers. In power-down mode, all clocks are stopped and no power is consumed.

The wait-for-interrupt (WAI) and STOP instructions put the MCU in low power-consumption standby modes. Table 16-2 summarizes the combinations of msCAN12 and CPU modes. A particular combination of modes is entered for the given settings of the bits CSWAI, SLPAK, and SFTRES. For all modes, an msCAN wakeup interrupt can occur only if SLPAK = WUPIE = 1. While the CPU is in wait mode, the msCAN12 can be operated in normal mode and emit interrupts. (Registers can be accessed via background debug mode.)

SFTRES — Soft-Reset Bit

When this bit is set by the CPU, the msCAN12 immediately enters the soft-reset state. Any on-going transmission or reception is aborted and synchronization to the bus is lost.

These registers will go into and stay in the same state as out of hard reset: CMCR0, CRFLG, CRIER, CTFLG, and CTCR.

Registers CMCR1, CBTR0, CBTR1, CIDAC, CIDAR0–CIDAR7, and CIDMR0–CIDMR7 can be written only by the CPU when the msCAN12 is in soft-reset state. The values of the error counters are not affected by soft reset.

When this bit is cleared by the CPU, the msCAN12 will try to synchronize to the CAN bus. For example, if the msCAN12 is not in bus-off state, it will be synchronized after 11 recessive bits on the bus; if the msCAN12 is in bus-off state, it continues to wait for 128 occurrences of 11 recessive bits.

Clearing SFTRES and writing to other bits in CMCR0 must be in separate instructions.

0 = Normal operation
1 = msCAN12 in soft-reset state

16.12.2 msCAN12 Module Control Register 1

Address: $0101

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	LOOPB	WUPM	CLKSRC
Write:	▨	▨	▨	▨	▨			
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 16-17. msCAN12 Module Control Register 1 (CMCR1)

LOOPB — Loop Back Self-Test Mode Bit

When this bit is set, the msCAN12 performs an internal loop back which can be used for self-test operation. The bit stream output of the transmitter is fed back to the receiver. The RxCAN input pin is ignored and the TxCAN output goes to the recessive state (1). In this state the msCAN12 ignores the bit sent during the ACK slot of the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

0 = Normal operation
1 = Activate loop back self-test mode

NOTE

The ACK bit is added to the CAN frame by the protocol. For more information on the CAN frame and the ACK bit, refer to the Bosch CAN 2.0 specification.

WUPM — Wakeup Mode Flag

This flag defines whether the integrated low-pass filter is applied to protect the msCAN12 from spurious wakeups. See **16.7.4 Programmable Wakeup Function**.

0 = msCAN12 will wake up the CPU after any recessive-to- dominant edge on the CAN bus.
1 = msCAN12 will wake up the CPU only in the case of a dominant pulse on the bus which has a length of approximately t_{WUP}.

IDAM1 and IDAM0— Identifier Acceptance Mode Flags

The CPU sets these flags to define the identifier acceptance filter organization. See **16.4 Identifier Acceptance Filter**. Table 16-8 summarizes the different settings. In filter closed mode, no messages are accepted so that the foreground buffer is never reloaded.

Table 16-9. Identifier Acceptance Mode Settings

IDAM1	IDAM0	Identifier Acceptance Mode
0	0	Two 32-bit acceptance filters
0	1	Four 16-bit acceptance filters
1	0	Eight 8-bit acceptance filters
1	1	Filter closed

IDHIT2–IDHIT0— Identifier Acceptance Hit Indicator Flags

The msCAN12 sets these flags to indicate an identifier acceptance hit. See **16.4 Identifier Acceptance Filter**. Table 16-8 summarizes the different settings.

Table 16-10. Identifier Acceptance Hit Indication

IDHIT2	IDHIT1	IDHIT0	Identifier Acceptance Hit
0	0	0	Filter 0 hit
0	0	1	Filter 1 hit
0	1	0	Filter 2 hit
0	1	1	Filter 3 hit
1	0	0	Filter 4 hit
1	0	1	Filter 5 hit
1	1	0	Filter 6 hit
1	1	1	Filter 7 hit

The IDHIT indicators are always related to the message in the foreground buffer. When a message gets copied from the background to the foreground buffer, the indicators are updated as well.

NOTE

The CIDAC register can be written only if the SFTRES bit in CMCR0 is set.

16.12.10 msCAN12 Receive Error Counter

Address: $010E

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
Write:	▨	▨	▨	▨	▨	▨	▨	▨
Reset:	0	0	0	0	0	0	0	0

▨ = Unimplemented

Figure 16-25. msCAN12 Receive Error Counter (CRXERR)

This register reflects the status of the msCAN12 receive error counter. The register is read only.

Read: Anytime

Write: Has no meaning or effect

ADRxH[15:8]–ADRxH[7:0] — ATD Conversion Result Bits

The reset condition for these registers is undefined.

These bits contain the left-justified, unsigned result from the ATD conversion. The channel from which this result was obtained is dependant on the conversion mode selected. These registers are always read-only in normal mode.

17.4 ATD Mode Operation

Stop

Causes all clocks to halt (if the S bit in the CCR is 0). The system is placed in a minimum-power standby mode. This aborts any conversion sequence in progress. During STOP recovery, the ATD must delay for the STOP recovery time (t_{SR}) before initiating a new ATD conversion sequence.

Wait

ATD conversion continues unless the AWAI bit in ATDCTL2 register is set.

BDM

Debug options available as set in register ATDCTL3.

User

ATD continues running unless ADPU is cleared.

ADPU

ATD operations are stopped if ADPU = 0, but registers are accessible.

17.5 Using the ATD to Measure a Potentiometer Signal

This exercise allows the student to utilize the ATD on the HC12 to measure a potentiometer signal output routed from the UDLP1 board to the HC12 ATD pin PAD6. First the ATDCTL registers are initialized. A delay loop of 100 µs is then executed. The resolution is set up followed by a conversion set up on channel 6. After waiting for the status bit to set, the result goes to the D accumulator. If the program is working properly, a different value should be found in the D accumulator as the left potentiometer is varied for each execution of the program.

17.5.1 Equipment

For this exercise, use the M68HC912B32EVB emulation board.

17.5.2 Code Listing

NOTE

A comment line is delimited by a semi-colon. If there is no code before comment, an `";"` *must be placed in the first column to avoid assembly errors.*

18.4.1 Breakpoint Modes

Three modes of operation determine the type of breakpoint in effect.

1. Dual address-only breakpoints, each of which causes a software interrupt (SWI)
2. Single full-feature breakpoint which causes the part to enter background debug mode (BDM)
3. Dual address-only breakpoints, each of which causes the part to enter BDM

Breakpoints do not occur when BDM is active.

18.4.1.1 SWI Dual Address Mode

In this mode, dual address-only breakpoints can be set, each of which causes a software interrupt. This is the only breakpoint mode which can force the CPU to execute an SWI. Program fetch tagging is the default in this mode; data breakpoints are not possible. In dual mode each address breakpoint is affected by the respective BKALE bit. The BKxRW, BKxRWE, BKMBH, and BKMBL bits are ignored. In dual address mode, the BKDBE becomes an enable for the second address breakpoint.

18.4.1.2 BDM Full Breakpoint Mode

This is a single full-featured breakpoint which causes the part to enter background debug mode. BK1ALE, BK1RW, and BK1RWE have no meaning in full breakpoint mode.

BKDBE enables data compare but has no meaning if BKPM = 1. BKMBH and BKMBL allow masking of high and low byte compares but has no meaning if BKPM = 1. BK0ALE enables compare of low address byte.

- Breakpoints are not allowed if the BDM mode is already active. Active mode means the CPU is executing out of the BDM ROM.
- BDM should not be entered from a breakpoint unless the ENABLE bit is set in the BDM. This is important because even if the ENABLE bit in the BDM is negated, the CPU actually does execute the BDM ROM code. It checks the ENABLE and returns if not set. If the BDM is not serviced by the monitor, then the breakpoint would be re-asserted when the BDM returns to normal CPU flow. There is no hardware to enforce restriction of breakpoint operation if the BDM is not enabled.

18.4.1.3 BDM Dual Address Mode

This mode has dual address-only breakpoints, each of which causes the part to enter background debug mode. In dual mode, each address breakpoint is affected by the BKPM bit, the BKxALE bits, and the BKxRW and BKxRWE bits. In dual address mode, the BKDBE becomes an enable for the second address breakpoint. The BKMBH and BKMBL bits have no effect when in a dual address mode. BDM may be entered by a breakpoint only if an internal signal from the BDM indicates background debug mode is enabled. If BKPM = 1, then BKxRW, BKxRWE, BKMBH, and BKMBL have no meaning.

- Breakpoints are not allowed if the BDM is already active. Active mode means the CPU is executing out of the BDM ROM.
- BDM should not be entered from a breakpoint unless the ENABLE bit is set in the BDM. This is important because even if the ENABLE bit in the BDM is negated, the CPU actually does execute the BDM ROM code. It checks the ENABLE and returns if not set. If the BDM is not serviced by the monitor, then the breakpoint would be re-asserted when the BDM returns to normal CPU flow. There is no hardware to enforce restriction of breakpoint operation if the BDM is not enabled.

Chapter 20
Mechanical Specifications

20.1 Introduction

This section provides dimensions for the 80-pin quad flat pack (QFP).