Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
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 ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	80-QFP
Supplier Device Package	80-QFP (14x14)
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=mchc912b32vfue8

Table 1-3. Signal Description Summary

Pin Name	Pin Number	Description
PW3–PW0	3–6	Pulse-width modulator channel outputs
ADDR7–ADDR0 DATA7–DATA0	25–18	External bus pins share function with general-purpose I/O ports A and B. In single-chip modes, the pins can be used for I/O. In expanded modes, the pins are used for the external buses.
ADDR15–ADDR8 DATA15–DATA8	46–39	
IOC7–IOC0	16–12, 9–7	Pins used for input capture and output compare in the timer and pulse accumulator subsystem
PAI	16	Pulse accumulator input
AN7–AN0	58–51	Analog inputs for the analog-to-digital conversion module
$\overline{\text{DBE}}$	26	Data bus control and, in expanded mode, enables the drive control of external buses during external reads
MODB, MODA	27, 28	State of mode select pins during reset determines the initial operating mode of the MCU. After reset, MODB and MODA can be configured as instruction queue tracking signals IPIPE1 and IPIPE0 or as general-purpose I/O pins.
IPIPE1, IPIPE0	27, 28	
ECLK	29	E-clock is the output connection for the external bus clock. ECLK is used as a timing reference and for address demultiplexing.
$\overline{\text{RESET}}$	32	An active low bidirectional control signal, $\overline{\text{RESET}}$ acts as an input to initialize the MCU to a known startup state and an output when COP or clock monitor causes a reset.
EXTAL	33	Crystal driver and external clock input pins. On reset all the device clocks are derived from the EXTAL input frequency. XTAL is the crystal output.
XTAL	34	
$\overline{\text{LSTRB}}$	35	Low byte strobe (0 = low byte valid), in all modes this pin can be used as I/O. The low strobe function is the exclusive-NOR of A0 and the internal $\overline{\text{SZ8}}$ signal. The $\overline{\text{SZ8}}$ internal signal indicates the size 16/8 access.
$\overline{\text{TAGLO}}$	35	Pin used in instruction tagging
R/$\overline{\text{W}}$	36	Indicates direction of data on expansion bus; shares function with general-purpose I/O; read/write in expanded modes
$\overline{\text{IRQ}}$	37	Maskable interrupt request input provides a means of applying asynchronous interrupt requests to the MCU. Either falling edge-sensitive triggering or level-sensitive triggering is program selectable (INTCR register).
$\overline{\text{XIRQ}}$	38	Provides a means of requesting asynchronous non-maskable interrupt requests after reset initialization
BKGD	17	Single-wire background interface pin is dedicated to the background debug function. During reset, this pin determines special or normal operating mode.
TAGHI	17	Pin used in instruction tagging
DLCRx/RxCAN(1)	76	BDLC receive pin
DLCTx/TxCAN(1)	75	BDLC transmit pin
$\overline{\text{CS}}$/$\overline{\text{SS}}$	68	Slave-select output for SPI master mode; input for slave mode or master mode
SCK	67	Serial clock for SPI system
SDO/MOSI	66	Master out/slave in pin for serial peripheral interface
SDI/MISO	65	Master in/slave out pin for serial peripheral interface
TxD0	62	SCI transmit pin
RxD0	61	SCI receive pin

1. The RxCAN and TxCAN designations are for the MC68HC(9)12BC32 only.

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$0052	PWM Channel Duty Register 2 (PWDTY2) See page 135.	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
$0053	PWM Channel Duty Register 3 (PWDTY3) See page 135.	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
$0054	PWM Control Register (PWCTL) See page 136.	Read:	0	0	0	PSWAI	CENTR	RDPP	PUPP	PSBCK
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0055	PWM Special Mode Register (PWTST) See page 137.	Read:	DISCR	DISCP	DISCAL	0	0	0	0	0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0056	Port P Data Register (PORTP) See page 137.	Read:	PP7	PP6	PP5	PP4	PP3	PP2	PP1	PP0
		Write:								
		Reset:	U	U	U	U	U	U	U	U
$0057	Port P Data Direction Register (DDRP) See page 138.	Read:	DDP7	DDP6	DDP5	DDP4	DDP3	DDP2	DDP1	DDP0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0058	Reserved		R	R	R	R	R	R	R	R
↓										
$005F	Reserved		R	R	R	R	R	R	R	R
$0060	ATD Control Register 0 (ATDCTL0) See page 279.	Read: Write:	0	0	0	0	0	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$0061	ATD Control Register 1 (ATDCTL1) See page 279.	Read: Write:	0	0	0	0	0	0	0	0
		Reset:	0	0	0	0	0	0	0	0
$0062	ATD Control Register 2 (ATDCTL2) See page 279.	Read:	ADPU	AFFC	AWAI	0	0	0	ASCIE	ASCIF
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0063	ATD Control Register 3 (ATDCTL3) See page 280.	Read:	0	0	0	0	0	0	FRZ1	FRZ0
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$0064	ATD Control Register 4 (ATDCTL4) See page 281.	Read:	S10BM	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		Write:								
		Reset:	0	0	0	0	0	0	0	1

= Unimplemented R = Reserved U = Unaffected

Notes:
1. Available only on MC68HC912B32 and MC68HC912BC32 devices.
2. Available only on MC68HC912B32 and MC68HC12BE32 devices.
3. Available only on MC68HC(9)12BC32 devices.

Figure 2-1. Register Map (Sheet 5 of 19)

Addr.	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
$009A	Timer Input Capture/Output Compare 5 Register High (TC5H) See page 150.	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
$009B	Timer Input Capture/Output Compare 5 Register Low (TC5L) See page 150.	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
$009C	Timer Input Capture/Output Compare 6 Register High (TC6H) See page 151.	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
$009D	Timer Input Capture/Output Compare 6 Register Low (TC6L) See page 151.	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
$009E	Timer Input Capture/Output Compare 7 Register High (TC7H) See page 151.	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
$009F	Timer Input Capture/Output Compare 7 Register Low (TC7L) See page 151.	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
$00A0	Pulse Accumulator Control Register (PACTL) See page 151.	Read:	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
		Write:								
		Reset:	0	0	0	0	0	0	0	0
$00A1	Pulse Accumulator Flag Register (PAFLG) See page 153.	Read: Write:	0	0	0	0	0	0	PAOVF	PAIF
		Reset:	0	0	0	0	0	0	0	0
$00A2	Pulse Accumulator Count Register 3 (PACN3) See page 178.	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
$00A3	Pulse Accumulator Count Register 2 (PACN2) See page 178.	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
$00A4	Pulse Accumulator Count Register 1 (PACN1) See page 178.	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
$00A5	Pulse Accumulator Count Register 0 (PACN0) See page 178.	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

= Unimplemented R = Reserved U = Unaffected

Notes:
1. Available only on MC68HC912B32 and MC68HC912BC32 devices.
2. Available only on MC68HC912B32 and MC68HC12BE32 devices.
3. Available only on MC68HC(9)12BC32 devices.

Figure 2-1. Register Map (Sheet 10 of 19)

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 5
Operating Modes and Resource Mapping

5.1 Introduction

The MCU can operate in eight different modes. Each mode has a different default memory map and external bus configuration. After reset, most system resources can be mapped to other addresses by writing to the appropriate control registers.

5.2 Operating Modes

The states of the BKGD, MODB, and MODA pins during reset determine the operating mode after reset.

The SMODN, MODB, and MODA bits in the MODE register show current operating mode and provide limited mode switching during operation. The states of the BKGD, MODB, and MODA pins are latched into these bits on the rising edge of the reset signal. During reset an active pullup is connected to the BKGD pin (as input) and active pulldowns are connected to the MODB and MODA pins. If an open occurs on any of these pins, the device will operate in normal single-chip mode.

Table 5-1. Mode Selection

BKGD	MODB	MODA	Mode	Port A	Port B
0	0	0	Special single chip	General-purpose I/O	General-purpose I/O
0	0	1	Special expanded narrow	ADDR[15:8] DATA[7:0]	ADDR[7:0]
0	1	0	Special peripheral	ADDR DATA	ADDR DATA
0	1	1	Special expanded wide	ADDR DATA	ADDR DATA
1	0	0	Normal single chip	General-purpose I/O	General-purpose I/O
1	0	1	Normal expanded narrow	ADDR[15:8] DATA[7:0]	ADDR[7:0]
1	1	0	Reserved (forced to peripheral)	—	—
1	1	1	Normal expanded wide	ADDR DATA	ADDR DATA

The two basic types of operating modes are:

1. Normal modes — Some registers and bits are protected against accidental changes.
2. Special modes — Protected control registers and bits are allowed greater access for special purposes such as testing and emulation.

A system development and debug feature, background debug mode (BDM) is available in all modes. In special single-chip mode, BDM is active immediately after reset.

5.2.2.4 Special Peripheral Mode

The CPU is not active in this mode. An external master can control on-chip peripherals for testing purposes. It is not possible to change to or from this mode without going through reset. Background debugging should not be used while the MCU is in special peripheral mode as internal bus conflicts between BDM and the external master can cause improper operation of both modes.

5.2.3 Background Debug Mode

Background debug mode (BDM) is an auxiliary operating mode that is used for system development. BDM is implemented in on-chip hardware and provides a full set of debug operations. Some BDM commands can be executed while the CPU is operating normally. Other BDM commands are firmware based and require the BDM firmware to be enabled and active for execution.

In special single-chip mode, BDM is enabled and active immediately out of reset. BDM is available in all other operating modes, but must be enabled before it can be activated. BDM should not be used in special peripheral mode because of potential bus conflicts.

Once enabled, background mode can be made active by a serial command sent via the BKGD pin or execution of a CPU12 BGND instruction. While background mode is active, the CPU can interpret special debugging commands, read and write CPU registers, peripheral registers, and locations in memory.

While BDM is active, the CPU executes code located in a small on-chip ROM mapped to addresses $FF00 to $FFFF; BDM control registers are accessible at addresses $FF00 to $FF06. The BDM ROM replaces the regular system vectors while BDM is active. While BDM is active, the user memory from $FF00 to $FFFF is not in the map except through serial BDM commands.

BDM allows read and write access to internal memory-mapped registers and RAM and read access to EEPROM, FLASH EEPROM, or ROM without interrupting the application code executing in the CPU. This non-intrusive mode uses dead bus cycles to access the memory and in most cases will remain cycle deterministic. Refer to 18.3 Background Debug Mode (BDM) for more details.

5.3 Internal Resource Mapping

The internal register block, RAM, FLASH EEPROM/ROM, and EEPROM have default locations within the 64-Kbyte standard address space but may be reassigned to other locations during program execution by setting bits in mapping registers INITRG, INITRM, and INITEE. During normal operating modes, these registers can be written once. It is advisable to explicitly establish these resource locations during the initialization phase of program execution, even if default values are chosen, to protect the registers from inadvertent modification later.

Writes to the mapping registers go into effect between the cycle that follows the write and the cycle after that. To assure that there are no unintended operations, a write to one of these registers should be followed with a no operation (NOP) instruction.

If conflicts occur when mapping resources, the register block will take precedence over the other resources; RAM, FLASH EEPROM/ROM, or EEPROM addresses occupied by the register block will not be available for storage. When active, BDM ROM takes precedence over other resources, although a conflict between BDM ROM and register space is not possible. Table 5-2 shows resource mapping precedence.

In expanded modes, all address space not utilized by internal resources is by default external memory.

8.3.4 FLASH EEPROM Control Register

Address: $00F7

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	0	0	0	FEESWAI	SVFP	ERAS	LAT	ENPE
Reset:	0	0	0	0	0	0	0	0

Figure 8-4. FLASH EEPROM Control Register (FEECTL)

This register controls the programming and erasure of the FLASH EEPROM.

FEESWAI — FLASH EEPROM Stop in Wait Control Bit

0 = Do not halt FLASH EEPROM clock when in wait mode.
1 = Halt FLASH EEPROM clock when in wait mode.

NOTE

The FEESWAI bit cannot be asserted if the interrupt vector resides in the FLASH EEPROM array.

SVFP — Status V_{FP} Voltage Bit

SVFP is a read-only bit.

0 = Voltage of V_{FP} pin is below normal programming voltage levels.
1 = Voltage of V_{FP} pin is above normal programming voltage levels.

ERAS — Erase Control Bit

This bit can be read anytime or written when ENPE = 0. When set, all locations in the array will be erased at the same time. The boot block will be erased only if BOOTP = 0. This bit also affects the result of attempted array reads. See Table 8-1 for more information. Status of ERAS cannot change if ENPE is set.

0 = FLASH EEPROM configured for programming
1 = FLASH EEPROM configured for erasure

LAT — Latch Control Bit

This bit can be read anytime or written when ENPE = 0. When set, the FLASH EEPROM is configured for programming or erasure and, upon the next valid write to the array, the address and data will be latched for the programming sequence. See Table 8-1 for the effects of LAT on array reads. A high voltage detect circuit on the V_{FP} pin will prevent assertion of the LAT bit when the programming voltage is at normal levels.

0 = Programming latches disabled
1 = Programming latches enabled

ENPE — Enable Programming/Erase Bit

0 = Disables program/erase voltage to FLASH EEPROM
1 = Applies program/erase voltage to FLASH EEPROM

ENPE can be asserted only after LAT has been asserted and a write to the data and address latches has occurred. If an attempt is made to assert ENPE when LAT is negated, or if the latches have not been written to after LAT was asserted, ENPE will remain negated after the write cycle is complete. The LAT, ERAS, and BOOTP bits cannot be changed when ENPE is asserted. A write to FEECTL may affect only the state of ENPE. Attempts to read a FLASH EEPROM array location in the FLASH EEPROM module while ENPE is asserted will not return the data addressed. See Table 8-1 for more information.

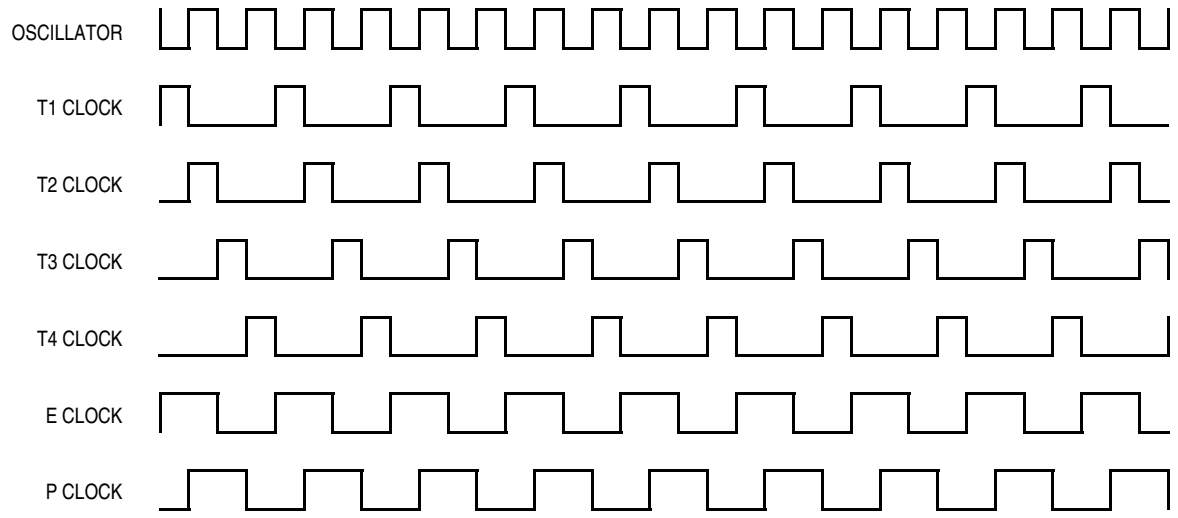
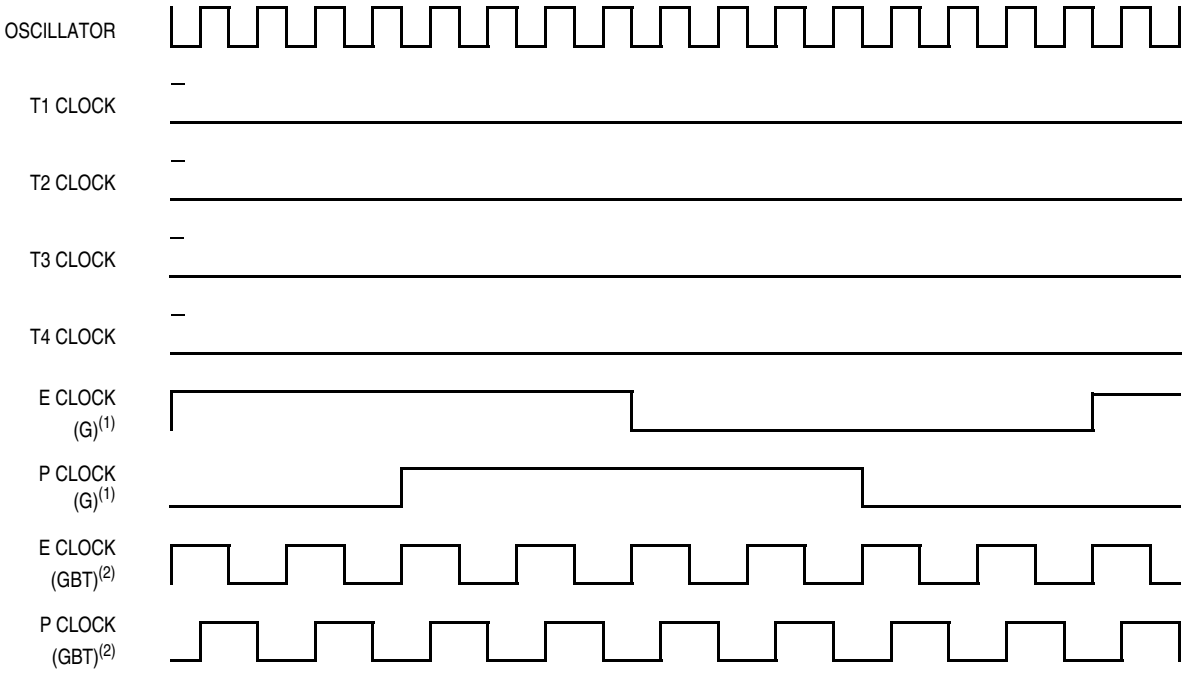


Figure 10-3. Internal Clock Relationships in Normal Run Modes



Notes:
1. Driven by slow clock divider in wait mode. Drives on-chip peripherals except BDLC and timer.
2. Remains at oscillator divided by 2 rate in wait mode. Drives BDLC and timer.

Figure 10-4. Internal Clock Relationships in Wait Mode

10.5 Slow Mode Divider

The slow mode divider is included to deliver a variable bus frequency to the MCU in wait mode. The bus clocks are derived from the constant P clock. The slow clock counter divides the P clock and E clock frequency in powers of 2, up to 128. When the slow control register is cleared or the part is not in wait mode, the slow mode divider is off and the bus clock's frequency is not changed.

NOTE

The clock monitor is clocked by the system clock (oscillator) reference; the slow mode divider allows operation of the MCU at clock periods longer than the clock monitor trigger time.

10.6 Clock Functions

The CGM generates and controls the timing of the reset and POR logic.

10.6.1 Computer Operating Properly (COP)

The computer operating properly (COP) or watchdog timer is an added check that a program is running and sequencing properly. When the COP is being used, software is responsible for keeping a free-running watchdog timer from timing out. If the watchdog timer times out, it is an indication that the software is no longer being executed in the intended sequence; thus, a system reset is initiated. Three control bits allow selection of seven COP timeout periods or COP disable. When COP is enabled, sometime during the selected period the program must write $55 and $AA (in this order) to the arm/reset COP register (COPRST). If the program fails to do this, the part resets. If any value other than $55 or $AA is written to COPRST, the part is reset.

10.6.2 Real-Time Interrupt

There is a real-time (periodic) interrupt available to the user. This interrupt occurs at one of seven selected rates. An interrupt flag and an interrupt enable bit are associated with this function. There are three bits for the rate select.

10.6.3 Clock Monitor

The clock monitor circuit is based on an internal resistor-capacitor (RC) time delay. If no MCU clock edges are detected within this RC time delay, the clock monitor can optionally generate a system reset. The clock monitor function is enabled/disabled by the CME control bit in the COP control register (COPCTL). This timeout is based on an RC delay so that the clock monitor can operate without any MCU clocks.

Clock monitor timeouts are shown in Table 10-1.

Table 10-1. Clock Monitor Timeout

Supply	Range
5 V ± 10%	2–20 µs

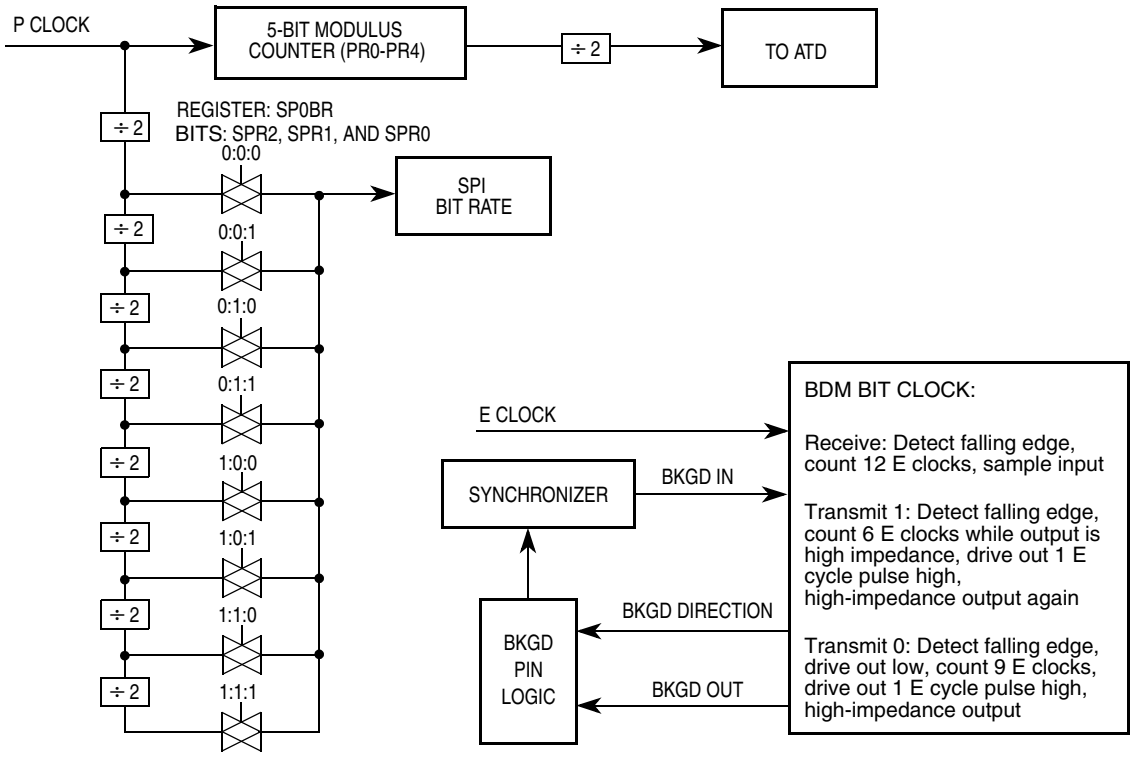


Figure 10-13. Clock Chain for SPI, ATD, and BDM

Address: $008A

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
Reset:	0	0	0	0	0	0	0	0

Figure 12-10. Timer Control Register 3 (TCTL3)

Address: $008B

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
Reset:	0	0	0	0	0	0	0	0

Figure 12-11. Timer Control Register 4 (TCTL4)

Read: Anytime

Write: Anytime

EDGnB and EDGnA — Input Capture Edge Control Bits

These 8 pairs of control bits configure the input capture edge detector circuits. See Table 12-2.

Table 12-2. Edge Detector Circuit Configuration

EDGnB	EDGnA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

12.3.7 Timer Interrupt Mask Registers

Address: $008C

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
Reset:	0	0	0	0	0	0	0	0

Figure 12-12. Timer Interrupt Mask 1 Register (TMSK1)

Read: Anytime

Write: Anytime

The bits in TMSK1 correspond bit-for-bit with the bits in the TFLG1 status register. If cleared, the corresponding flag is disabled from causing a hardware interrupt. If set, the corresponding flag is enabled to cause a hardware interrupt.

By setting TFMOD in queue mode, when NOVW bit is set and the corresponding capture and holding registers are emptied, an input capture event will first update the related input capture register with the main timer contents. At the next event, the TCn data is transferred to the TCnH register, the TCn is updated, and the CnF interrupt flag is set. See Figure 13-19. In all other input capture cases, the interrupt flag is set by a valid external event on PTn.

0 = The timer flags C3F–C0F in TFLG1 ($8E) are set when a valid input capture transition on the corresponding port pin occurs.

1 = If in queue mode (BUFEN = 1 and LATQ = 0), the timer flags C3F–C0F in TFLG1 ($8E) are set only when a latch on the corresponding holding register occurs. If the queue mode is not engaged, the timer flags C3F–C0F are set the same way as for TFMOD = 0.

PACMX — 8-Bit Pulse Accumulators Maximum Count Bit

0 = Normal operation. When the 8-bit pulse accumulator has reached the value $FF, with the next active edge, it will be incremented to $00.

1 = When the 8-bit pulse accumulator has reached the value $FF, it will not be incremented further. The value $FF indicates a count of 255 or more.

BUFEN — IC Buffer Enable Bit

0 = Input capture and pulse accumulator holding registers are disabled.

1 = Input capture and pulse accumulator holding registers are enabled. The latching mode is defined by LATQ control bit. Writing a 1 into ICLAT bit in MCCTL ($A6) when LATQ is set, will produce latching of input capture and pulse accumulator registers into their holding registers.

LATQ — Input Control Latch or Queue Mode Enable Bit

The BUFEN control bit should be set to enable the IC and pulse accumulators' holding registers. Otherwise, LATQ latching modes are disabled.

Writing one into ICLAT bit in MCCTL ($A6), when LATQ and BUFEN are set will produce latching of input capture and pulse accumulators registers into their holding registers.

0 = Queue mode of input capture is enabled. The main timer value is memorized in the IC register by a valid input pin transition. With a new occurrence of a capture, the value of the IC register will be transferred to its holding register and the IC register memorizes the new timer value.

1 = Latch mode is enabled. Latching function occurs when modulus down-counter reaches 0 or a 0 is written into the count register MCCNT (see 13.3.1.2 Buffered IC Channels). With a latching event the contents of IC registers and 8-bit pulse accumulators are transferred to their holding registers. The 8-bit pulse accumulators are cleared.

13.4.20 Timer Test Register

Address: $00AD

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	0	0	0	0	0	TCBYP	PCBYP(1)
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

1. Available only on MC68HC912B32 devices.

Figure 13-41. Timer Test Register (TIMTST)

Read: Anytime

Write: Only in special mode (SMOD = 1)

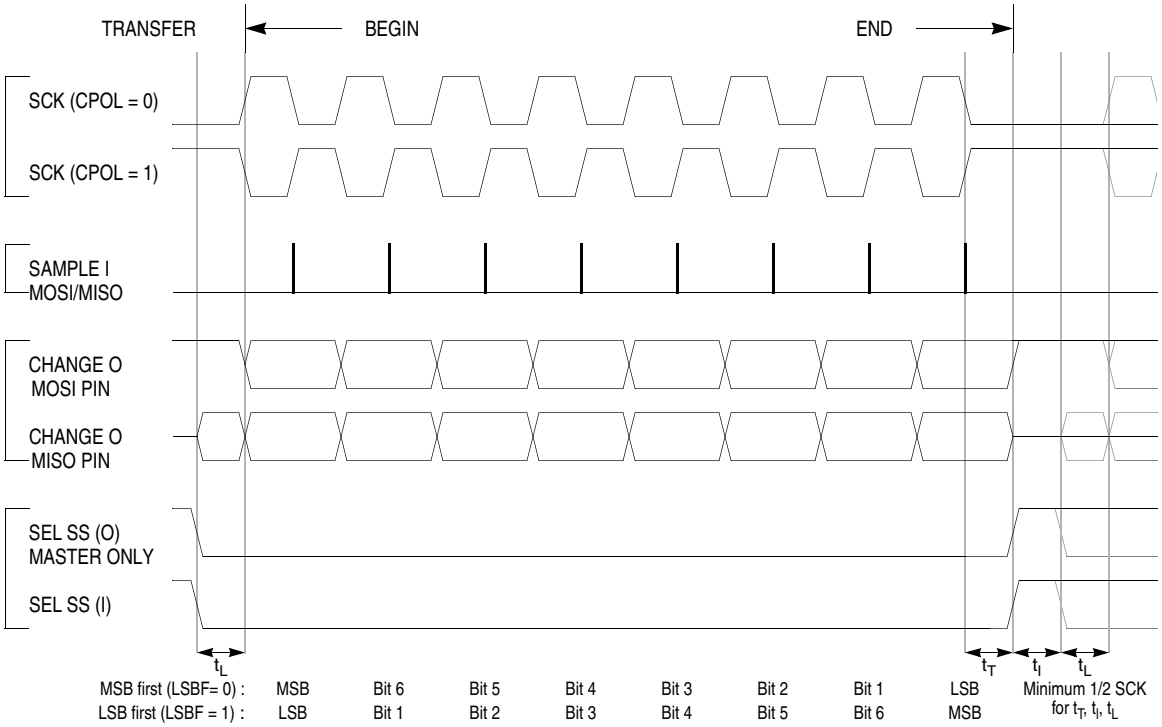


Figure 14-13. SPI Clock Format 1 (CPHA = 1)

14.3.3 $\overline{SS}$ Output

Available in master mode only, $\overline{SS}$ output is enabled with the SSOE bit in the SP0CR1 register if the corresponding DDRS bit is set. The $\overline{SS}$ output pin is connected to the $\overline{SS}$ input pin of the external slave device. The $\overline{SS}$ output automatically goes low for each transmission to select the external device and it goes high during each idling state to deselect external devices.

Table 14-3. $\overline{SS}$ Output Selection

DDS7	SSOE	Master Mode	Slave Mode
0	0	$\overline{SS}$ input with MODF feature	$\overline{SS}$ input
0	1	Reserved	$\overline{SS}$ input
1	0	General-purpose output	$\overline{SS}$ input
1	1	$\overline{SS}$ output	$\overline{SS}$ input

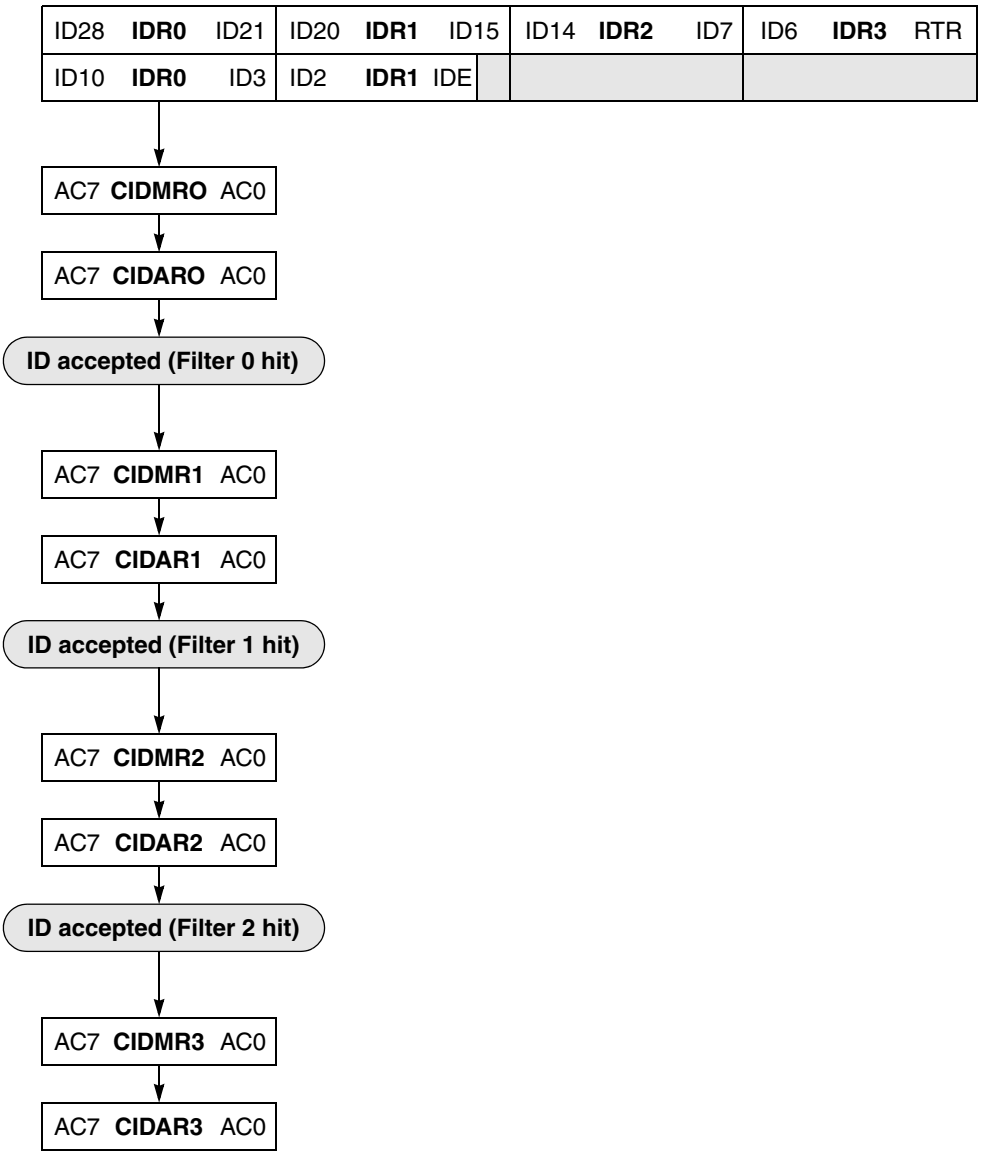


Figure 16-5. 8-Bit Maskable Acceptance Filters

CLKSRC — msCAN12 Clock Source Flag

This flag defines which clock source the msCAN12 module is driven from (only for system with CGM module. See **16.9 Clock System** and Figure 16-7.

0 = msCAN12 clock source is EXTALi.

1 = msCAN12 clock source is twice the frequency of ECLK.

NOTE

The CMCR1 register can be written only if the SFTRES bit in CMCR0 is set.

16.12.3 msCAN12 Bus Timing Register 0

Address: $0102

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	SJW1	SJW0	BRP5	BRP4	BPR3	BPR2	BPR1	BPR0
Reset:	0	0	0	0	0	0	0	0

Figure 16-18. msCAN12 Bus Timing Register 0 (CBTR0)

SJW1 and SJW0 — Synchronization Jump Width Bits

The synchronization jump width defines the maximum number of time quanta (Tq) clock cycles by which a bit may be shortened, or lengthened, to achieve resynchronization on data transitions on the bus (see Table 16-5).

Table 16-5. Synchronization Jump Width

SJW1	SJW0	Synchronization Jump Width
0	0	1 Tq clock cycle
0	1	2 Tq clock cycles
1	0	3 Tq clock cycles
1	1	4 Tq clock cycles

BRP5–BRP0 — Baud Rate Prescaler Bits

These bits determine the time quanta (Tq) clock, which is used to build up the individual bit timing, according to Table 16-6.

Table 16-6. Baud Rate Prescaler

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	Prescaler Value (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	1	64

NOTE

The CBTR0 register can be written only if the SFTRES bit in CMCR0 is set.

16.12.11 msCAN12 Transmit Error Counter

Address: $010F

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
Write:								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 16-26. msCAN12 Transmit Error Counter (CTXERR)

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

NOTE

Both error counters may be read only when in sleep or soft-reset mode.

16.12.12 msCAN12 Identifier Acceptance Registers

On reception, each message is written into the background receive buffer. The CPU is only signalled to read the message if it passes the criteria in the identifier acceptance and identifier mask registers (accepted); otherwise, the message will be overwritten by the next message (dropped).

The acceptance registers of the msCAN12 are applied on the IDR0 to IDR3 registers of incoming messages in a bit-by-bit manner.

For extended identifiers, all four acceptance and mask registers are applied. For standard identifiers only the first two (CIDMR0/CIDMR1 and CIDAR0/CIDAR1) are applied.

Address: $0110

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	Unaffected by reset							

Address: $0111

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	Unaffected by reset							

Address: $0112

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	Unaffected by reset							

Address: $0113

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset:	Unaffected by reset							

Figure 16-27. First Bank msCAN12 Identifier Acceptance Registers (CIDAR0–CIDAR3)

FRZ1 and FRZ0 — Background Debug (Freeze) Enable Bits

When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint is encountered. These two bits determine how the ATD will respond when background debug mode becomes active. See Table 17-1.

Table 17-1. ATD Response to Background Debug Enable

FRZ1	FRZ0	ATD Response
0	0	Continue conversions in active background mode
0	1	Reserved
1	0	Finish current conversion, then freeze
1	1	Freeze when BDM is active

17.3.5 ATD Control Register 4

Address: $0064

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	S10BM	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
Reset:	0	0	0	0	0	0	0	1

Figure 17-6. ATD Control Register 4 (ATDCTL4)

The ATD control register 4 (ATDCTL4) selects the clock source and sets up the prescaler. Writes to the ATD control registers initiate a new conversion sequence. If a write occurs while a conversion is in progress, the conversion is aborted and ATD activity halts until a write to ATDCTL5 occurs.

S10BM — ATD 10-Bit Mode Control Bit

0 = 8-bit operation
1 = 10-bit operation

SMP1 and SMP0 — Select Sample Time Bits

These bits are used to select one of four sample times after the buffered sample and transfer has occurred. See Table 17-2.

Table 17-2. Final Sample Time Selection

SMP1	SMP0	Final Sample Time	Total 8-Bit Conversion Time	Total 10-Bit Conversion Time
0	0	2 ATD clock periods	18 ATD clock periods	20 ATD clock periods
0	1	4 ATD clock periods	20 ATD clock periods	22 ATD clock periods
1	0	8 ATD clock periods	24 ATD clock periods	26 ATD clock periods
1	1	16 ATD clock periods	32 ATD clock periods	34 ATD clock periods

PRS4–PRS0 — Select Divide-By Factor for ATD P-Clock Prescaler Bits

The binary value written to these bits (1 to 31) selects the divide-by factor for the modulo counter-based prescaler. The P clock is divided by this value plus one, and then fed into a divide-by-two circuit to generate the ATD module clock. The divide-by-two circuit ensures symmetry of the output clock signal.

Program information is fetched a few cycles before it is used by the CPU. To monitor cycle-by-cycle CPU activity, it is necessary to externally reconstruct what is happening in the instruction queue. Internally, the MCU only needs to buffer the data from program fetches. For system debug it is necessary to keep the data and its associated address in the reconstructed instruction queue. The raw signals required for reconstruction of the queue are ADDR, DATA, R/$\overline{W}$, ECLK, and status signals IPIPE1 and IPIPE0.

The instruction queue consists of two 16-bit queue stages and a holding latch on the input of the first stage. To advance the queue means to move the word in the first stage to the second stage and move the word from either the holding latch or the data bus input buffer into the first stage. To start even (or odd) instruction means to execute the opcode in the high-order (or low-order) byte of the second stage of the instruction queue.

18.3 Background Debug Mode (BDM)

Background debug mode (BDM) is used for system development, in-circuit testing, field testing, and programming. BDM is implemented in on-chip hardware and provides a full set of debug options.

Because BDM control logic does not reside in the CPU, BDM hardware commands can be executed while the CPU is operating normally. The control logic generally uses CPU dead cycles to execute these commands, but can steal cycles from the CPU when necessary. Other BDM commands are firmware based and require the CPU to be in active background mode for execution. While BDM is active, the CPU executes a firmware program located in a small on-chip ROM that is available in the standard 64-Kbyte memory map only while BDM is active.

The BDM control logic communicates serially with an external host development system, via the BKGD pin. This single-wire approach minimizes the number of pins needed for development support.

18.3.1 BDM Serial Interface

The BDM serial interface requires the external controller to generate a falling edge on the BKGD pin to indicate the start of each bit time. The external controller provides this falling edge whether data is transmitted or received.

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 E-clock cycles per bit (nominal speed). The interface times out if 512 E-clock cycles occur between falling edges from the host. The hardware clears the command register when this timeout occurs.

The BKGD pin can receive a high or low level or transmit a high or low level. Figure 18-1, Figure 18-2, and Figure 18-3 show timing for each of these cases. Interface timing is synchronous to MCU clocks but asynchronous to the external host. The internal clock signal is shown for reference in counting cycles.

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.

19.12.2 Example V_{FP} Protection Circuitry

Figure 19-2 shows an example of a circuit which, if properly implemented, can maintain the appropriate voltage levels on the V_{FP} pin. This section outlines the design for this circuit, what each component is intended to do, and some design considerations when designing V_{FP} pin protection.

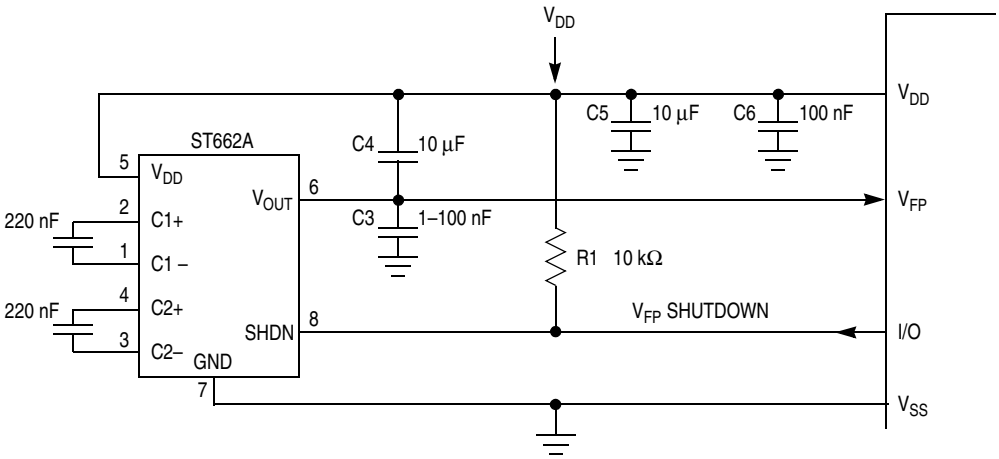


Figure 19-2. V_{FP} Supply Circuit

The general idea of this circuit implementation is to supply V_{FP} from a dc-dc converter. This dc-dc converter, like most, provides a shutdown feature which allows the converter's output to be shut off. When the SHDN pin on the converter is pulled high, as the 10-kΩ pullup resistor (R1) does, the output V_{Out} is shorted to the V_{DD} supply. This requires that the programming and erasing routines assert a port pin on the MCU to turn on the converter and supply the 12-volt programming voltage during the programming or erasing cycle. Simple programming and erasing routines, such as those shown earlier in this application note, will no longer suffice.

By implementing this solution, V_{FP} is tied to V_{DD} on power-up and power-down, ensuring that they rise and fall together. Capacitors C5 and C6 are the normal decoupling capacitors on the V_{DD} supply lines. C3 is used to reduce electromagnetic interference (EMI) in the circuit. If C3 is too large, V_{FP} will not be allowed to fall with V_{DD}, potentially causing data corruption in the FLASH array. (Refer to Figure 19-3.) C4 is where the dc-dc converter stores charge to supply V_{Out} to the target device. The supply must be able to source approximately 30 mA of current for at least 20 µs (based on programming cycle requirements) and 4 mA of current for at least 10 ms (based on erase cycle requirements).

A certain degree of experimentation might be required when selecting C4 and C3. When trying different capacitor values, always monitor the effects on V_{FP} decay during power-down and current supplied to the V_{FP} pin.

R1 must be no larger than 10 kΩ, to make certain that the SHDN pin on the dc-dc converter is never allowed to fall below V_{DD} unless the output pin of the microcontroller is driven low. The external pullup ensures this behavior, no matter what port pin is used on the microcontroller or what the internal structure of that pin looks like. Without a strong enough pullup resistor on R1, the voltage on the SHDN pin might drop during a reset event, causing the dc-dc converter to activate and begin driving the voltage on V_{Out} to begin to rise to 12 volts. This would result in data corruption in the FLASH.